

Proceedings of the 24th Philippine Computing Science Congress Volume I: Main Conference

Editors: Dr. Judith J. Azcarraga, Dr. Ethel Chua Joy Ong & Mark Edward M. Gonzales

Copyright © 2024 Computing Society of the Philippines

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without the prior written permission of the copyright owner. Individual papers may be uploaded to institutional repositories or other academic sites for self-archival purposes.

ISSN 1908-1146



Computing Society of the Philippines
csp.org.ph



Contents

About the Philippine Computing Science Congress	1
About the Computing Society of the Philippines	1
Organizing Committee	2
Accepted Papers	
Incorporating a Computer Vision System to a Web Server to Parse Real-time Data for Valorant Esports <i>Jan Jozef R. Laguer & Jenilyn A. Casano</i>	3
Predicting Student Performance Based on Moodle Forum Interaction Logs <i>Maya Louise E. Asuero, Angelu Ferdinand A. Garcia, Julius Czar Makiling & Jun Rangie C. Obispo</i>	11
Factoring a Quadratic from a Degree N Polynomial Using Newton's Method <i>Pablo R. Manalastas</i>	19
ChattyTicket: Classifying Emotion and Toxicity in Valorant Chats using Multi-Task Learning with Bi-LSTM Algorithm and BERT <i>Johndel N. Encabo, Maria Cassandra B. Vitug & Tita R. Herradura</i>	25
Utilizing Machine Learning Methods to Predict Student Re-engagement via Minecraft Data <i>Mylene Villegas</i>	32
Exploring Photogrammetry and Neural Radiance Fields for Preserving Baguio Statues and Monuments <i>Yohann C. Gayao, Bill R. Malitao & Christella Marie R. Ocampo</i>	40
AniFrame: A Programming Language for 2D Drawing and Frame-Based Animation <i>Mark Edward M. Gonzales, Hans Oswald A. Ibrahim, Elyssia Barrie H. Ong & Ryan Austin Fernandez</i>	46
Comparative Analysis of Heuristics for the Offline Food Delivery Problem on Starlike Graphs <i>Jose Alfonso Barreiro, Mikael Giannes Bernardino, Erick Gabriel Lopez & John Paul Vergara</i>	54
Performance of various protein representations for predicting phage-host interaction <i>Franz Stewart V. Dizon, Jennifer C. Ureta & Anish M.S. Shrestha</i>	62
Coronavirus Herd Immunity Optimizer with Tent and Circle Chaotic Maps for the p -Center Facility Location Problem: The Case of Davao City, Digos City and Tagum City, Philippines <i>Hanna Mae Lina Limpag & Ritchie Mae T. Gamot</i>	70
Data-centric Advanced Knowledge Interface for Legal Archives: An NLP-Enhanced Tax Statute Text Search System in the Philippines <i>Matthew Roque, Shirley Chu, Melvin Cabatuan, Nicole Abejuela, Michael Ng, Hans Nolasco & Jenine Valencia</i>	77

Evaluating ChatGPT for Joint Intent Detection and Slot Filling: Zero-Shot vs. Few-Shot Prompting <i>Hannah Fong & Ethel Ong</i>	85
Performance Evaluation: Triplet Loss Implementation on AlexNet With MTCNN In Detecting Deepfake <i>Rich Tristan Lim, Angie Ceniza-Canillo & Raymond Anthony Aya-ay</i>	93
Summarization-Driven Collaborative Filtering for Explainable Recommendation <i>Reinald Adrian Pugoy & Hung-Yu Kao</i>	99
ErrgoEngine: A Contextualized Programming Error Analysis Translation Engine <i>Ned Isaiah Palacios, James Bryan Parcasio, Joseph Anthony Pornillos, Rogelio Badiang Jr. & Aurora Cristina Manseras</i>	107
Immune Survivors: A Serious Game for Raising Immune System Awareness <i>Junma Ogawa, Setiel Olivarez, James Gabriel Omalin, Gil Perfecto Noel Alva, Ian Joshua Nemeño & Thomas James Tiam-Lee</i>	115
An Arduino-Based Wearable Gesture Device as a Controller for Quadcopter Drones <i>Clement Ong, Jon Bryan Dela Cruz, Yasmin Audrey Tutanés Datarío, Juan Christian Geronimo, Justin Thomas Lanting & Lester Andrei Opulencia</i>	123
Halogen: A 2D Simulation Game on Dental Health Challenges in the Philippines <i>Julliana Zarah Cruz, Norberto Tadeo & John Ryan Tamano</i>	131
Formal Verification of Shortest Job First Scheduling Algorithm in Coq <i>Jian Lawrence Luteria, Earl Wilbur Nogra, Andrei Tiangco, Alfonso B. Labao & Henry N. Adorna</i>	139
AI Skill Check: An Examination of ChatGPT’s Consistency in Linguistics and Mathematics using the SAT <i>Ma. Elisha Kaye B. Nazario, Eliza Janel L. Tan, Steven Vincent D. Lim, Keith Alexandre L. Ramos & Shirley B. Chu</i>	145
Formal Verification of a Custom Scheduling Algorithm and Specification of a Round-Robin Implementation using Coq <i>Christian Choa, Brylle Logroño, Raymart Villos, Alfonso B. Labao & Henry N. Adorna</i>	153
Modeling Perceptual Organization in Abstract Art Using Eye Tracking Data <i>Carlo Ram M. Ferrer & Maria Mercedes T. Rodrigo</i>	159
Author Index	167
Institution Index	168



24th PHILIPPINE COMPUTING SCIENCE CONGRESS (PCSC 2024)

May 9–11, 2024 · De La Salle University – Laguna Campus

About the Philippine Computing Science Congress

The **Philippine Computing Science Congress (PCSC)** is organized by the Computing Society of the Philippines to enable local and neighboring computing educators, researchers, information and communications technology (ICT) professionals, and students to interact and share their work in computing, computer science, computational science, and ICT. The conference features special lectures by prominent researchers and educators and contributed papers in ICT, computing, computer science, computational science, and related disciplines.

About the Computing Society of the Philippines

The **Computing Society of the Philippines (CSP)** is an organization promoting research and development in computing science, computer science, computer engineering, computational science, computing education, and related disciplines by promoting the exchange of knowledge in this field, especially at the tertiary and graduate levels. CSP's main objective is to advocate progressive policies and develop programs that affect the computing sector. CSP establishes linkages with other organizations in the pursuit of common goals.



24th PHILIPPINE COMPUTING SCIENCE CONGRESS (PCSC 2024)

May 9–11, 2024 · De La Salle University – Laguna Campus

Organizing Committee

Program Chair and Local Host Chair

Judith J. Azcarraga, Ph.D.
De La Salle University

Student Research Workshop Chair

Ethel Chua Joy Ong, Ph.D.
De La Salle University

Organizing Committee Members

Ann Franchesca B. Laguna, Ph.D.
De La Salle University

Neil Justin V. Romblon
De La Salle University

Michelle Renee D. Ching, DIT
De La Salle University

Candy Joyce H. Espulgar
De La Salle University

Organizers



Computing Society of the Philippines



Association for Computing Machinery Women in Computing



College of Computer Studies, De La Salle University

Incorporating a Computer Vision System to a Web Server to Parse Real-time Data for Valorant Esports

Jan Jozef R. Laguer
Ateneo de Manila University
Quezon, Metro Manila
jan.laguer@student.ateneo.edu

Jenilyn A. Casano
Ateneo de Manila University
Quezon, Metro Manila
jagapito@ateneo.edu

ABSTRACT

There is currently no tool that allows broadcasting teams to visualize in-game events in Valorant. Unlike its competitor, Counter Strike, Valorant supports no such feature that allows a software to request information pertinent to Esports broadcasting. This paper made a system to get information from Valorant by making use of computer vision to parse frames and web servers to distribute it. Previous systems were explored to serve as a foundation and to see if any improvements can be made. A Convolutional Neural Network and Optical Character Recognition engine were used to perform Image Classification and text extraction, respectively. Although a lot of information can be extracted, there exists data that is not visually represented and cannot be extracted. Due to the nature of the game being a fast-paced tactical shooter, there is a need for the tool to be able to parse and deliver the data at a high speed. The system was tested on its execution speed and its accuracy by testing the individual modules and the system as a whole. After collecting the needed metrics, it is found that the tool is accurate and is able to return data at a near real-time speed.

KEYWORDS

Neural networks, web server, esports, broadcasting

1 INTRODUCTION

Esports tournaments are important to gaming culture. Kerttula has said "Online streams and even some television shows create a spectacular atmosphere around these tournaments and the best players are now celebrities" [15]. These tournaments help grow the games and communities they are a part of and entice new people to join in the event. Events hosted by the game's creators especially have high showing. The PGL Major 2021 and the International 2021, a Counter Strike: Global Offensive (CS:GO) tournament and a DOTA 2 tournament, respectively, both hosted by Valve had an average of 590 thousand and 850 thousand viewers, respectively [6, 7]. More recently, the Valorant Champions Tour (VCT), a Valorant tournament hosted by Riot Games, had an average of 460 thousand viewers [8].

While Valorant has grown from the VCT, the community has also helped grow the player base through their own tournaments. This includes different content creators and communities like Magu-Cup and ValorantPH's public games. AcadArena, an organization for Campus Gaming and Esports Education in Southeast Asia [1], hosted their tournament, the Alliance Games, which included Valorant where the finals boasted 26 thousand views not including the live audience [2].

With how young the game is, there are many features present in other esports titles that are not present yet in the current game

itself. One such feature which this study focused on is a publicly available method to get data on a currently active game. CSGO has this feature called Game State Integration (GSI) which can expose "all game state, and send an update notification as soon as the client game state changes, to any local or remote HTTP POST endpoint using JSON as the game state structure" [27]. Currently, Valorant has no feature like this found on the official developer dashboard [9]. This feature is what powers other games' broadcasting tools use for their visualizations.

The goal of this study is to create a tool in order to address not being able to access the needed data as well as elevate the community's broadcasts and production similar to other tools used for other games to be similar to VCT. To fulfill this goal, the tool uses computer vision, machine learning, and web development. Another goal is to evaluate it to ensure that it is both accurate and has fast execution speed because this would be used in an environment where it is important to have both accurate and timely information.

To evaluate the tool, it must meet certain standards. One standard, accuracy, comes from ISO 5725. This describes accuracy as the combination of both precision and "trueness" [12]. This means that to be accurate, the evaluation must be close to the actual value and the spread of points must be close together. Another standard is that events that happened in the game must reflect on the tool with little delay. The Advanced Television Systems Committee (ATSC) has established synchronization limits where the delay is 45 ± 15 ms [16]. Additionally, the concept of real-time is often associated with the lack of delay and synchronization between two things. However, the term real-time is not strictly defined. One paper has described their tool, VisWiz as near real-time with the delay being less than 20 seconds [3] while another described real-time by stating how many frames can be processed within a second or frames per second (FPS) which in this case was between 24 and 60 FPS [14]. This is roughly a delay between 16ms to 41ms.

2 REVIEW OF RELATED LITERATURE

Each game has its own way of giving information which include the player's state in the game, current standings, and other elements specific to a certain game to spectators. For an in-game example, DOTA 2 allows spectators of a game to access different statistics that players in game have no access to. Other games, instead of having this type of tool built into the game, allow getting information about a current match outside the game. An example of this would be CS:GO's GSI Integration as this opens up a web server on a computer currently running CS:GO and exposes "all game state, and send an update notification as soon as the client game state changes, to any local or remote HTTP POST endpoint using JSON as the game state structure" [27].

DOTA 2 provides different statistics that can be shown like net worth, experience rate, and amount of last hits and denies. By default, each option shows the raw numbers for each statistic but can be transformed into a graph in order to make it easier to read and find patterns. Most of these statistics and visualizations aren't available to the players playing. At most, the players only have the statistics for their own team and selves' kills, deaths, assists, and current gold.

Production tools will be defined as tools that allow a broadcaster to visualize something to show in broadcast automatically. These tools make use of data in order to show a broadcast's audience pertinent data. For the case of esports, these tools visually display a game's current state at certain times or when the production team deems it necessary. A popular tool that is used in the CS:GO tournament scene is Lexograine [17]. This tool takes in data, parses it, and displays it depending on the event received from the GSI server. The GSI server makes the data available for both teams; however, a player will only have their team's data available at any time. The data for the other team is only updated during the start of each round. The figure also shows some data that is not received from the GSI server. These data include the team names and team logos, where Lexograine handles the state of each one.

Valorant has currently no way of getting this data for a normal user, as there's no in-game solution like DOTA 2 or an outside solution like in CS:GO. This means that production tools for Valorant are scarcely available. The reasons include a lack of an official implementation, the age of Valorant, and the difficulty of real-time data scraping itself. In order to build such a tool, there is a need to take a look at other Production Tools, especially those relating to Valorant. The lack of a production tool to assist those wanting to broadcast their community's games results in a big difference in audience experience.

2.1 Similar Systems

The closest application to this study is the Lexograine HUD Manager. The backend for Lexograine was written in JavaScript and used node.js and express.js for the server [17]. Lexograine also has a local database using NeDB.js to store any local data [17]. This meant that each broadcaster had to run their own instance of the Lexograine HUD Manager. The communication between the server and the HUD is mainly via a notification that a change in data has occurred via a websocket connection. This is followed by sending a GET request to the server for the new data.

Another 2 projects that aim to create production tools despite Valorant's lack of data are deepsidh9's Live-Valorant-Overlay [5] and tugamars' AcheronObs [26]. Each project used a Python server that used computer vision to extract data from the screen. Where each project differed was in the frontend, as they use either a JavaScript and HTML frontend or a Java frontend. It is of note that these 2 projects are no longer being maintained, as both projects have not been updated in at least 2 years as of March 2023.

A common point for these two project's backend is the use of a web server in order to deliver the data to the frontend. This is very similar to CS:GO's GSI feature though it was not written whether the developers used that as an inspiration for this backend. Both projects use Python to create the backend though the framework for

the web server for each project differs. deepsidh9's implementation uses the Flask framework, while tugamars uses a newer framework, FastAPI.

These projects have different methods in order to deliver the data to the frontend. deepsidh9 implemented a web-socket using Socket.io as the main method of sending data. This implementation lets the backend control when the data gets sent. tugamars has a different approach wherein the frontend sends a request which triggers the image processing. This implementation lets the frontend control when the data gets sent.

The backend for both these projects have the same role, getting data from the current Valorant window, though each project has a different way of doing it. Tugamars' implementation uses another program to capture Valorant in order to start the scraping process. The program is called OBS, which is an open source software for capturing screens and or windows and broadcasting to the internet [23]. This method requires OBS to be less than version 28 as the plugin used by tugamars is not compatible with the later versions. deepsidh9 has a different approach as it uses the Windows API library in Python in order to access the Valorant window itself. After some experimentation, this method does not seem to work for the current release of Valorant, as it is possible the way Valorant renders its screen has changed. After capturing each frame, the backend then parses it using computer vision via the OpenCV Python library.

2.2 Computer Vision Implementation in the Similar Systems

Each project goes about this process differently. deepsidh9's implementation was able to collect information such as a player's life status, shield type, primary weapon, and ultimate status, and current game score and spike status [5]. tugamars' implementation has a shorter list with only being able to get spike status, player's hp value, and whether a player has their ultimate up or not [26]. The reason for this discrepancy is that the former was able to scrape data from the scoreboard, while the latter had not implemented that feature. Additionally, the latter's implementation required the observer to go to an area of the map where the screen has only one color, black.

The advantage of this was the ability to get a numerical value for the health of a player, as there is currently only one element found on screen relating to the health value of a player. This element is a bar that decreases and reveals the background as the value becomes smaller. Due to the element's property of having a transparent background, collecting hp data can become inaccurate as the background could lead to the element looking like it is full when in actuality it is not. The disadvantage for this method is the computer scraping data cannot be used as an observer for the game, as the only thing the computer will see using this method is just a black background along with the in-game HUD. This would require another computer to act as an observer in the game that will receive data from the backend server. As deepsidh9's implementation does not use this method, only one computer would be needed in order to fully utilize the whole production tool, at the cost of not being able to get numerical values for HP.

Another area where these projects differ is how each project handles optical character recognition. deepsidh9 uses the python library EasyOCR while tugamars uses Tesseract [5], [26]. According to Liao's experimentation, EasyOCR is more accurate when compared to Tesseract when it comes to recognizing numbers, while the latter is more accurate when it comes to recognizing alphabet characters [18]. They also mention that Tesseract's processing time is faster compared to EasyOCR when it is done on a CPU [18].

Template matching is a technique absent from tugamars implementation and present in deepsidh9's. They use this technique to analyze and figure out what images represent. Specifically, they use this in order to figure out each player's character, armor type, and primary weapon. This can be further optimized as the current implementation loops through a set of templates for each character, weapon, and armor types. In the worst case, it's possible that the correct template would be at the end of the loop, which makes the system perform suboptimally. A notable use of template matching was detecting if the scoreboard was visible on screen, as there's only one template that needs to be checked.

3 METHODOLOGY

3.1 Programming Language

In fulfillment of this study's objectives, the researcher has outlined the use of Javascript and Python as programming languages.

The choice of JavaScript is due to its prevalent use in web applications and is, as Crockford wrote, the "Language of the Web" [4]. The reason for the need of a browser-based application is that OBS has a feature that allows for them to be integrated into the program. This integration allows the use of transparent backgrounds and elements for the frontend. While using chroma keys can achieve the same transparent backgrounds and elements, this method requires that a certain color is removed entirely from the UI. This would prove not possible as there are numerous agents and may result in an agent having transparent elements which would be distracting from the game that this HUD is supposed to be enhancing. In using this programming language, the application is able to communicate with a server and is able to change different elements based on what the application receives. This acts as the HUD that displays data that it receives from the server.

The choice of Python is due to the programming language's different modules that includes different web development frameworks [24] and computer vision and image processing tools from OpenCV [20]. Web development frameworks were mentioned due to how compatible they are with web applications. The compatibility comes from the server included in these web development frameworks, which enables communication between the said server and the web application as the client [19]. Computer vision is a requirement as this is the main method in order to get data from Valorant's game screen due to the game's unavailability of a method to get current game state [9]. All in all, this allows getting data from Valorant's game screen and sending it to the HUD.

Python was also chosen in order to create the image classification models. As mentioned previously, there are many libraries and modules in the Python ecosystem, and the library that was used to create an image classification model would be TensorFlow. According to Johnson, TensorFlow is an open-source end-to-end

platform for creating Machine Learning applications and is focused on training and inference of deep neural networks [13].

3.2 Image Classification Model Creation

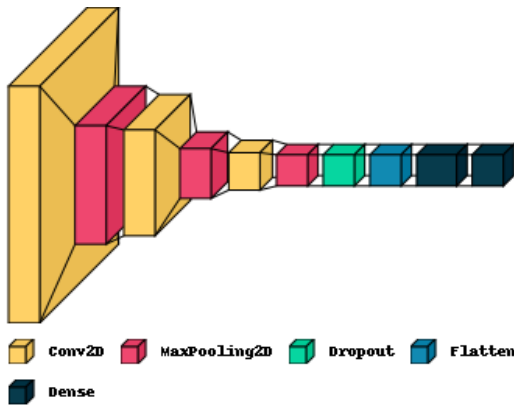
There are currently two visual elements that showcase an image in order to represent something. With this in mind, the researcher would need to create two image classification models, one to identify a player's character and another to identify a player's primary weapon. Using TensorFlow, the models would be created using a Convolutional Neural Networks (CNN). O'Shea and Nash wrote that CNNs are mainly used for their pattern recognition in images making them suitable to image focused tasks [21]. Another method would use Template Matching similar to what previous projects used but in Hmdaoy and Ahmed's study comparing the two, they have found that using a CNN was both faster and more accurate compared to template matching [11]. CNNs commonly use 4 layers, the input layer, the Convolutional Layer, the Pooling Layer, and the Fully-connected Layer.

3.2.1 CNN Architecture. The Architecture of the CNN for creating the models would follow a common way to set it up. The most common way of setting up the CNN Architecture is to set a Convolutional Layer followed by a Pooling Layer multiple times, then following it up with Fully-connected Layers. The specific Architecture for this thesis' CNN would be as follows: an input layer, three pairs of Convolutional Layer with rectified linear (ReLU) activation and Pooling Layer, A Flatten Layer to transform the image to a 1 dimensional array, A Fully-connected Layer with ReLU activation, and finally a Fully-connected Layer with softmax activation. In order to prevent overfitting, a Dropout Layer with a rate of 20% was added after the last Pooling Layer. Refer to Figure 1 for the visualizations of each model.

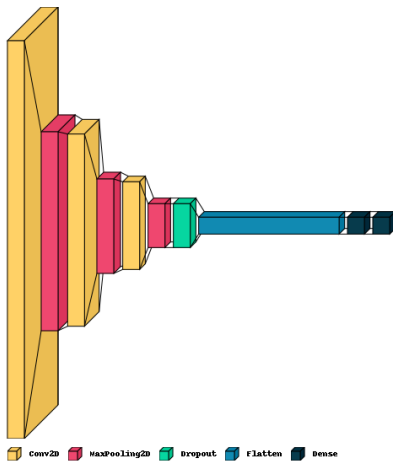
The models are mostly similar with some differences. The convolution layers had a kernel size of 3x3 with "relu" activation. The agent model had 16, 32, 16 filters while the weapon model had 16, 32, 64. Both models have the same pooling layers with a pool size of 2x2. The second to last dense layer has 256 units for the agent model while the weapon model had 128. Refer to Figure 2

3.2.2 Collection of Training Images. Before creating the CNN to create the model, the training images would need to be created first. The researcher first collected the display icons with transparent backgrounds for each Valorant agent. Figure 3b shows an example of a Valorant agent picture. The researcher then created a Python script to put each image in a directory named for the Valorant agent they represented. In each directory, the display icons were resized to a 40 pixel by 40 pixel image, mirrored, and had different background colors applied. The specific colors applied are as follows: #26493f, #5a221f, #494834, #53755d, #9aa36f, #90a366, white, black, blue, red, #307368, #5b2929, #285e54, #4a2323. After that, each image was copied and then resized to a 30 pixel by 30 pixel image. The resizing of pictures is done because there are visual elements that are these sizes and was made to ensure the model has no problem interpreting them regardless of size.

The collection of the training images for the weapon classifier had a different methodology. A custom game with the scoreboard in constant view was recorded in order to capture the different



(a) Agent Model



(b) Weapon Model

Figure 1: Visualization of Models using visualkeras [10]

weapon icons. The recording contained footage of the researcher traversing the map with the scoreboard in the center of the screen. For consistency, the same route was done for all weapons. The game was recorded in 1920x1080 resolution with 60 frames per second. Using a Python script, frames from the recording were extracted by getting every 6th frame and cropping the frame to only the cell in the scoreboard that shows the primary weapon of the player. Figure 3a shows an example screenshot. These images were saved in a directory with the name of the weapon it is representing as the name.

The dataset for the agent and weapon classifiers resulted in 5128 and 5648 images respectively. The dataset for the agent classifier resulted in 20 classes with an average of 256 images per class. The dataset for the weapon classifier resulted in 18 class with an average of 300 images per class.

3.3 OCR System

The Python module and library ecosystem allows for convenient installations for OCR. One of the previously used OCR Systems,

Name (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 38, 38, 16)	448
max_pooling2d (MaxPooling2D)	(None, 19, 19, 16)	0
conv2d_1 (Conv2D)	(None, 17, 17, 32)	4640
max_pooling2d_1 (MaxPooling2D)	(None, 8, 8, 32)	0
conv2d_2 (Conv2D)	(None, 6, 6, 16)	4624
max_pooling2d_2 (MaxPooling2D)	(None, 3, 3, 16)	0
dropout (Dropout)	(None, 3, 3, 16)	0
flatten (Flatten)	(None, 144)	0
dense (Dense)	(None, 256)	37120
dense_1 (Dense)	(None, 20)	5140

(a) Agent Model Summary

Name (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 30, 118, 16)	448
max_pooling2d (MaxPooling2D)	(None, 15, 59, 16)	0
conv2d_1 (Conv2D)	(None, 13, 57, 32)	4640
max_pooling2d_1 (MaxPooling2D)	(None, 6, 28, 32)	0
conv2d_2 (Conv2D)	(None, 4, 26, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 2, 13, 64)	0
dropout (Dropout)	(None, 2, 13, 64)	0
flatten (Flatten)	(None, 1664)	0
dense (Dense)	(None, 128)	213120
dense_1 (Dense)	(None, 18)	2322

(b) Weapon Model Summary

Figure 2: Model Summary



(a) Example Screenshot



(b) Example Agent

Figure 3: Example of Image Collection

Tesseract, has a different way to install it. This installation involves downloading a separate binary to ensure the respective module

works. To simplify the process, another OCR system would be chosen on the basis that it is on par or better in terms of accuracy and speed and has a more simple installation process.

The OCR Engine that was used for development is PaddleOCR which can be installed via pip, the built-in way to install packages in Python environments. It self-documents as a "practical ultra-lightweight OCR system" with considerations with regard to the balance of accuracy and speed [22]. The process for this system includes first detecting texts and setting up the bounding boxes and then performing text recognition on each of those boxes by using Differentiable Binarization (DB) and Convolutional Recurrent Neural Network (CRNN) [22].

Once the output has been generated from PaddleOCR, post-processing is done to keep the output consistent for each column of the scoreboard that uses OCR. The output also has any whitespace stripped from the beginning and the end.

OCR would be used on every section of the scoreboard, except for the agent portrait column, weapon column, and the ping column on each row. For sections that only have numbers, any special characters or letters are removed. One special case is the ultimate status column of the scoreboard. Majority of the time, this column only features numbers and a slash. Once a player's ultimate is ready to use, the scoreboard would show "READY" instead of the numbers. The post-processing would make sure to turn the "READY" into a number to keep things consistent. The number would be dependent on each agent, as each agent has a different number of max ultimate points.

3.4 Development Frameworks

To facilitate a more efficient development, frameworks would be used for both the creation of the backend server and frontend application. While a framework was used in creating the backend server for the previous projects, there was none used for the frontend application. tugamars and deepsidh9 has used FastAPI and Flask respectively for their projects. Both of these are popular web frameworks used to create web servers.

The chosen backend framework for this thesis would be FastAPI. According to the benchmarks created by TechEmpower, FastAPI has a higher performance score compared to Flask in terms of single query and multiple queries [25]. This framework also automatically creates interactive API documentation on the /docs and /redoc URLs. Besides that, FastAPI also creates a schema using the OpenAPI standard for creating APIs. This allows for easy testing of individual modules and testing of the system as a whole.

The chosen frontend framework for this thesis would be NextJS. NextJS is a web framework using the React UI library. By using this framework, it is possible to create multiple pages as well as handle different requests to the backend server in order to mutate any data that is on the server as well as any internal data on the client. This framework also allows faster development as the routes are automatically generated from the file structure during development.

3.5 Testing

Once the product is complete, testing was done to determine the tool's performance compared to broadcasting real-time standards. These broadcasting standards would follow what the ATSC has

established, where the synchronization limit delay should be in the range of 45 ± 15 ms [16]. The other standards mentioned previously would also be considered. Accuracy would determine the correctness of the information being displayed. Execution time would refer to the time it takes for the tool to parse data from the game and display it.

To illustrate, examining Figure 3a would result in the following data, which would be in a JSON format. The spike status would be false. The team on the left would be defending and the team on the right would be attacking with 1 and 0 players respectively. The player has Neon as an agent with 0 kills, deaths, and assists, has their ultimate ready to use, has 99,999 credits, and has the Sheriff as their highest value weapon.

These tests were done in order to evaluate the different parts of the tool in terms of accuracy and execution time. The accuracy tests were done on the models, both OCR and CNN. The tool parses two areas, the scoreboard and top bar, at the same time. The test for execution time was done on these two parts of the tool as well as the system as a whole.

4 RESULTS AND DISCUSSION

4.1 Accuracy

4.1.1 Agent Classification Model. The training data for this model was first loaded into an image dataset. This was done using the utility function "image_dataset_from_directory" method from the keras module which also separates the images into a number of batches. In that dataset, the images were then normalized to keep the values between 0 and 1. The dataset was then split into training, validation, and test datasets by splitting them into a ratio of 7:2:1. This divided the 161 batches into 113 batches, 32 batches, and 16 batches for the training, validation, test datasets. This ratio allows for the majority of the dataset to be used for training and validation. This also allows means that some images are not used for training or validation, which makes the testing more reliable as it is not recognizing images that it already has seen before.

The model was then created using the architecture stated above. The Convolutional Layers for this model had 16, 32, and 16 filters respectively. Each of these layers had a kernel size of 3x3 with a stride of 1 and had ReLu activation applied. The Pooling Layers had a size of 2x2 which essentially results in the image being cut in half after each layer. The Fully-connected Layers had 256 and 20 units respectively, and when combined with the softmax activation allows the model to give each class a confidence level. The model was compiled with the adam optimizer and the categorical crossentropy loss function. Finally, the model was trained for 20 epochs.

Figure 4 shows the loss and accuracy of each epoch of training. The figure also shows that the training and validation lines converge, which means that the resulting model does not suffer from over fitting. Over fitting, in the context of machine learning models, is when the model is good at identifying what it was trained on but not on things that it has not seen.

The model was then evaluated using the test dataset. This was done with the metrics for precision, recall, and binary accuracy. Precision measures how often the model is correct every time it predicts positive for a class. Recall measures how often the model

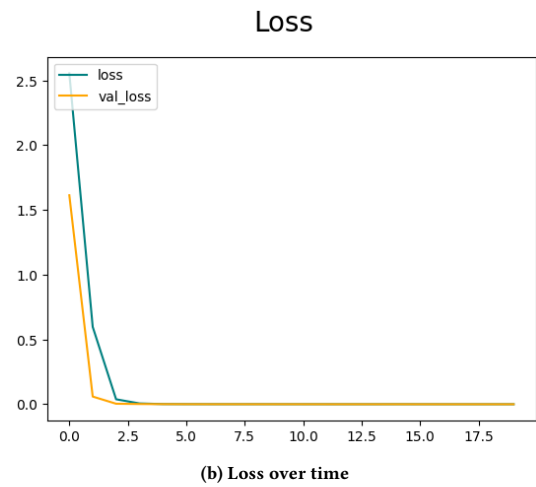
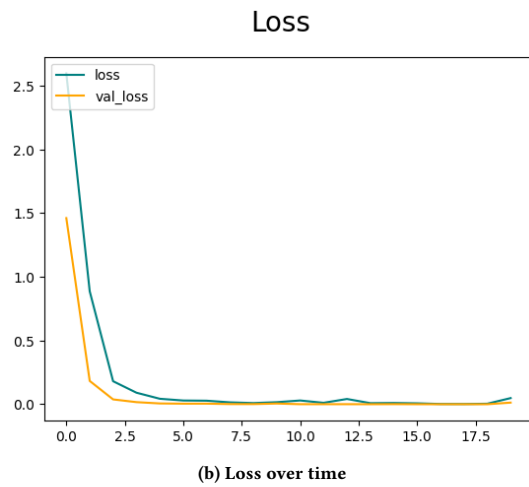
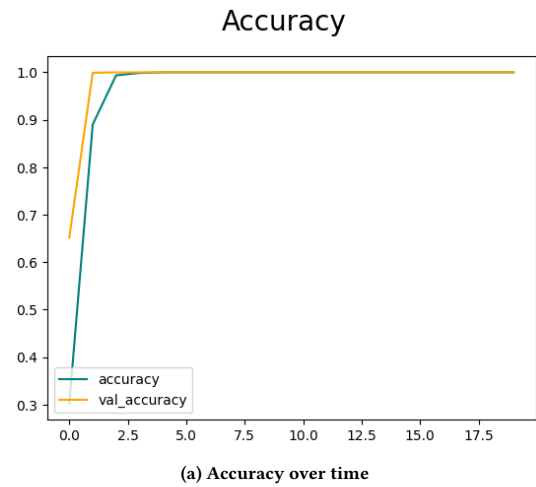
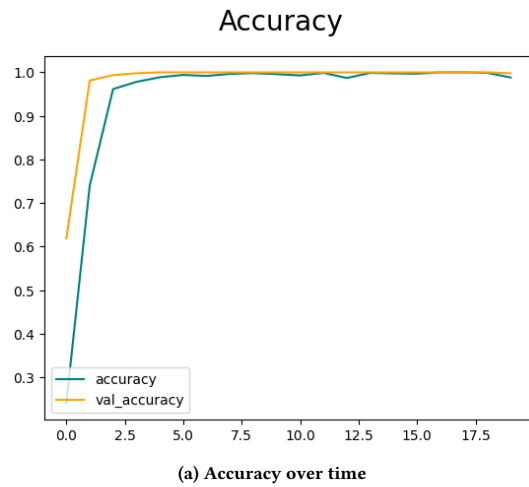


Figure 4: Agent Classification Model Training and Validation Metrics

Figure 5: Weapon Classification Model Training and Validation Metrics

predicts correct every time a class appears. Binary accuracy is a measure on how often the model gets a prediction correct.

Below are the results for the agent classification model. All metrics are seen to be above 99% which means that the model is considered to be highly accurate.

- Precision=0.99553573
- Recall=0.99553573
- Binary Accuracy=0.99955356

4.1.2 Weapon Classification Model. Similar to the previous model, the training images were loaded into an image dataset. The images were then normalized to be between 0 and 1. The dataset was also split into training, validation, and test datasets with the same ratio in the previous model. This divided the 177 batches into 125 batches for training, 35 batches for validation, and 17 batches for testing.

The model was created using the architecture stated previously. The Convolution Layers for this model had 16, 32, and 64 filters

respectively. The kernel size was 3x3 with a stride of 1 and had ReLU activation applied. Pooling Layers are the same. This differed as the Fully-connected Layers had 128 and 18 units respectively. The model was compiled with the adam optimizer and the categorical crossentropy loss function. Finally, the model was trained for 20 epochs.

Figure 5 are the loss and accuracy over each epoch of training. The figure shows, similar to figure 4, the training and validation lines converging. This means that the model also does not suffer from over fitting.

The model was then evaluated using the test dataset. Like previously, the evaluation was done using the precision, recall, and binary accuracy metrics. Below are the results for the weapon classification model. Similar to the previous model, these metrics indicate greater than 99% which means this model can be considered to be highly accurate.

- Precision=1.0

```

1 train1 = data.take(128)
2 val1 = data.skip(128).take(32)
3
4 train2 = data.take(96).skip(32).take(32)
5 val2 = data.skip(96).take(32)
6
7 train3 = data.take(64).skip(32).take(64)
8 val3 = data.skip(64).take(32)
9
10 train4 = data.take(32).skip(32).take(96)
11 val4 = data.skip(32).take(32)
12
13 train5 = data.skip(32).take(128)
14 val5 = data.take(32)
15
16 test = data.skip(train_size + val_size).take(test_size)
    
```

Figure 6: Python Code for Batch Distribution for K-fold Cross Validation

K-fold Iteration	Precision	Recall	Binary Accuracy
1	1.0000	1.0000	1.0000
2	1.0000	1.0000	1.0000
3	0.9871	0.9871	0.9902
4	0.9934	0.9934	0.9976
5	0.9959	0.9959	0.9995

Table 1: Accuracy Results after K-fold Cross Validation

- Recall=1.0
- Binary Accuracy=1.0

These results appeared unconventional which resulted in conducting a k-fold cross-validation to scrutinize the robustness and reliability of the findings. Having a 100% result on these metrics is unrealistic and may suggest the model is unable to perform well in a real-world scenario. This may also suggest that the model had overfitted to the training set. This may not be the case as the graphs in Figure 5 show the training and validation lines don't deviate greatly which indicate the model has not overfitted. The same dataset that was used previously was used to do the cross validation. The dataset had a total of 177 batches with 32 images in each batch. The split of data was the same as before, except the training and validation datasets were used for the cross-validation. Using 5 as k, the model was trained 5 times using the k-fold cross-validation method and the resulting model was tested against the test dataset. The reason for k equal to 5 is it results to each fold equaling 20% of the data which ensures each fold is large enough to capture meaningful patterns in the data. Having k equal to 5 is also a common choice when doing this methodology. The fold equaled 32 batches which resulted in the training, validation, and test datasets to have 128, 32, and 17 batches respectively. Figure 6 shows the Python code used to split the batches into the training, validation, and test datasets. The data variable represents the dataset as a whole. The variables train_size, val_size, and test_size were calculated via the ratio presented previously.

Table 1 shows the results of testing the model after each k-fold iteration. The results imply that the initial results may have occurred by chance. It can still be said that this model is highly accurate as,

Section	Average (s)	Minimum (s)	Maximum (s)
Scoreboard	1.291314	1.252528	24.384231
Top Bar	0.000566	0.000909	0.001002

Table 2: Execution Time Results

on average, the model achieved an accuracy greater than 98% on each iteration.

The initial results may have been caused by what the model is being used for. Valorant weapon icons don't change or have any variability which is the case for icons in general. The model may have learned this and achieved a high accuracy because of this. As the weapons icons in Valorant don't change or are obscured in any way, the fact the model has learned this way should not be a cause for concern and may have a positive influence when the model is predicting on real life scenarios.

Additional tests may be able to further verify the model. Performing K-fold Cross Validation with K equal to different sizes may show patterns that could validate or invalidate the model further. Augmentations to the test dataset could be done to test the robustness of the model.

4.1.3 OCR Accuracy. Getting the accuracy of the OCR was done by using it on 100 different images, specifically cells from the scoreboard, to tailor the results towards the use case of this study. The test itself did not use PaddleOCR by itself. The test would use PaddleOCR along with the post-processing done to the initial output. Each cell was assigned an expected value which was the value of the cell and was compared to what the OCR outputted.

Results show that the accuracy of the OCR was 99%. This means that the result of the text parsing process the tool uses deviates very little actual values found on the scoreboard. The inaccuracy came from a cell that had a "4" which resulted in the initial output treating it as a "b". For future works, it would be possible to gain better accuracy by adjusting the post-processing to recognize this edge case.

4.2 Execution Time

This metric represents how fast the software can update the data to the latest point. There are two sections of the screen that house all the data that can be collected, the scoreboard and the area at the top of the screen which will be called the top bar for this study. Each section was timed separately, since these sections would be executed independently of each other. This means that updates to the scoreboard happen without needing to wait for the top bar to finish its process. To measure the execution time, both processes are run 100 times with the first time excluded to get the average execution time without the cold start. Table 2 shows the average execution time as well as the minimum and maximum execution time for each section.

The difference between execution times may be related to how different the two sections are. The scoreboard is very text heavy and unlike the scoreboard, the Top Bar does not use any optical character recognition as there are no text-based elements in that section. The Top Bar is also static which means it only needs to use

the agent image recognition once while the scoreboard needs to keep using it as positions change very often throughout the game.

Comparing the results to previous real-time standards, the average execution time of one part is outside some standards. The scoreboard average execution time is outside the range defined by ATSC of 45 ± 15 ms [16] or the delay established by Jung et al. [14]. This, however, is within range of the paper discussing VisWiz [3].

5 CONCLUSION

The results show that the system as a whole can be considered to be highly accurate and updates at near real-time speed. The accuracy of the image classification models created and the OCR engine used were above 95% and can be considered to be accurate. The time to execute for the system as a whole has updates under the limits set by ATSC. Individually, the scoreboard module was not able to meet the execution time standards.

This computer vision based system was implemented by using and improving previous projects. The base of this system was a Python web-server which received and sent data that held data from the game. To collect the real-time information from the game, computer vision was used. A CNN was used to recognize different characters and weapons from the game and OCR was used to collect textual data. The data came from the 2 sections, Top Bar and the scoreboard, which updates the Python web-server independently of each other. The created visuals come from the data sent by the web-server.

There are a few points of improvement to consider. It was found during the tests that while the system as a whole updates within the standards set, the module handling the scoreboard fell behind. This can be improved upon by updating the main store as soon as a row in the scoreboard has been parsed. The way it was implemented in this study was to batch process the entire scoreboard via pipelining and then update the store once completed. The update function could be done per row so that updates from the scoreboard happen faster. It may also be possible to have each row be executed independently of the others to update more often and the time between updates would fall under the previously stated standards.

REFERENCES

- [1] AcadArena. [n. d.]. About Us. <https://www.acadarena.com/about-us>. Accessed: 2022-9-28.
- [2] AcadArena. 2022. DLSU vs ADMU - Alliance Games 2022 S1 Finals CONQuest.
- [3] Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samuel White, and Tom Yeh. 2010. VizWiz: Nearly real-time answers to visual questions. In *Proceedings of the 23rd annual ACM symposium on User interface software and technology*. ACM, New York, NY, USA.
- [4] Douglas Crockford. 2008. *JavaScript: The Good Parts*. O'Reilly Media, Sebastopol, CA.
- [5] Deep. 2021. Live-Valorant-Overlay: An Overlay proof-of-concept Application for Valorant.
- [6] escharts. [n. d.]. The International 10. <https://escharts.com/tournaments/dota2/international-10>. Accessed: 2022-9-9.
- [7] escharts. [n. d.]. PGL Major Stockholm 2021. <https://escharts.com/tournaments/csgo/pgl-major-stockholm-2021>. Accessed: 2022-9-9.
- [8] escharts. [n. d.]. Valorant Champions 2021. <https://escharts.com/tournaments/valorant/valorant-champions-2021>. Accessed: 2022-9-9.
- [9] Riot Games. [n. d.]. Riot developer portal. <https://developer.riotgames.com/apis>. Accessed: 2022-6-26.
- [10] Paul Gavrikov. 2020. visuakeras. <https://github.com/paulgavrikov/visuakeras>.
- [11] Shayma A Hmdaoy and Hanaa M Ahmed. 2022. A comparison between the use of CNN and matching templates in recognizing the Iraqi license plate number. *Al-Nahrain Journal of Science* 25, 3 (2022), 43–50.
- [12] ISO 5725-1:1994(en). 1994. *Accuracy (trueness and precision) of measurement methods and results – Part 1: General principles and definitions*. Technical Report.
- [13] Daniel Johnson. 2020. What is TensorFlow? How it Works? Introduction & Architecture. <https://www.guru99.com/what-is-tensorflow.html>. Accessed: 2022-11-30.
- [14] Ilchae Jung, Jeany Son, Mooyeol Baek, and Bohyung Han. 2018. Real-Time MDNet. In *Computer Vision – ECCV 2018*. Springer International Publishing, Cham, 89–104.
- [15] Tero Kerttula. 2020. Early Television Video Game Tournaments as Sports Spectacles. In *Proceedings of the 2019 Esports Research Conference (ESC)*, Jason G Reitman, Craig G Anderson, Mark Deppe, and Constance Steinkuehler (Eds.). ETC Press, Carnegie Mellon University.
- [16] Sara Kudrle, Michel Proulx, Pascal Carrieres, and Marco Lopez. 2011. Fingerprinting for solving A/V synchronization issues within broadcast environments. *SMPTE Motion Imaging J.* 120, 5 (2011), 36–46.
- [17] lexogrine. 2020. Lexogrine HUD Manager.
- [18] Chejui Liao. 2020. OCR engine comparison – tesseract vs. EasyOCR. <https://medium.com/swlh/ocr-engine-comparison-tesseract-vs-easyocr-729be893d3ae>. Accessed: 2022-10-12.
- [19] Haroon Shakirat Oluwatosin. 2014. Client-Server Model. *IOSR journal of computer engineering* 16, 1 (Feb. 2014), 67–71.
- [20] OpenCV. 2016. Opencv-python.
- [21] Keiron O’Shea and Ryan Nash. 2015. An Introduction to Convolutional Neural Networks. (2015).
- [22] PaddlePaddle. 2022. PaddleOCR.
- [23] OBS Project. [n. d.]. Open Broadcaster Software. <https://obsproject.com/>. Accessed: 2022-6-26.
- [24] Sheetal Taneja and Pratibha Gupta. 2014. Python as a Tool for Web Server Application Development. *JIMS 8i-International Journal of Information, Communication and Computing Technology* II, I (2014), 77–83.
- [25] TechEmpower. 2022. Web Framework Benchmarks Round 21. <https://www.techempower.com/benchmarks/>. Accessed: 2022-12-1.
- [26] tugamars. 2020. AcheronObs. <https://github.com/tugamars/AcheronObs>. Accessed: 2022-6-10.
- [27] Valve. 2020. Counter-strike: Global offensive game state integration. https://developer.valvesoftware.com/wiki/Counter-Strike:_Global_Offensive_Game_State_Integration. Accessed: 2022-6-26.

Predicting Student Performance Based on Moodle Forum Interaction Logs

Maya Louise E. Asuero
Department of Computer Science
Xavier University-Ateneo de Cagayan
Cagayan de Oro City, Philippines
masuero@xu.edu.ph

Angelu Ferdinand A. Garcia
Department of Computer Science
Xavier University-Ateneo de Cagayan
Cagayan de Oro City, Philippines
angelu.ferdinand.10@gmail.com

Julius Czar Makiling
Department of Computer Science
Xavier University-Ateneo de Cagayan
Cagayan de Oro City, Philippines
juliusczarmakiling@gmail.com

Jun Rangie C. Obispo
Department of Computer Science
Xavier University-Ateneo de Cagayan
Cagayan de Oro City, Philippines
jobispo@xu.edu.ph

ABSTRACT

This paper predicted students' performance on a Moodle course based on their forum interaction logs and the community of inquiry model. The research aimed to identify key attributes of students' forum interactions, based on the community of inquiry model's indicators of social presence, that can be used to create models that predict a student's course performance. To conduct this study, the researchers used a methodology grounded in data mining which involved data collection, data preprocessing, training, and model evaluation. The researchers used tools and algorithms provided by the machine learning software WEKA in creating the model. Results of the study showed that all indicators of social presence are present among the forum discussion logs, with *continuing a thread* significantly affecting the final grade. However, this may have been influenced by the large volume of messages classified as continuing a thread in comparison to the other indicators. Generally, the indicators of social presence can predict whether the student will merit a final grade that is high grade and low grade to a limited extent.

CCS CONCEPTS

• Theory of computation → Theory and algorithms for application domains → Machine learning theory → Machine learning theory

KEYWORDS

data science; data mining; Moodle logs; forum logs; student performance; computer science

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
PCSC2024, May 2024, Laguna, Philippines
© 2024 Copyright held by the owner/author(s).
<https://doi.org/10.1145/1234567890>

ACM Reference format:

Maya Louise Asuero, Angelu Ferdinand Garcia, Julius Czar Makiling and Jun Rangie Obispo. 2024. Predicting Student Performance Based on Moodle Forum Interaction Logs. In *Proceedings of Philippine Computing Science Congress (PCSC2024)*. Laguna, Philippines, 8 pages.

1 INTRODUCTION

Online learning has seen a dramatic increase in the midst of the global COVID-19 pandemic [19]. Due to stay-at-home orders, educational institutions have shifted from face-to-face course delivery to distance learning. However, online learning is not new. In fact, the market for online learning and educational technologies has been steadily growing for the past 10 years [18, 19]. Among the many technologies and modes used in delivering online courses, learning management systems (LMS) are one of the most used. Massive open online courses and in-person universities have integrated the use of LMS in its courses which allows them to create and organize lessons and other materials used in class such as quizzes [20]. This study is focused on Moodle, an open-source LMS, that aims to provide educational institutions a single robust, secure, and integrated system [1].

Moodle hosts notable local universities such as Ateneo de Davao University, Mindanao State University - Iligan Institute of Technology, University of the Philippines – Baguio, University of the Philippines Open University, and Xavier University – Ateneo de Cagayan. These institutions combine or blend the use of Moodle and the traditional face-to-face classes for their approach to education. However, in response to the COVID 19 pandemic, the universities adapted to full online education, wherein face-to-face classes are substituted with synchronous and recorded video calls, and usual classroom activities are done on Moodle. Forums have also been used by instructors to stimulate discussion and collaboration in asynchronous learning where students and

instructors can initiate discussion threads and interact with one another.

In the context of online education, the Community of Inquiry Model is important to understand the process required for effective learning by considering its three key elements: Teaching Presence, Social Presence, and Cognitive Presence. This model serves as a framework for text-based interactions and has been used in exploring forum logs generated by Learning Management Systems [4].

While numerous studies guided by the Community of Inquiry model have been conducted to investigate the relationship between LMS interaction logs and student performance, studies focusing on forum interaction logs and its relationship to student performance remain limited. This leaves the relationship between forum activities and course performance unclear. A better understanding of this relationship is needed to enhance the learning experience, especially in an online-learning environment.

This study aims to investigate forum interaction logs and its relationship with student performance by using data mining techniques to find patterns from the collected data, create models to predict the students' final grades, and analyze its results. Grounded by the studies of [4] and [8] which underlines the theory of Community of Inquiry and its key element, Social Presence, it is assumed that the interaction between the students manifests Social Presence. With these considerations, this study aims to answer the following research questions:

1. What indicators of social presence are present among the forum discussion logs?
2. Which indicators of social presence features from the forum discussion logs significantly affect the students' final grade?
3. To what extent is social presence in forum discussion logs a good predictor for students' final grade?

2 RELATED LITERATURE

In this section, the researchers discussed topics relevant to ground the study. Aside from Educational Data Mining, and Mining LMS Data for Student Performance, this section focuses on studies that use interaction logs from LMSs in relation to student performance, and the use of Social Presence as a predictor.

2.1 Mining Forum Interaction Logs for Student Performance

There are several notable studies that focus on mining forum interaction logs. A study investigated the course forums of Shanghai Fudan University's sociology program [7]. The forums studied included interactions and the contents of the posts between students and teacher's assistants in relation to the performance of the students in the course. Results of the study showed that students who are more active in forum discussions tend to have higher grades. The researchers assumed that those who got higher grades have better learning attitudes whereas those who received lower marks did not have enough time in the course.

2.2 Social Presence as a Predictor of Student Performance

Social Presence is one of the three key elements of the Community of Inquiry Model. [4] explains that Community of Inquiry serves as a framework for educators in using text-based computer conferencing in delivering a successful higher educational experience to their students. What makes this model important is the fact that research shows that there is a relationship between the three key elements (Teaching Presence, Social Presence, and Cognitive Presence), and the students' perceived learning, satisfaction with the course, satisfaction with the instructor, actual learning, and sense of belonging [3, 5, 13].

Among the studies that associate social presence with students' grades, and perhaps the one that is most similar to the current study is authored by [9]. Their study investigated the use of indicators of social presence as predictors of final grades of a Master's level Computer Science online course. To determine the association between the indicators of social presence and the grades, descriptive statistics and multiple regression analysis were applied. Correlation analysis showed that *continuing a thread*, *asking questions*, *complimenting*, *expressing appreciation* and *vocatives* indicators were strongly positively correlated with the final grade, while the correlation with *self-disclosure* tends to be marginally significant. On the other hand, the results of multiple regression analysis showed that *continuing a thread* is positively associated, and *complimenting*, *expressing appreciation* is negatively correlated with grades.

[22] emphasized the importance of Social Presence in Online Forums among Distance Learners. Their methodology involved manually reviewing and coding forum posts that contained indicators of social presence. Learners found that lack of engagement and participation among themselves, and this made them anxious about presenting their ideas. [14] found that a high degree of social presence affects learners' perceived learning and increases their satisfaction in an online learning experience. Therefore, through a good sense of social presence it could bring the learners to a greater emotional satisfaction.

A separate study [23] adapted a similar methodology as the one previously mentioned. This focused on exploring the relationship between learners' social presence in MOOC forums and learners' prestige with the use of automated content analysis and social network analysis. The study adopted the use of a revised framework for social presence [16] from the first proposed framework [4]. From 4,650 posts with an equivalent of 23,755 sentences, the researchers randomly chose 3,500 to manually code using the categories of social presence. Three researchers participated in the coding process with each coder being randomly assigned 1,200 sentences to code. To ensure correctness in coding, the coders constantly compared codes until they reach an agreement rate of 100%. This was then used to train and validate the text classification model. Results of the study showed that certain indicators of social presence have positive correlations with learners' prestige such as *asking questions*, *expressing gratitude*,

self-disclosure, sharing resources and using vocative. On the other hand, *disagreement/doubts/criticism and negative emotions* had a negative correlation with learners' prestige.

Although there have been numerous studies that attempt to relate student performance to general interaction logs, only a few investigations made use of forum interaction logs against student performance. Even so, none of these studies have given a special focus to classifier algorithms to investigate student performance.

3 METHODOLOGY

The research design used by this study leans towards the direction of data mining (Figure 1). Thus, the following steps were adapted: data collection, preprocessing, feature engineering, training, and evaluation [15].

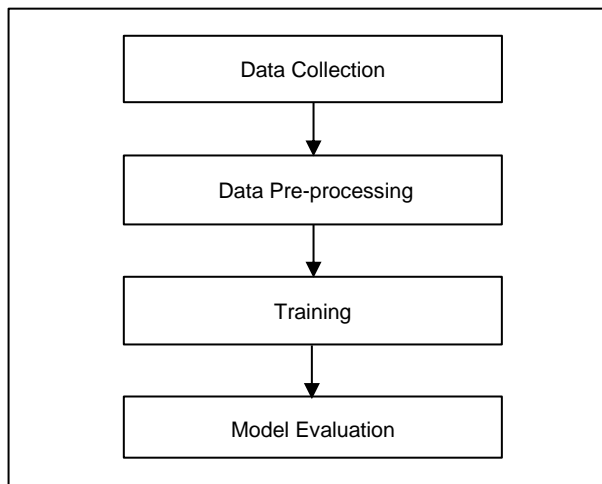


Figure 1. Research Design

To provide a brief overview, firstly, data will be collected from the Moodle LMS. Then, data is prepared through preprocessing. The preprocessing phase starts with the data cleaning, manual classification of forum messages, and feature engineering. After that, the machine learning algorithm will use the data to produce models. Finally, the performance of the resulting models is evaluated through statistical methods.

3.1 Data Collection

The datasets were taken from a university learning management system implemented on Moodle. In terms of the selection of courses, a criteria was applied similar to a study conducted on Coursera MOOC [21] and as suggested by the study of [11]. This was done to ensure that there are enough data points from a course to conduct this study. The criteria were decided based on the indicators of social presence since these courses have an increased level of meaningful interactions between students [9]. Specifically, the selected courses must show the presence of the indicators of social through the forum discussion logs and have variations of the said indicators.

The dataset includes data from six courses of the school year 2020 - 2021. To avoid overfitting, the study selected courses with a distinct composition from the other selected course based on year level and program. Each course spans a quarter of the academic year which is approximately two months. Students in the dataset were anonymized. The datasets include the students' grades, which reflect the breakdown of scores in activities and tests as well as the final grade in that course, forum discussion logs which show the contents and data submitted on discussion forums, and general interaction logs of the students in the course.

Moodle has five forum types. This study used forum discussion logs regardless of the type of forum.

3.2 Preprocessing

Before the data can be processed, it must be prepared to match the requirements of the mining software. This includes converting the data into the desired format. The datasets were presented in an MS Excel spreadsheet and were converted into a file format recognized by WEKA. Specifically, the datasets were converted into single-sheet xlsx format and uploaded to the mining software using extension packages found in WEKA.

3.2.1 Manual Classification

To distinguish the quality of the posts, the messages in discussion forums were classified into categories grounded in the open communication indicators of social presence [8]. Specifically, this study makes use of six categories that correspond to the indicators. Messages were coded or classified following the definitions of each category. It is worth noting that the posts or messages were manually classified by the researchers, similar to the studies of [6, 9]. Each message was classified as a single unit, therefore the researchers took into consideration the general idea and intent of the message based on its content.

Posts that fall into multiple categories were classified according to the value of the overall forum message. These will be ranked in the following order of precedence:

1. Quoting from others' messages,
2. Referring explicitly to others' messages,
3. Asking questions,
4. Expressing agreement.
5. Complimenting, expressing appreciation,
6. Continuing a thread

Along with the order of precedence, to further minimize the subjectivity of the classification, each entry was cross validated among the researchers; the messages were carefully reread, and their classification was made sure to be correct at least three times. To do this, the forum discussion logs which contained the messages were divided into three sets of messages. Each set was evaluated and classified by three different researchers.

Once all the forum discussion logs were classified, the researchers then checked the classification. If all evaluators gave the same classification for a message, then the message is classified

according to the said classification. However, if there are differences or conflicts in classification, the evaluators will have to discuss the said message and come to a unanimous conclusion as to its final classification.

3.2.2 Feature Engineering

An important step of data preprocessing is the identification of variables from the datasets which will be mined. The variables, in this case, refer to spreadsheet columns. Therefore, the columns that contain data that are not relevant to the study were removed. From the data set, the columns that are relevant to the study are *userid* to uniquely identify the student, *message* which shows the content of the forum post, *parent* which distinguishes replies from discussion starters, *wordcount* of the message, and course total in percentage.

Table 1. Features

Features	Description
total_OcT	Total number of forum posts classified as <i>Continuing a thread</i>
total_wordcount_OcT	Total number of words used in total_OcT
total_OQ	Total number of posts classified as <i>quoting from others' messages</i>
total_wordcount_OQ	Total number of words used in total_OQ
total_OR	Total number of posts classified as <i>referring explicitly to others' messages</i>
total_wordcount_OR	Total number of words used in total_OR
total_OA	Total number of forum post classified as <i>asking questions</i>
total_wordcount_OA	Total number of words used in total_OA
total_OCa	Total number of forum posts classified as <i>complimenting, expressing appreciation</i>
total_wordcount_OCa	Total number of words used in total_OCa
total_OE	Total number of forum posts classified as <i>expressing agreement</i>
total_wordcount_OE	Total number of replies received, including indirect replies
replies_received	Total number of words used in replies_received
replies_wordcount	Total number of characters used in replies_received
replies_charcount	Total number of times a thread was clicked and opened
Disc_viewed	Total number of times a course module (which may contain discussions/threads) was clicked and opened
Module_viewed	Total number of forum posts classified as <i>Continuing a thread</i>

3.2.3 Analysis

Analysis involves making use of the built-in tools of WEKA that can be specifically found in its the Experiment Environment. This automates the creation of statistical tests and the running of multiple learning schemes on the data sets. For instance, the test facility allows statistical significance test of different learning schemes.

Further, Spearman's Rank Correlation Coefficient and Point-Biserial Correlation were used to understand the relationship between each of the different features found in Table 1 and the final grade. A correlational matrix was also used to show the relationship between the features. Histograms were used to show the distribution of the data between different features.

3.3 Training

The study used the mining tool WEKA to process the datasets. This tool was developed by the University of Waikato in New Zealand. It is a software that has a collection of machine learning algorithms and visualization tools that can be used for data mining. WEKA's built-in Experiment Environment was used to run and train the prediction models.

The data was processed using Naive Bayes and Logistic Regression classification algorithms. Classification algorithms are used to model data and its relationship to a certain result. The Naive Bayes classifier assumes that features are independent given a class, and was found to be widely used and effective in practice [14]. After evaluation of the results, Logistic Regression was used for the High-Low Grade classification. The algorithm Logistic Regression can be used by predicting binary outcomes such as the high-low grade classification.

3.4 Model Evaluation

To test the accuracy of the models, k-fold cross-validation was used. For this study, 10-folds cross-validation was used where 90% of the data points are trained and 10% are tested with over 10 different runs.

The results of the experiment made on WEKA are taken directly from the data mining software. To identify the attributes which have the greatest effect on final marks, the researchers used feature-selection algorithms. Finally, to rank the attributes or features, the frequency of selection for each attribute as decided by each algorithm is recorded. Completion of this step will give a list of features sorted according to how much influence they have on the final course percentage.

4 RESULTS AND FINDINGS

Initially, data from 14 classes were released to the researchers. These were then filtered to reduce overfitting. An overfit could occur in this context when the same student is found in at least two sections or blocks. From this point onwards, a block (referred to as a class in normal circumstances) should be understood as a group

of students studying the same subject. Furthermore, two blocks may or may not study the same subject.

Table 2. Summary of Forum Interaction Logs from 11 Blocks

Summary	Value
Number of students	291
Total Messages	1513
Threads Started by Students	1304
Threads Started by Teachers	37

This filtering process reduced the number of blocks to 11, with an average of 26 students. Table 2 shows a general summary of the forum interaction logs from the final set of blocks. With 291 students, most of the messages are thread-starters created by the students, while only an ample number are created by teachers.

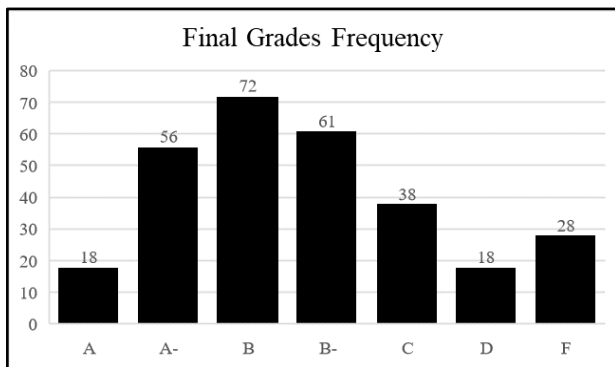


Figure 2. Histogram of Letter Grades

The quantity of the final grades is shown in Figure 2. As observed, most of the students have a letter grade of either A-, B, or B-. This creates an imbalance between the other letter grades which could mean that the algorithm will have fewer data points to learn from.

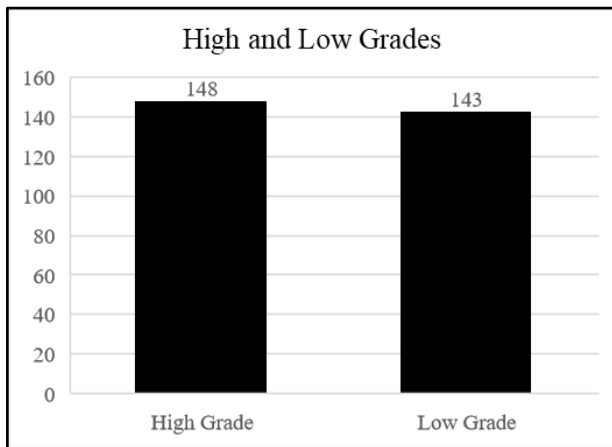


Figure 3. Histogram of High-Low Grades

To mitigate the imbalance, the researchers have decided to classify the students as having low grades or low performance and high grades or high performance. The median was used to determine

how the data would be classified as high grade or low grade. With a median of 76.18, final grades of A, A-, and B were classified as high grade while final letter grades of B-, C, D, and F were classified as low grade. Figure 3 shows the histogram of the high and low grades classification.

In the previous sections, the researchers posed three research questions for this study. The succeeding sections will discuss the findings of the study and answers to these questions.

4.1 What indicators of social presence are present among the forum discussion logs?

Manual classification showed all the indicators of social presence can be found among the forum discussion logs. Although complete, the frequency for each indicator is imbalanced. Table 3 lists each indicator according to its frequency as classified in the messages.

Table 3. Message Classification

Indicators of Open Communication	Count
Continuing a thread	1132
Asking a question	241
Expressing agreement	75
Complimenting, expressing appreciation	48
Referring explicitly to others' messages	14
Quoting from others' messages	3

This result is consistent with other studies, wherein, likewise in online forums, all the types of indicators of social presence under open communication can be found [9, 22].

The social presence category of open communication allows for reciprocal and respectful exchanges, especially the mutual awareness and recognition of others' contributions [4]. Since tools that are integrated into these systems are designed to overcome the limitations of distance learning, social presence prevails naturally because it promotes and encourages such interaction [10].

The teacher's module design in the LMS has a major influence on the students' activity in the forums. For example, messages classified as *continuing a thread* are mostly responses to graded activities. Thus, the ranking of each indicator gives a good insight into how the teacher utilizes the LMS.

However, it is best to take the interpretation with caution. Following the order of precedence during the manual classification of messages means that when it fails to be captured by the other indicators, the default classification will always be *continuing a thread*. With this thought, a convincing argument as to why there is an overwhelming number of messages classified as *continuing a thread* is because the latter can be broken down into finer subcategories of indicators, or simply because the correct classification does not exist in the set of indicators of social presence.

4.2 Which indicators of social presence features from the forum discussion logs significantly affect the students' final grades?

Although the study explored two ways of classifying student performance through letter grade classification and high-low grade classification, *continuing a thread* as an interaction and its word count were present in the feature selection results for both types of classification. This may have been influenced by the large volume of messages classified under this indicator. Moreover, the results of the analysis showed that *continuing a thread*, as an interaction and its word count, also has the highest correlation and a strong relationship with the class feature. However, these two features were determined as highly correlated which prompted the researchers to drop the feature for continuing a thread word count from the feature selection.

Aside from continuing a thread, *referring to others' messages* was also selected from the feature selection process as having a significant contribution to the model. However, it was only considered for the high-low grade classification and not the letter grade classification. This non-inclusion can be accredited to the limited range and variation of the data in this feature which can make it a challenge to distinguish for the 7-letter grade classification. Close inspection of the dataset shows that students with *referring to others' messages* were classified as having A, A-, or B letter grade or high grade for high-low classification.

Notably, the word count for the indicator *asking questions* was included in the results of the feature selection. However, the total interaction of this indicator was not selected. This could indicate that, for instance, the actual number of questions asked does not matter, but rather the content of the question being asked. However, without a method for scoring the content of the message, it cannot be generalized that longer messages have more substantial content; only that longer messages are predictive of high performance.

Similar studies have shown similar results for the indicators [5, 9, 11, 22] and the correlation of word counts [12]. Other studies have reported on other indicators, such as *asking questions*, *quoting others' messages*, and *complimenting others*, have better relationships with student performance [17]. The results of this dataset mainly highlight *continuing a thread* and *asking questions* word count as the indicator that can significantly affect the students' performance. Moreover, replies word count, and discussions viewed are also predictive of student performance. *Referring to others' messages* can be considered predictive of the student performance; however, its significance diminishes as the number of classes increases.

4.3 To what extent is social presence in forum discussion logs a good predictor for students' final grades?

Given the model evaluation results, the social presence of a student can be used as a predictor for students' final grades to a limited extent. Between the two methods of grade classification, the model

has a higher performance rate when predicting high grades vs low grades in comparison to the university marking system classification. This is proven by observing the increased model precision, recall, and accuracy when classification decisions are limited to only high and low seen in Table 4.

Looking into the histogram of student final grades, a convincing explanation is that there exists an imbalance in the representation. This result is similar to [2]'s results where the model could not accurately predict 5 to 10-grade categories, but could accurately predict failing students from other classifications.

Table 4. Summary of Evaluation Results

Metric	Letter Grades (Naive-Bayes)	High-Low (Naive-Bayes)	High-Low (Logistic Regression)
Accuracy	29.21%	69.76%	69.42%
Kappa Statistic	.1733	.3988	.3899
Precision	.332	.731	.701
Recall	.292	.698	.694

The resulting models of the 2-class datasets using both Naïve Bayes and logistic regression revealed that both models perform very similarly. This can be proven by observing the close metrics for its accuracy, kappa statistic, precision, and recall. Considering that both models perform unsatisfactorily at accurately predicting student performance, it further strengthens the previously mentioned remarks regarding the lack of variety in student forum post types to correctly classify high-low grades.

In summary, the resulting models perform unsatisfactorily in predicting students' performance using forum interactions following the indicators of social presence, particularly for open communication. For the courses included in the dataset, participation in forum discussion counts toward the final grades. However, this comprises only a small portion of the final grade, so other factors might come into play. This includes how well the student performed in other activities, quizzes, and exams. In addition, the data imbalance for the letter grade classification limits the performance of the model as some classes only have limited data in comparison to others. From the results gathered, it may be sufficient to say that the social presence in forum discussion logs contributes as a predictor for final grade only to a limited extent of accuracy, but more data is needed for it to be counted as a reliable result.

5 CONCLUSION

Based on the results, data analysis, and model evaluation, the researchers reached three conclusions.

First, all indicators of social presence for the open communication category are present among the forum discussion logs. Manual classification of the messages showed that all the indicators of social presence can be found among the forum discussion logs.

Subsequently, *continuing a thread* significantly affects the students' final grade. Analysis on both the 7-class and 2-class features consistently showed that *continuing a thread* has a strong positive correlation with the students' final grade. A possible explanation for this is that majority of the messages are classified to this indicator. It could be that messages that fall into this category are mostly responses to questions posted by instructors and are therefore graded.

Finally, indicators of social presence can predict whether the student will merit a final grade that is above average or below average to a limited extent. Predicting the university-standard letter grades resulted in an upsetting prediction performance. However, predicting whether the student will merit a grade that is high (above the median) or low (below the median) resulted in a much better performance.

In furthering the study conducted, the researchers recommend four avenues to explore:

5.1 Wider Scope

To provide a better model with increased performance, a larger dataset is essential to provide a more reliable result. Due to the manual classification nature of preprocessing the forum post type, the researchers were not able to work on a much larger scale to work around the time constraint. The researchers recommend exploring other programs and colleges to see if similar results can be obtained.

The profile of the data used in this study has shown the large number of messages classified as *continuing a thread* which may have led to its correlation with student performance. The addition of more data may be able to balance the representation of different indicators with a more generalized audience. In turn, working with a bigger dataset can include more instances that may balance the distribution letter grades and may improve the model.

5.2 Enhanced Preprocessing

Machine Learning offers a wide variety of tools and techniques. To come up with different results, the researchers recommend exploring other methods particularly in the preprocessing stage such as incorporating natural language processing (NLP). The results of the study focused more on the statistical metrics of the forum messages. Use of NLP may be used to explore the semantics of forum messages and capture patterns not presented in this study.

Future iterations of this study may also consider other factors or attributes (assignments, exams, time between posts, etc.) to a certain extent with a focus on forum logs, and other categorization methods.

5.3 Exploring Methods

Using different algorithms may affect the results of the study. The researchers recommend exploring other algorithms, especially more advanced algorithms, to see if the results are better. This may

include not only classification but regression and clustering models as well. A combination of different algorithms is also worth exploring. Specifically, the researchers recommend the exploration of random trees and support vector machines (SVM) classifiers since both are also widely used in related studies.

5.4 Include Other Categories of Social Presence

This study limited the use of the indicators of social presence to the indicators under the open communication category. Use of the indicators under the categories interpersonal communication and cohesive communication may reveal more insight into the context of the messages posted by students. The researchers recommend including both categories in future studies to gain more understanding of the dataset.

ACKNOWLEDGMENTS

This research was supported by the Xavier University – Ateneo de Cagayan College of Computer Studies and the Kinaadman University Research Office. With the guidance of the research adviser, Mr. Jun Rangie Obispo, along with the research panelists, Engr. Gerardo S. Doroja and Ms. Shayryl Mae R. Sabal, and the special help of EDM and learning analytics expert, Dr. May Talandron-Felipe, this would not have been possible.

REFERENCES

- [1] About Moodle - MoodleDocs: https://docs.moodle.org/311/en/About_Moodle. Accessed: 2021-07-19.
- [2] Ademi, N., Loshkovska, S. and Kalajdziski, S. 2019. Prediction of Student Success Through Analysis of Moodle Logs: Case Study. *ICT Innovations 2019: Big Data Processing and Mining*. Springer Nature Switzerland AG. 27–40.
- [3] Akyol, Z. and Garrison, D.R. 2008. The Development of a Community of Inquiry over Time in an Online Course: Understanding the Progression and Integration of Social, Cognitive and Teaching Presence. *Journal of Asynchronous Learning Networks*. 12, (Dec. 2008), 3–22.
- [4] Anderson, T., Garrison, D.R. and Archer, W. 1999. Critical Inquiry in a Text-Based Environment: Computer Conferencing in Higher Education. *The Internet and Higher Education*. 2, 2 (Mar. 1999), 87–105. DOI:[https://doi.org/10.1016/S1096-7516\(00\)00016-6](https://doi.org/10.1016/S1096-7516(00)00016-6).
- [5] Arbaugh, J.B. 2008. Does the Community of Inquiry Framework Predict Outcomes in Online MBA Courses? *International Review of Research in Open and Distributed Learning*. 9, 2 (2008), 1–21. DOI:<https://doi.org/10.19173/irrodl.v9i2.490>.
- [6] Bhatia, S., Biyani, P. and Mitra, P. 2012. Classifying User Messages For Managing Web Forum Data. *WebDB* (2012).
- [7] Feng, Y., Chen, D., Zhao, Z., Chen, H. and Xi, P. 2015. The Impact of Students And TAs' Participation on Students' Academic Performance in MOOC. *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015* (New York, NY, USA, Aug. 2015), 1149–1154.
- [8] Garrison, D.R. 2011. *E-Learning in the 21st Century: A Framework for Research and Practice*. Routledge.
- [9] Joksimović, S., Gašević, D., Kovanović, V., Riecke, B.E. and Hatala, M. 2015. Social presence in online discussions as a process predictor

- of academic performance. *Journal of Computer Assisted Learning*. 31, 6 (2015), 638–654. DOI:<https://doi.org/10.1111/jcal.12107>.
- [10] Kear, K. Social presence in online learning communities. 9.
- [11] Liu, S.Y., Gomez, J. and Yen, C.-J. 2009. Community College Online Course Retention and Final Grade: Predictability of Social Presence. (2009), 19.
- [12] López, M.I., Luna, J.M., Romero, C. and Ventura, S. 2012. Classification via clustering for predicting final marks based on student participation in forums. (Jan. 2012), 4.
- [13] Richardson, J. and Lowenthal, P. 2017. Instructor Social Presence: Learners' Needs and a Neglected Component of the Community of Inquiry Framework. *Social Presence in Online Learning: Multiple Perspectives on Practice and Research*. (Jan. 2017).
- [14] Richardson, J. and Swan, K. 2003. An Examination of Social Presence in Online Courses in Relation to Students' Perceived Learning and Satisfaction. *Journal of Asynchronous Learning Network*. 7, 1 (Feb. 2003).
- [15] Romero, C. and Ventura, S. 2010. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*. 40, 6 (Nov. 2010), 601–618. DOI:<https://doi.org/10.1109/TSMCC.2010.2053532>.
- [16] Shea, P., Hayes, S., Vickers, J., Gozza-Cohen, M., Uzuner, S., Mehta, R., Valchova, A. and Rangan, P. 2010. A re-examination of the community of inquiry framework: Social network and content analysis. *The Internet and Higher Education*. 13, 1 (Jan. 2010), 10–21. DOI:<https://doi.org/10.1016/j.iheduc.2009.11.002>.
- [17] Taghizadeh, M. and Vaezi, S. 2011. Exploring Social Presence in Virtual Learning Environment. *World Applied Sciences Journal*. 15, (Jan. 2011), 120–128.
- [18] The Evolution of Distance Education in 2020: 2020. <https://educationonline.ku.edu/community/distance-education-evolution-in-2020>. Accessed: 2021-06-24.
- [19] The Rise Of Online Learning: 2020. <https://www.forbes.com/sites/ilkerkoksal/2020/05/02/the-rise-of-online-learning/>. Accessed: 2021-06-24.
- [20] What is a Learning Management System: 2017. <https://www.cmswire.com/digital-workplace/what-is-a-learning-management-system/>. Accessed: 2021-06-24.
- [21] Xing, W., Tang, H. and Pei, B. 2019. Beyond positive and negative emotions: Looking into the role of achievement emotions in discussion forums of MOOCs. *The Internet and Higher Education*. 43, (Oct. 2019), 100690. DOI:<https://doi.org/10.1016/j.iheduc.2019.100690>.
- [22] Zaini, S.H. and Ayub, A.F.M. Exploring Social Presence in Online Forums among Distance Learners. 16.
- [23] Zou, W., Hu, X., Pan, Z., Li, C., Cai, Y. and Liu, M. 2021. Exploring the relationship between social presence and learners' prestige in MOOC discussion forums using automated content analysis and social network analysis. *Computers in Human Behavior*. 115, (Feb. 2021), 106582. DOI:<https://doi.org/10.1016/j.chb.2020.106582>.

Factoring a Quadratic from a Degree N Polynomial Using Newton's Method

Pablo R. Manalastas
Ateneo de Manila University
Quezon City, Philippines
pmanalastas@gmail.com

ABSTRACT

Given a polynomial $P(x)$ of degree n , namely the polynomial $P(x) = c_0x^n + c_1x^{n-1} + c_2x^{n-2} + \dots + c_{n-1}x + c_n$, where the coefficients $c_j, j = 0, 1, \dots, n$ are real, we want to numerically find a quadratic factor $K(x) = x^2 + ax + b$ of this polynomial. In order to find zeroes of $P(x)$, we just find a quadratic factor $K(x)$, and find the zeroes of $K(x)$ using the quadratic formula. Repeatedly factoring out a quadratic gives all the zeroes of the polynomial $P(x)$. Starting with initial guesses for the coefficients a and b , say the initial guesses a_0 and b_0 , we use long division of $P(x)$ by $K_0(x) = x^2 + a_0x + b_0$, to obtain a quotient $Q(x) = q_0x^{n-2} + q_1x^{n-3} + \dots + q_{n-2}$ and remainder $R(x) = r_0x + r_1$. If both $r_0 = 0$ and $r_1 = 0$, then $K_0(x)$ is a quadratic factor of $P(x)$, and we are done. Otherwise we look for better choices for a and b , using Newton's method. The new ideas presented in this paper are (1) the two-loop algorithm in Section 5 for long division of a degree n polynomial $P(x)$ by a monic quadratic $K(x)$, and (2) the recursive algorithm in Section 6 for computing the partial derivatives $\partial r_0/\partial a, \partial r_0/\partial b, \partial r_1/\partial a$ and $\partial r_1/\partial b$.

KEYWORDS

Polynomial, quadratic, Newton's method

1 LONG DIVISION OF $P(x)$ BY $K(x)$

Long division of the polynomial $P(x) = c_0x^n + c_1x^{n-1} + c_2x^{n-2} + \dots + c_{n-1}x + c_n$ by the quadratic $K(x) = x^2 + ax + b$ can be carried out using the following long division scheme, shown in Section 2. In this scheme, the quotient obtained is $Q(x) = q_0x^{n-2} + q_1x^{n-3} + \dots + q_{n-2}$ and the remainder is $R(x) = r_0x + r_1$, where $r_0 = q_{n-1} = c_{n-1} - bq_{n-3} - aq_{n-2}$ and $r_1 = q_n = c_n - bq_{n-2}$. The first row in this scheme is the divisor $K(x) = x^2 + ax + b$. The second row in this scheme are the column headers, which are the powers of x in decreasing order from x^n to 1. The third row are the coefficients $q_0, q_1, q_2, \dots, q_{n-2}$ of the quotient $Q(x)$, the coefficients only without the powers of x , which are in the column headers. The fourth row are the coefficients of the dividend polynomial $P(x)$, namely $c_0, c_1, c_2, \dots, c_n$.

We obtain the fifth row by multiplying each of the terms of $K(x) = x^2 + ax + b$ by $q_0 = c_0$, to get the results q_0, aq_0 , and bq_0 , which we put on the fifth row under the columns for x^n, x^{n-1} , and x^{n-2} . Fourth row minus fifth row gives the results 0, $q_1 = c_1 - aq_0$, and $c_2 - bq_0$, which we put in the sixth row. Then we multiply $K(x) = x^2 + ax + b$ by q_1 to get the results q_1, aq_1 , and bq_1 which we put in the seventh row, under the columns for x^{n-1}, x^{n-2} , and x^{n-3} . Then we do sixth row minus seventh row and put the results in the eighth row, and so on. This is just the traditional long division of $P(x)$ by $K(x)$.

The coefficients of the quotient polynomial obtained by this long division are, in sequence,

$$\begin{aligned} q_0 &= c_0, \\ q_1 &= c_1 - aq_0, \\ q_2 &= c_2 - bq_0 - aq_1, \\ &\dots, \end{aligned} \tag{1}$$

$$q_{n-2} = c_{n-2} - bq_{n-4} - aq_{n-3}$$

and the remainder is $R(x) = r_0x + r_1$, where

$$\begin{aligned} r_0 &= q_{n-1} = c_{n-1} - bq_{n-3} - aq_{n-2}, \\ r_1 &= q_n = c_n - bq_{n-2}. \end{aligned} \tag{2}$$

2 LONG DIVISION SCHEMA

Long division of the degree n polynomial $P(x) = c_0x^n + c_1x^{n-1} + c_2x^{n-2} + \dots + c_{n-1}x + c_n$ by the quadratic factor $K(x) = x^2 + ax + b$ is shown in Figure 1.

3 NEWTON'S METHOD OF COMPUTING $K(x)$

Starting with initial guesses for the coefficients a and b of $K(x)$, say the initial guesses a_0 and b_0 , we use long division of $P(x)$ by $K_0(x) = x^2 + a_0x + b_0$, to obtain a quotient $Q(x) = q_0x^{n-2} + q_1x^{n-3} + \dots + q_{n-2}$ and remainder $R(x) = r_0x + r_1$. If both $r_0 = 0$ and $r_1 = 0$, then $K_0(x)$ is a quadratic factor of $P(x)$, and we are done. Otherwise we look for better choices for a and b , using the quadratically converging Newton's method. Note that both r_0 and r_1 are functions of a_0 and b_0 , namely that $r_0 = f(a_0, b_0)$ and $r_1 = g(a_0, b_0)$, where $f()$ and $g()$ are well-defined by the long division of $P(x)$ by $K_0(x)$ in Section 1 and Section 2. Thus better guesses a_1 and b_1 can be obtained by solving the system of two nonlinear equations $f(a_0, b_0) = 0$ and $g(a_0, b_0) = 0$, by using the Newton's method.

For notational convenience, when treating as variables we use a and b , but when talking about specific values of a and b , we use the subscripted versions a_0, b_0, a_1 , and b_1 . We want the remainder of long division to be zero. That is, we want to solve the system of two nonlinear equations, namely

$$\begin{aligned} r_0 &= f(a, b) = 0, \\ r_1 &= g(a, b) = 0. \end{aligned}$$

Expand both functions $f()$ and $g()$ by Taylor series around the initial guess (a_0, b_0) to get the infinite series expansions, namely

$$\begin{aligned} f(a, b) &= f(a_0, b_0) + f_a(a_0, b_0)(a - a_0) + f_b(a_0, b_0)(b - b_0) + \dots \\ g(a, b) &= g(a_0, b_0) + g_a(a_0, b_0)(a - a_0) + g_b(a_0, b_0)(b - b_0) + \dots \end{aligned}$$

Here $f_a() = \partial f()/\partial a$, $f_b = \partial f()/\partial b$, $g_a() = \partial g()/\partial a$, and $g_b() = \partial g()/\partial b$. Using only the first three terms of each series as shown,

$x^2 + ax + b$	x^{n-1}	x^{n-2}	x^{n-3}	x^{n-4}	\dots	x	1
c_0	c_1	q_0	q_1	q_2	\dots	q_{n-3}	q_{n-2}
$q_0 = c_0$	aq_0	bq_0	c_3	c_4	\dots	c_{n-1}	c_n
0	$q_1 = c_1 - aq_0$	$c_2 - bq_0$	aq_1	bq_1			
	0	$q_2 = c_2 - bq_0 - aq_1$	$c_3 - bq_1$	aq_2	bq_2		
		0	$q_3 = c_3 - bq_1 - aq_2$	$c_4 - bq_2$	aq_3	\dots	
			0	$q_4 = c_4 - bq_2 - aq_3$	\dots	\dots	
					\dots	$q_{n-1} = c_{n-1} - bq_{n-3} - aq_{n-2}$	$q_n = c_n - bq_{n-2}$

Figure 1: Long division of the degree n polynomial $P(x) = c_0x^n + c_1x^{n-1} + c_2x^{n-2} + \dots + c_{n-1}x + c_n$ by the quadratic factor $K(x) = x^2 + ax + b$

by dropping the second and higher degree terms in $(a - a_0)$ and $(b - b_0)$, and equating them to zero, we get

$$\begin{bmatrix} f(a_0, b_0) \\ g(a_0, b_0) \end{bmatrix} + \begin{bmatrix} f_a(a_0, b_0) & f_b(a_0, b_0) \\ g_a(a_0, b_0) & g_b(a_0, b_0) \end{bmatrix} \cdot \begin{bmatrix} a - a_0 \\ b - b_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

By transposing the first matrix to the right hand side of the equation, then multiplying both sides of the equation by the inverse of the square matrix, we get the resulting equation

$$\begin{bmatrix} a - a_0 \\ b - b_0 \end{bmatrix} = - \begin{bmatrix} f_a(a_0, b_0) & f_b(a_0, b_0) \\ g_a(a_0, b_0) & g_b(a_0, b_0) \end{bmatrix}^{-1} \cdot \begin{bmatrix} f(a_0, b_0) \\ g(a_0, b_0) \end{bmatrix}$$

Carrying out the indicated matrix operations, and simplifying, we get our desired solution for (a, b) , namely

$$\begin{aligned} a &= a_0 - \frac{f(a_0, b_0)g_b(a_0, b_0) - g(a_0, b_0)f_b(a_0, b_0)}{D} \\ b &= b_0 - \frac{g(a_0, b_0)f_a(a_0, b_0) - f(a_0, b_0)g_a(a_0, b_0)}{D} \quad (3) \\ D &= f_a(a_0, b_0)g_b(a_0, b_0) - g_a(a_0, b_0)f_b(a_0, b_0) \end{aligned}$$

If the determinant D is nonzero, we are able to compute solutions for new values of a and b , using the above equations.

4 NEWTON'S METHOD

We summarize the Newton's method for computing a monic quadratic factor $K(x)$ of polynomial $P(x)$ as follows.

- (1) We start with some initial guesses for the values of a and b , say our initial guesses are $a_0 = 1$ and $b_0 = 2$.
- (2) Then we do long division of polynomial $P(x)$ by the quadratic $K_0(x) = x^2 + a_0x + b_0$, to get the quotient coefficients $q_0, q_1, q_2, \dots, q_{n-2}$, and the remainder coefficients r_0 and r_1 .
- (3) If $r_0 = 0$ and $r_1 = 0$, then $K_0(x)$ is a quadratic factor of $P(x)$ and we stop computation.
- (4) Otherwise, if either r_0 or r_1 is nonzero, we compute "better" values of a and b , using the Newton's formulas in Equation 3 in Section 3 above to obtain new values a_1 and b_1 . Then we replace a_0 and b_0 by a_1 and b_1 , and repeat the computation from Step 2.

The Newton's method is known to converge to the actual a and b values quadratically. That is, if the error of the current guess

is $\epsilon = \max\{|a - a_0|, |b - b_0|\}$, then the error of the next guess is approximately ϵ^2 . We shall observe this quadratic convergence in the examples that will be given in Section 9.

5 COMPUTER CODE FOR DIVISION OF $P(x)$

BY $K_0(x)$

The following is a C program fragment for doing long division of the degree n polynomial $P(x)$ by the monic quadratic $K_0(x)$, that is explained in Section 1.

The inputs to the program are: (1) the coefficients $c_0, c_1, c_2, \dots, c_n$ of the dividend polynomial $P(x)$, and (2) the coefficients a_0 and b_0 of the divisor polynomial $K_0(x)$. (3) DEG is the degree n of the polynomial $P(x)$.

The outputs of the program are (1) the coefficients $q_0, q_1, q_2, \dots, q_{n-2}$ of the quotient $Q(x)$, and (2) the coefficients $r_0 = q_{n-1}$ and $r_1 = q_n$ of the remainder $R(x)$.

```
for(j = 0; j <= DEG; j++) q[j] = c[j];
for(j = 1; j < DEG; j++) {
    q[j] = q[j] - (q[j-1] * a0);
    q[j+1] = q[j+1] - (q[j-1] * b0);
}
```

In the first loop, we copy the coefficients $c[j]$ to $q[j]$ for j from 0 to n . Here we copy $c[0]$ to $q[0]$.

In the second loop, we compute the quotient coefficients $q[j]$ for j from 1 to $n - 2$ and remainder coefficients $q[j]$ for j from $n - 1$ to n . In this second loop, we actually compute $q[1]=c[1]-a0*q[0]$. We then compute $q[j]=c[j] - b0*q[j-2] - a0*q[j-1]$ for j from 2 to $n - 1$. Finally we compute $q[n]=c[n] - b0*q[n-2]$.

6 COMPUTING $f()$ AND $g()$ AND DERIVATIVES

Consider the remainder $R(x) = r_0x + r_1$ obtained when we do long division of $P(x)$ by the quadratic divisor $K(x)$. The coefficient r_0 is a function of a and b , since $r_0 = q_k = c_k - bq_{k-2} - aq_{k-1}$, where $k = n - 1$. Thus we can rewrite r_0 as $q_k(a, b) = f(k, a, b)$, where $k = n - 1$. Similarly the coefficient r_1 is a function of a and b , since $r_1 = q_k = c_k - bq_{k-2}$ where $k = n$. Thus we can rewrite r_1 as $q_k(a, b) = g(k, a, b)$, where $k = n$.

Let us define a new function $F(k, a, b) = q_k, k = 0, 1, 2, \dots, n$. Thus our functions $f()$ and $g()$ and be defined as follows $f(a, b) =$

$F(n-1, a, b)$ and $g(a, b) = F(n, a, b)$. From the computation in Equation 1 and Equation 2 (in Section 1) of the coefficients q_k , with $k = 0, 1, 2, \dots, n$, we can define $F(k, a, b)$ casewise as follows.

$$F(k, a, b) = \begin{cases} c_0 & k = 0 \\ c_1 - c_0 a & k = 1 \\ c_k - aF(k-1, a, b) - bF(k-2, a, b) & k = 2, 3, \dots, n-1 \\ c_k - bF(k-2, a, b) & k = n \end{cases}$$

We can compute the partial derivatives $F_a()$ and $F_b()$ by differentiating the above definitions. The partial derivative $F_a()$ and its C program implementation `long double dFa()` are

$$F_a(k, a, b) = \begin{cases} 0 & k = 0 \\ -c_0 & k = 1 \\ -aF_a(k-1, a, b) - F(k-1, a, b) & k = 2, 3, \dots, n-1 \\ -bF_a(k-2, a, b) & k = n \end{cases}$$

```
long double dFa(int k)
{
    if(k == 0) return 0.0;
    if(k == 1) return -q[0];
    if(k >= 2 && k < DEG) return -a*dFa(k-1)-q[k-1]-b*dFa(k-2);
    if(k == DEG) return -b*dFa(k-2);
}
```

The partial derivative $F_b()$ and its C program implementation `long double dFb()` are

$$F_b(k, a, b) = \begin{cases} 0 & k = 0, 1 \\ -aF_b(k-1, a, b) - bF_b(k-2, a, b) & k = 2, 3, \dots, n-1 \\ -bF_b(k-2, a, b) - F(k-2, a, b) & k = n \end{cases}$$

```
long double dFb(int k)
{
    if(k == 0 || k == 1) return 0.0;
    if(k >= 2 && k < DEG) return -a*dFb(k-1)-b*dFb(k-2)-q[k-2];
    if(k == DEG) return -b*dFb(k-2)-q[k-2];
}
```

The four partial derivatives $f_a()$, $f_b()$, $g_a()$, $g_b()$ that we need to solve Equation 3 are therefore:

$$\begin{aligned} f_a(a, b) &= F_a(n-1, a, b) \\ f_b(a, b) &= F_b(n-1, a, b) \\ g_a(a, b) &= F_a(n, a, b) \\ g_b(a, b) &= F_b(n, a, b) \end{aligned}$$

7 IMPLEMENTATION OF NEWTON'S METHOD

Given the values of the coefficients $c_0, c_1, c_2, \dots, c_n$ of the dividend polynomial $P(x)$, and the initial guesses a_0 and b_0 of the coefficients of the divisor polynomial $K_0(x)$, the following C code fragment carries out Newton's method iteratively, at most NTRIES number of times, or until $\{a_0 \neq 0 \text{ and } |(a_1 - a_0)/a_0| < \epsilon \text{ and } b_0 \neq 0 \text{ and } |(b_1 - b_0)/b_0| < \epsilon\}$, whichever comes first. The long division algorithm

is the one discussed in Section 5, and the partial derivatives are recursively computed using the code in Section 6.

```
for(i = 1; i <= NTRIES; i++) {
    // Long division algorithm
    for(j = 0; j <= DEG; j++) q[j] = c[j];
    for(j = 1; j < DEG; j++) {
        q[j] = q[j] - (q[j-1] * a0);
        q[j+1] = q[j+1] - (q[j-1] * b0);
    }
    // Remainder coefficients r[0] and r[1]
    ff = q[DEG-1];
    gg = q[DEG];
    if(ff == 0.0 && gg == 0.0) break;
    // The four partial derivatives
    fa = dFa(DEG-1);
    fb = dFb(DEG-1);
    ga = dFa(DEG);
    gb = dFb(DEG);
    // Computing the determinant
    dt = fa * gb - ga * fb;
    if(ABS(dt) < ZERO) exit(1);
    da = (ff * gb - gg * fb) / dt;
    db = (gg * fa - ff * ga) / dt;
    // Computing new values a1 and b1
    a1 = a0 - da;
    b1 = b0 - db;
    if(a0 != 0.0 && ABS(da/a0) < EPS &&
        b0 != 0.0 && ABS(db/b0) < EPS) break;
    a0 = a1;
    b0 = b1;
}
```

The source code of the complete C program `qfactor.c` is available by writing email to the author. The program contains input and output so that it is a stand-alone program. It looks for one quadratic factor, given initial values of a and b , and after the quadratic factor is found, computes the two roots of the quadratic.

8 USING QFACTOR.C PROGRAM

The program requires that the user enter the $(n+1)$ real coefficients of the degree n polynomial $P(x) = c_0x^n + c_1x^{n-1} + c_2x^{n-2} + \dots + c_{n-1}x + c_n$, and then enter the initial guess for the two coefficients a_0 and b_0 of the quadratic factor $K(x) = x^2 + a_0x + b_0$. These values can be entered using any of these formats: (1) integer like 23, -57, or 0, (2) decimal number like -10.75, 0.0, or 52.5, (3) scientific notation like -2.234e14 meaning $-2.234 \cdot 10^{14}$, or $76.34e-12$, meaning $76.34 \cdot 10^{-12}$, or (4) any mix of these formats.

Upon completing the entry of the coefficients of the polynomial and the initial guess of the coefficients of the quadratic factor, the program computes better and better guesses for the values of a and b , until the values are correct to a predetermined accuracy (EPS, ϵ) , or until the maximum number of iterations (NTRIES) of Newton's method is reached, whichever comes first. Then the program stops and prints (1) the degree $(n-2)$ quotient polynomial $Q(x)$, (2) the quadratic factor $K(x)$ that the Newton's method found, and (3) the two roots of this quadratic factor. At this point, the user can do one of three procedures.

Procedure 1. The user can stop.

Procedure 2. The user can enter a new set of starting values a_0 and b_0 in order to possibly find a new quadratic factor of the original degree n polynomial

Procedure 3. The user can stop the current program run, then run the program again, but using as input the degree $n - 2$ polynomial printed out by the previous run, using copy-paste from the previous run to the new program run, so that there is no retyping of coefficient values. Then we can continue this “collapsing” of degree of the polynomial until we get all the roots.

9 SOLVED EXAMPLES

For the examples given here, we sometimes have to break a single long line of computer output (printout) into two or more lines, so that the entire line can be seen on the page. Other than that, we made no modifications in the computer output.

9.1 Example from Rafiq [8]

Find a quadratic factor of the following degree four polynomial, which is Example 3 in Rafiq [8]: $P(x) = 1.9520 \cdot 10^{-14}x^4 - 9.5838 \cdot 10^{-11}x^3 + 9.7215 \cdot 10^{-8}x^2 + 1.671 \cdot 10^{-4}x - 0.20597$. When we start with $a_0 = 100$ and $b_0 = -100$, we get the quadratic factor $x^2 + 163.940632x - 1452496.708774$, with roots 1126.009751 and -1289.950382 , which are the same real roots found by Rafiq [8].

```
Enter degree of polynomial [>2]: 4
Enter 5 coeffs decr order: 1.952e-14 -9.5838e-11
9.7215e-08 0.0001671 -0.20597
Deg 4 poly entered: 1.952e-14 -9.5838e-11 9.7215e-08
0.0001671 -0.20597
Enter a b [0 0 to end]: 100 -100
Quad coeffs:
Iter 0: 100.0000000000000000 -100.0000000000000000
Iter 1: -204.3860830346749444 -1925019.3555461554128669
Iter 2: 411.0866081826201767 -1134217.9762273866740543
Iter 3: 210.0010925603964294 -1393117.1294690310264741
Iter 4: 165.9440620727080588 -1449912.9227901836666206
Iter 5: 163.9446382221588250 -1452491.5408436994840713
Iter 6: 163.9406316534402896 -1452496.7087738420658525
Iter 7: 163.9406316373749578 -1452496.7087738420658525
Iter 8: 163.9406316373749578 -1452496.7087738420659662
Deg 2 Quotient: 0.000000 -0.000000 0.000000
Quadratic factor: x^2 +163.940632x -1452496.708774
Roots are:1126.009751 and -1289.950382
```

When we start with $a_0 = -5000$ and $b_0 = 6000000$, we get the quadratic factor $x^2 - 5073.674238x + 7264554.707450$, with complex roots $2536.837119 + 910.501037i$ and $2536.837119 - 910.501037i$, which are the same complex roots found by Rafiq [8].

```
Enter a b [0 0 to end]: -5000 6000000
Quad coeffs:
Iter 0: -5000.0000000000000000 6000000.0000000000000000
Iter 1: -4925.3135204350711991 6739960.3174731292501747
Iter 2: -5099.3963153108729891 7334285.4845929701623390
Iter 3: -5074.1386846732093225 7265539.2090234159468309
Iter 4: -5073.6743357811892570 7264554.7901784460523231
Iter 5: -5073.6742381947481726 7264554.7074500004719084
Iter 6: -5073.6742381947520069 7264554.7074500292092125
```

```
Deg 2 Quotient: 0.000000 0.000000 -0.000000
Quadratic factor: x^2 -5073.674238x +7264554.707450
Roots are:2536.837119+910.501037i and
2536.837119-910.501037i
```

9.2 Laguerre polynomials [12]

Find a quadratic factor of the fifth degree Laguerre polynomial $L_5(x) = (-x^5 + 25x^4 - 200x^3 + 600x^2 - 600x + 120)/120$. For purposes of finding quadratic factors, we can drop the factor $1/120$ and multiply by -1 to get the simpler polynomial $-120 \cdot L_5(x) = x^5 - 25x^4 + 200x^3 - 600x^2 + 600x - 120$. If we start with $a_0 = -1$ and $b_0 = 1$, the roots found are 1.413403 and 0.263560, which are the same roots given by Salzer [11].

```
Enter degree of polynomial [>2]: 5
Enter 6 coeffs decr order: 1 -25 200 -600 600 -120
Deg 5 poly entered: 1 -25 200 -600 600 -120
Enter a b [0 0 to end]: -1 1
Quad coeffs:
Iter 0: -1.0000000000000000 1.0000000000000000
Iter 1: -1.4742143692395518 0.4460485972359056
Iter 2: -1.6512809504470201 0.3744097884444698
Iter 3: -1.6764817698291903 0.3724802596240043
Iter 4: -1.6769632072206185 0.3725169354173948
Iter 5: -1.6769633788246361 0.3725169621487077
Iter 6: -1.6769633788246577 0.3725169621487120
Deg 3 Quotient: 1.000000 -23.323037 160.515605
-322.132982
Quadratic factor: x^2 -1.676963x +0.372517
Roots are: 1.413403 and 0.263560
```

If we run the qfactor program again, giving the degree three quotient above as input, namely the polynomial $1.000000x^3 - 23.323037x^2 + 160.515605x - 322.132982$, and if we start with $a_0 = -1$ and $b_0 = 1$, the roots found are 7.085809 and 3.596426, which are the same roots given by Salzer [11]. The remaining degree one quotient, namely $1.000000x - 12.640802$, has root 12.640802.

```
Enter degree of polynomial [>2]: 3
Enter 4 coeffs decr order: 1.000000 -23.323037
160.515605 -322.132982
Deg 3 poly entered: 1 -23.323 160.516 -322.133
Enter a b [0 0 to end]: -1 1
Quad coeffs:
Iter 0: -1.0000000000000000 1.0000000000000000
Iter 1: -6.7919792691737646 14.6899797401748590
Iter 2: -9.4972166412802723 21.8904845703736939
Iter 3: -10.4967954382757727 24.8820133132670366
Iter 4: -10.6762176607512623 25.4632166840639728
Iter 5: -10.6822283676961523 25.4835643087874602
Iter 6: -10.6822350233573271 25.4835874017388127
Iter 7: -10.6822350233654133 25.4835874017672732
Iter 8: -10.6822350233654132 25.4835874017672732
Deg 1 Quotient: 1.000000 -12.640802
Quadratic factor: x^2 -10.682235x +25.483587
Roots are: 7.085809 and 3.596426
```

To summarize, the five roots of the Laguerre polynomial $L_5(x)$ are 0.263560, 1.413403, 3.596426, 7.085809, and 12.640802, which are the five roots given by Salzer [11].

9.3 Legendre polynomials [13]

Find a quadratic factor of the eight degree Legendre polynomial $P_8(x) = (6435x^8 - 12012x^6 + 6930x^4 - 1260x^2 + 35)/128$. This polynomial is symmetric since $P_8(-x) = P_8(x)$, and so we only need to find the positive roots. For purposes of finding quadratic factors, we can drop the factor $1/128$. If we start with $a_0 = -1$ and $b_0 = 1$, the roots found are 0.525532 and 0.183435, which are the same roots given by Lowan [5].

```
Enter degree of polynomial [>2]: 8
Enter 9 coeffs decr order: 6435 0 -12012 0 6930 0
-1260 0 35
Deg 8 poly entered: 6435 0 -12012 0 6930 0 -1260 0 35
Enter a b [0 0 to end]: -1 1
Quad coeffs:
Iter 0: -1.0000000000000000 1.0000000000000000
Iter 1: -0.9165466406754666 0.6862151332071878
Iter 2: -0.8481340483475369 0.4592487346715779
Iter 3: -0.7940460635972765 0.2991915280826729
Iter 4: -0.7537103360782697 0.1922807479153829
Iter 5: -0.7267567446934254 0.1292608837302444
Iter 6: -0.7128130963480143 0.1020724784389720
Iter 7: -0.7091617860047420 0.0966138318667571
Iter 8: -0.7089674431118241 0.0964011661958203
Iter 9: -0.7089670524130436 0.0964008497335634
Iter 10: -0.7089670524119788 0.0964008497328791
Iter 11: -0.7089670524119788 0.0964008497328791
Deg 6 Quotient: 6435.000000 4562.202982 -9397.887867
-7102.593104 2800.459878 2670.129796 363.067339
Quadratic factor: x^2 -0.708967x +0.096401
Roots are:0.525532 and 0.183435
```

If we start with $a_0 = -1$ and $b_0 = 0$, the roots found are 0.960290 and 0.796666, which are the same roots given by Lowan [5].

```
Enter a b [0 0 to end]: -1 0
Quad coeffs:
Iter 0: -1.0000000000000000 0.0000000000000000
Iter 1: -1.3479941413924910 0.3763440860215054
Iter 2: -1.9508272202635273 0.9464537929883982
Iter 3: -1.8603954658225640 0.8624518647859902
Iter 4: -1.7996517864098767 0.8055078706089895
Iter 5: -1.7674303038954489 0.7750142253360990
Iter 6: -1.7577791385829587 0.7658177616274100
Iter 7: -1.7569619157828724 0.7650360866802702
Iter 8: -1.7569563341699185 0.7650307375201544
Iter 9: -1.7569563339111630 0.7650307372719293
Iter 10: -1.7569563339111630 0.7650307372719293
Deg 6 Quotient: 6435.000000 11306.014009 2929.200130
-3502.971512 -1465.496120 105.068188 45.749796
Quadratic factor: x^2 -1.756956x +0.765031
Roots are:0.960290 and 0.796666
```

By symmetry of the Legendre polynomial, the eight roots are therefore ± 0.183435 , ± 0.525532 , ± 0.796666 , and ± 0.960290 .

9.4 Constant B3 of Bifurcation Theory [6]

"B3 is the third bifurcation point of the logistic map $x_{k+1} = rx_k(1 - x_k)$, which exhibits period doubling shortly before the onset of

chaos. Computations using a predecessor algorithm to PSLQ found that B3 is a root of the polynomial, $0 = 4913 + 2108t^2 - 604t^3 - 977t^4 + 8t^5 + 44t^6 + 392t^7 - 193t^8 - 40t^9 + 48t^{10} - 12t^{11} + t^{12}$. if we start with $a_0 = -1$ and $b_0 = 1$, the roots found are 3.960769 and 3.544090, and 3.544090 is the same root given in MathOverflow [6].

```
Enter degree of polynomial [>2]: 12
Enter 13 coeffs decr order: 1 -12 48 -40 -193 392 44
8 -977 -604 2108 0 4913
Deg 12 poly entered: 1 -12 48 -40 -193 392 44 8 -977
-604 2108 0 4913
Enter a b [0 0 to end]: -1 1
Quad coeffs:
Iter 0: -1.0000000000000000 1.0000000000000000
Iter 1: 0.9604192460035566 -0.7738861558109435
Iter 2: -6.0041965546515304 -11.2421615740261158
Iter 3: -5.4731756863249682 -10.4464404566472117
Iter 4: -4.9946061671995209 -9.7292755261615639
Iter 5: -4.5643126702724995 -9.0843440289446050
Iter 6: -4.1786778608638634 -8.5060436004679316
Iter 7: -3.8347000133191092 -7.9893710690607148
Iter 8: -3.5301517678797782 -7.5295554740640297
Iter 9: -3.2640087644562776 -7.1207867148359447
...
Iter 41: -7.5048590119585681 14.0373219974098523
Deg 10 Quotient: 1.000000 -4.495141 0.227279 24.805436
-10.029086 -31.468765 -51.387132 64.084002 225.279118
187.120245 349.995533
Quadratic Factor: x^2 -7.504859x +14.037322
Roots are: 3.960769 and 3.544090
```

10 PROGRAM LIMITATIONS

The C program presented here uses the data type long double for all floating point computations. The number of decimal digits of precision of long double depends on the hardware and the C compiler used. In this paper, we used an Intel x86_64 and gcc-11.4.0, which provide 17-18 digits of precision. For problems where coefficients of the polynomial exceed 18 digits, the C program given here is not the right tool. Instead, the algorithms given here should be implemented using the GNU MultiPrecision (GMP) library.

11 CONCLUSION

This paper derives the mathematical formulas for numerically computing a quadratic factor $K(x) = x^2 + ax + b$ of a degree n polynomial $P(x) = c_0x^n + c_1x^{n-1} + c_2x^{n-2} + \dots + c_{n-1}x + c_n$ using Newton's Method. It also provides a C program implementation of this method. The author believes that the C program implementation of the computation of the quotient $Q(x) = q_0x^{n-2} + q_1x^{n-3} + \dots + q_{n-2}$ and the remainder is $R(x) = r_0x + r_1$ is new, although the Division Algorithm is quite old and well known, the particular C program implementation in this paper is new. In particular, the C program implementation of the recursive evaluation of the partial derivatives $\partial r_0/\partial a$, $\partial r_0/\partial b$, $\partial r_1/\partial a$ and $\partial r_1/\partial b$ is new.

REFERENCES

- [1] Burachik, R.S., Caldwell, B.I., Kaya, C.Y., "A generalized multivariable Newton method", as cited in <https://fixedpointtheoryandalgorithms.springeropen.com/articles/10.1186/s13663-021-00700-9>

- [2] Eagan, N. and Hauser, G., "Newton's Method on a System of Non-linear Equations" as cited in https://www.cmu.edu/math/undergrad/suami/pdfs/2014_newton_method.pdf
- [3] Gaik, T.K., Kahar, R.A., Long, K.S., "Solving non-linear systems by Newton's method using spreadsheet Excel" as cited in <https://core.ac.uk/download/pdf/12006987.pdf>
- [4] Licht, M., "Newton's method", UC San Diego, Winter Quarter 2021, as cited in <https://mathweb.ucsd.edu/mlight/wina2021/pdf/lecture11.pdf>
- [5] Lowan, A.N., Davids, N., Levenson, A., "Table of zeroes of the Legendre Polynomials of order 1-16 and the weight coefficients for Gauss' mechanical quadrature formula", as cited in <https://www.ams.org/journals/bull/1942-48-10/S0002-9904-1942-07771-8/S0002-9904-1942-07771-8.pdf>
- [6] Mathoverflow, "What Are Some Naturally-Occurring High-Degree Polynomials?", as cited in <https://mathoverflow.net/questions/27324/what-are-some-naturally-occurring-high-degree-polynomials>
- [7] Overton, M., "Quadratic Convergence of Newton's Method" in Numerical Computing, Spring 2017, as cited in <https://cs.nyu.edu/overton/NumericalComputing/newton.pdf>
- [8] Rafiq, N., Shams, M., Mir, N.A., Gaba, Y.U., "A Highly Efficient Computer Method for Solving Polynomial Equations Appearing in Engineering Problems", as cited in <https://www.hindawi.com/journals/mpe/2021/9826693/>
- [9] Remani, C., "Numerical Methods for Solving Systems of Nonlinear Equations" as cited in <https://www.lakeheadu.ca/sites/default/files/uploads/77/docs/RemaniFinal.pdf>
- [10] Saheya, B., Chen, G.Q., Sui, Y.K., Wu, C.Y., "A new Newton-like method for solving nonlinear equations" as cited in <https://springerplus.springeropen.com/articles/10.1186/s40064-016-2909-7>
- [11] Salzer, H.E., Zucker, R., "Table of the zeros and weight factors of the first fifteen Laguerre polynomials", as downloaded from <https://projecteuclid.org/journals/bulletin-of-the-american-mathematical-society/volume-55/issue-10/Table-of-the-zeros-and-weight-factors-of-the-first-bams/1183514167.full>
- [12] Wikipedia, "Laguerre polynomials", as cited in https://en.wikipedia.org/wiki/Laguerre_polynomials
- [13] Wikipedia, "Legendre polynomials", as cited in https://en.wikipedia.org/wiki/Legendre_polynomials
- [14] Wikipedia, "Proof of quadratic convergence for Newton's iterative method" in the article 'Newton's method' as cited in https://en.wikipedia.org/wiki/Newton's_method

ChattyTicket: Classifying Emotion and Toxicity in Valorant Chats using Multi-Task Learning with Bi-LSTM Algorithm and BERT

Johndel N. Encabo
College of Science and Computer
Studies
De La Salle University - Dasmariñas
EJN2008@dlsud.edu.ph

Maria Cassandra B. Vitug
College of Science and Computer
Studies
De La Salle University - Dasmariñas
VMB0604@dlsud.edu.ph

Tita R. Herradura
College of Science and Computer
Studies
De La Salle University - Dasmariñas
trherradura@dlsud.edu.ph

ABSTRACT

This study investigates communication dynamics within the popular multiplayer game Valorant, particularly focusing on the challenges posed by toxic conversations despite its potential for fostering connections, especially during the pandemic when entertainment serves as a form of leisure. To address this issue, we propose a novel multi-task learning architecture that integrates Bi-LSTM (Bidirectional Long-Short Term Memory network) and BERT (Bidirectional Encoder Representations from Transformers) models. This architecture was chosen for its superior performance classification problem and a BERT pre-trained model which can provide additional features as a backbone of the model.

Experimentally, our classifier achieves an accuracy of 91.81 percent in toxicity prediction and 86.74 percent in emotion prediction, shedding light on prevalent emotions such as anger and instances of cyberbullying. The insights garnered from these results hold the potential to cultivate healthier gaming communities. We introduce the ChattyTicket API for chat text evaluation, alongside a web application that has garnered positive user feedback, featuring its usability and interactivity. Continuous revisions guided by constructive criticism, are aimed at enhancing the platform and improving user experience. These revisions address challenges such as discerning sarcasm and fostering a more positive gaming environment.

KEYWORDS

Toxicity, emotion, Bi-LSTM, BERT backbone, toxicity classification, emotion classification, multi-task learning architecture, deep learning, game chat analysis

1 INTRODUCTION

Amidst the pandemic, individuals sought solace in their homes, turning to various forms of entertainment, including video games, which became a significant leisure activity, especially among the youth. In a time marked by social isolation, multiplayer games like Valorant emerged as a refuge, offering a platform for forging connections, shared experiences, and camaraderie. The interactive nature and immersive storytelling of these games not only entertain but also serve as therapeutic outlets, contributing to their widespread appeal as means of social interaction during the pandemic [11, 12, 19].

Valorant, a prominent first-person shooter game, gained traction during this period, emphasizing the crucial role of effective communication in its team-based gameplay. While features like text and voice chat enrich the gaming experience by facilitating strategic

collaboration, the competitive and anonymous nature of these platforms often leads to toxic interactions among players. This toxicity, characterized by verbal abuse, harassment, and bullying, has been pervasive, affecting not only the gaming experience but also the mental well-being of players [1, 10, 14, 22, 26].

Communication in chat, whether through text or voice, often carries with it a range of emotions. In competitive games like Valorant, players frequently express their reactions to their teammates, which can manifest as anger, sadness, happiness, disgust, or fear, depending on the circumstances. These reactions can significantly impact the outcome of the game, either positively or negatively as they can influence the emotional state of others. As Davidson et al (1994) [6] noted, universal emotions are characterized by distinct signals, physiologies, and timelines, with variations in onset, duration, and decline. Typically, emotions do not endure beyond an hour, and if they persist for an extended period without interruption, they are more likely to be classified as a mood [6]. This understanding allows researchers to examine the emotional content of text and its effect on gameplay, providing insights into player experience.

The prevalence of gaming toxicity has spurred researchers to explore deep learning algorithms, such as Bi-LSTM, for analyzing the emotional and toxic content of in-game chat. Studies have shown promising results, with algorithms like Bi-LSTM demonstrating high accuracy in detecting abusive or toxic language having a high classification accuracy compared to other existing cyberbullying detection algorithms. Bi-LSTM is to detect abusive or toxic content in any messaging application. [18] By employing multi-task learning architectures, researchers aim to further enhance performance and provide developers and community managers with effective tools to identify and mitigate toxic behavior within online gaming communities.

This study employs the Bi-LSTM deep learning algorithm to classify emotion and toxicity in Valorant text chat, aiming to enhance the gaming experience for all players. The research focuses on developing web applications to gather data, aiming to improve the online gaming experience by identifying emotions and tackling toxic behavior in text chat. It introduces ChattyTicket, a system that utilizes a Bi-LSTM algorithm for classifying emotion and toxicity in Valorant chats, aiming to improve the gaming experience.

The research targets players within the South-East Asia server, the desired population sample comprises one thousand Valorant players from this region. The dataset gathered only comprises of Valorant text chat logs, it excludes communication forms beyond text chat like voice chat and external messaging platforms.

The study’s findings are anticipated to benefit the Valorant community by offering insights into common toxic language and prevalent emotions during gameplay. Additionally, developers can utilize the data to refine in-game censorship mechanisms and possibly initiate warning and banning systems to filter out the players who are toxic. Future researchers can explore alternative algorithms for improved accuracy in toxicity detection and emotion classification. Leveraging the Bi-LSTM algorithm, developers can create practical applications for monitoring and managing in-game communication, enhancing player experience, and promoting a healthier gaming community.

2 RELATED WORKS

Online gaming has witnessed a surge in popularity during the pandemic, facilitating interactions and shared experiences among millions worldwide. However, this rise has coincided with an increase in toxic behavior, including harassment and hate speech, posing challenges to players’ mental well-being [4]. Shen et al. (2020) suggest that toxic behavior in online games is often rationalized and perpetuated by players, with exposure in prior games increasing the likelihood of future toxic acts, particularly among experienced players [25]. Tyler (2020) notes that certain competitive games like Counter-Strike: Global Offensive and League of Legends are particularly prone to toxicity, with varying degrees of enforcement and consequences for toxic behavior. To address this issue, there is a need for effective toxicity detection models, such as the one proposed by the researchers in this study [10, 20, 21].

Asgher et al. (2022) highlight the challenge of accurately analyzing text emotions and propose a Deep Learning approach, specifically Bi-LSTM, for improved emotion detection. Their study focuses on enhancing emotion classification accuracy, albeit with limitations regarding word embedding and language scope [2]. Lee et al. (2023) suggests the use of transformer transfer learning for emotion annotation, offering a faster alternative to manual annotation, albeit with potential limitations in capturing subtle social emotions [11]. Meanwhile, the researchers in this study adopt a Modified Discrete Emotions model, adding a "neutral" category to Dr. Ekman’s well-established model, which identifies six basic emotions [6]. This model facilitates the categorization of sentiments and enhances understanding of associated emotions, thereby providing insights into the emotional states expressed in textual data."

Toxicity in online gaming can manifest in various forms, including verbal toxicity, grieving behavior, and cheating. Dzigurski (2022) further delineates sub-themes within these categories, underscoring the multifaceted nature of toxic behavior in gaming environments in his study of Toxicity in the game World of Tanks: A participant observation ethnography, thematic analysis, content analysis and autoethnography (Dissertation), this category can be seen on Table 1 [8]. Understanding these nuances is crucial for developing effective strategies to mitigate toxicity and foster a positive gaming experience for all users [1, 8, 14]. Table 1 comprises themes and sub-themes representing toxic behaviors observed in the game of World of Tanks. These classifications were from battle chats and were tallied across 120 games, totaling 383 occurrences across all classified chats [8].

Table 1: Occurrence of themes/sub-themes in battle chats

No	Theme/sub-theme	Occurrence	%	No. of battles	%
1	Gamesplaining	92	24	37	31
2	Ableism	89	23	53	44
3	Male preserve	60	15.7	36	30
4	Sarcasm	53	14	29	24
5	Positive	27	7	16	13.3
6	Blaming others	18	4.7	12	10
7	RNG complaints	13	3.4	4	3.3
8	Sexism	10	2.6	10	8.3
9	SPG complaints	5	1.3	2	1.7
10	MM complaints	5	1.3	5	4
11	Ageism	4	1	3	2.5
12	EBR complaints	3	0.8	1	0.8
13	Cyberbullying	/	/	3	2.5
14	Game complaints	2	0.5	2	1.7
15	Racism	1	0.26	1	0.8
16	Map Complaints	1	0.26	1	0.8

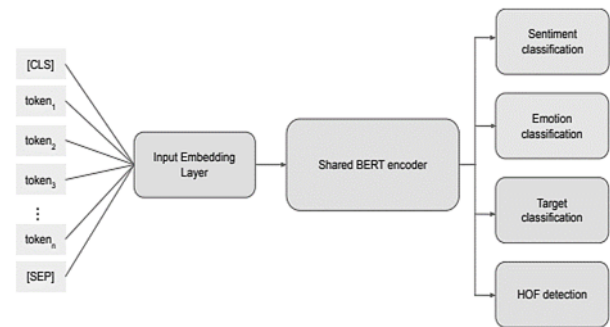


Figure 1: Vanilla Tree-like Architecture in with Sentiment, Emotion, Target Detection

Multi-Task Learning (MTL) has emerged as a powerful approach in machine learning, enabling the simultaneous training of multiple tasks to capture generalized and complementary knowledge from specific tasks [3]. Huang et al. (2022) used MTL in their study on abuse and emotion classification, utilizing linguistic context and pretrained language models like BERT to enhance the algorithm’s representational capability. They employed different decoders and a cross-attention component, achieving superior performance compared to other methods [13].

Researchers often explore various architectures of MTL to optimize model performance. The focus of this study lies in Parallel Architectures, particularly the Vanilla Tree-like Architecture and its variants, which allow tasks to run in parallel, sharing certain layers for efficiency [3]. Plaza-del-Arco et al. (2021) applied Tree-Like Architecture with a Shared Bert Encoder in their study on hate speech and offensive language recognition, leveraging transfer learning to enhance model performance [7], their model architecture shown on the Figure 1.

Moreover, Parallel Feature-fusion is another MTL architecture that actively combines features from different tasks to create task-specific representations [3]. While Supervision at Different Feature Levels also shows promise, its applicability to this study may require further exploration. Overall, understanding and implementing different MTL architectures offers researchers opportunities to enhance model efficiency and performance across various NLP tasks.

Cruz and Cheng's (2022) research focus on advancing natural language processing technology for Filipino speakers, with the aim of developing resources and models for the Filipino language to enhance access to crucial services. By conducting a literature review, they identify areas for improvement and suggest further research directions, laying the groundwork for enhancing technology accessibility among Filipino-speaking communities [5]. Similarly, Molina et al. (2021) contribute to this goal by creating Tagalog language models using multi-source data and the BERT pre-training technique, addressing challenges in developing language models for resource-constrained languages [16].

Alampay et al. (2020) delve into the connection between cyberbullying/cybervictimization and empathy among adolescent Filipinos, underscoring the importance of understanding this relationship for intervention and prevention programs [24]. Pujante (2021) investigates the use and impacts of "trash talk" in gaming environments, shedding light on social dynamics, and emphasizing the need for promoting a positive gaming culture [17]. Ferrer et al. (2021) tackle the identification of profanity in the Filipino language, aiming to develop efficient methods for monitoring and controlling online communication to foster a civil and safe online environment [9].

Ong's (2022) project, together with collaborators, focuses on building a context-aware digital lexicon for Tagalog and English to address challenges in lacking local resources and continuously updated datasets [21]. Sagum et al. (2019) contributes to this effort by creating a Filipino WordNet using semi-supervised learning, further enriching language resources for Tagalog. These studies collectively contribute to advancing natural language processing technology for Filipino languages and fostering a safer and more inclusive online environment for Filipino speakers [23].

3 METHODOLOGIES

3.1 Equipment

This study utilized the following equipment: (1) personal computers and (2) internet. The website served as the platform for generating output, using Django to create the API connecting to the model deployed on an AWS server. Next.js was utilized to develop the interface featuring an input box for entering words/sentences, deployed on a serverless instance of Vercel.

3.2 Data Collection

The aim of data collection was to acquire authentic text chat data from the online game Valorant. Convenience sampling was employed, distributing the survey to public Facebook groups, Discord servers, group chats, and other online communities frequented by Valorant players aged 18 and above.

There was a total of 789 participants, all were asked to complete a survey questionnaire detailing their Valorant gameplay experience,

including interactions within the in-game chat. This information could be submitted by typing text into the provided input box or by attaching screenshots of their game chat box. Only in-game chat data was collected, excluding any other personal information like their full name. The survey questionnaire was integrated into a website developed by the researchers. Google Forms was also used as an alternative way to collect data on gameplay experience and text chat screenshots. Demographic information such as gender, country, age, and in-game details such as rank were also collected for further analysis. The dataset gathered consists of 4 columns: username, text/chat, toxicity label and emotion label.

A post-evaluation survey, distributed through Microsoft Forms, was sent to individuals who utilized the web application, inviting them to share feedback on their experience with the website. This feedback serves to guide future improvements to the application. To protect sensitive information like unfiltered datasets containing usernames, researchers pledged to delete screenshots and archive the dataset post the removal of any identifiable participant details. This archival and deletion process was implemented after the utilization of data for training and result acquisition.

3.3 Data Processing

The course of the system's data processing started in gathering the data from the survey answered by the participants. The screenshots of the in-game chat box were included, then extracted using OCR and manual extraction. Take note that OCR was used just for extraction and not included in the model made, and it does not affect the performance of it. We excluded the number of games played due to the absence of information given by the participants.

In the context of emotion classification for Valorant chats, Ekman's renowned model was applied, which delineates six primary emotions universally expressed and recognized across cultures: happiness, sadness, anger, fear, surprise, and disgust [6]. In this paper, a modified discrete emotion model was used, introducing an additional category, "neutral," to account for instances where game chats exhibit a lack of strong emotional intensity or are used to express information to teammates.

The proponents utilized Dzigurski's comprehensive classification framework from 2022, which encompasses various toxic behaviors prevalent in Valorant chats, as referenced in Table 1. To simplify the categorization process and enhance the coherence of the analysis, similar forms of toxicity were grouped as seen on Table 2. Ableism, Male Preserve, Ageism, Racism, and Sexism were labeled "Multiple Discrimination". Toxicity related to in-game experiences, including RNG Complaints, MM Complaints, EBR Complaints, Map Complaints, and Game complaints, was placed within the category of "Gameplay Experience Complaints". Lastly, the "Gamesplaining" category was created by merging instances from both Gamesplaining and Blaming Others [1, 14]. The proponents omitted the other toxicity classifications as they were not relevant to the game under study.

3.3.1 Data Labelling Validation. To ensure the accuracy and reliability of emotion classification, the proponents engaged the expertise of a board-certified psychometrician. Collaborative consultations were conducted with the psychometrician for suitable labels

Table 2: Game Toxicity Classifications

Toxicity	Combined Classifications
Sexism Ableism Male Preserve Ageism Racism	Multiple Discrimination
RNG complaints Map complaints MM complaints EBR complaints Game complaints	Gameplay Experience Complaints
Sarcasm	Sarcasm
Cyberbullying	Cyberbullying
Blaming others	Blaming others
Gamesplaining	Gamesplaining
Not Toxic	Not Toxic

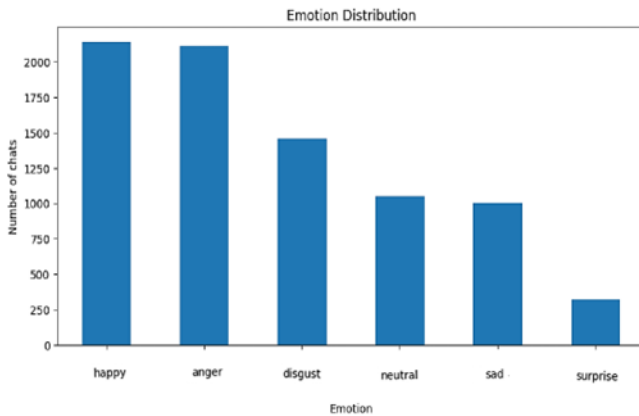


Figure 2: Emotion Distribution After Data Labelling

for the dataset, specifically utilizing Ekman’s six basic emotions and a neutral category.

The labeled dataset then underwent rigorous validation process in subsequent sessions with the psychometrician. The psychometrician provided official endorsement by signing the validation form and issuing a certificate upon successful validation. This action affirmed the robustness of the emotion classification framework employed in this study. After labelling the emotions and toxicity, uneven distribution of them was discovered as shown on Figure 2. This indicates augmentation of the dataset was required.

3.3.2 Data Augmentation. To address an unbalanced dataset characterized by a low number of categories in emotion, proponents employed oversampling techniques. This process aimed to augment the representation of underrepresented categories, as illustrated in Figure 2 of the emotion classification results. Oversampling was applied to the toxicity class to achieve balance between the two classes. Subsequently, undersampling was conducted on the toxicity class to mitigate the disproportionately high number of non-toxic

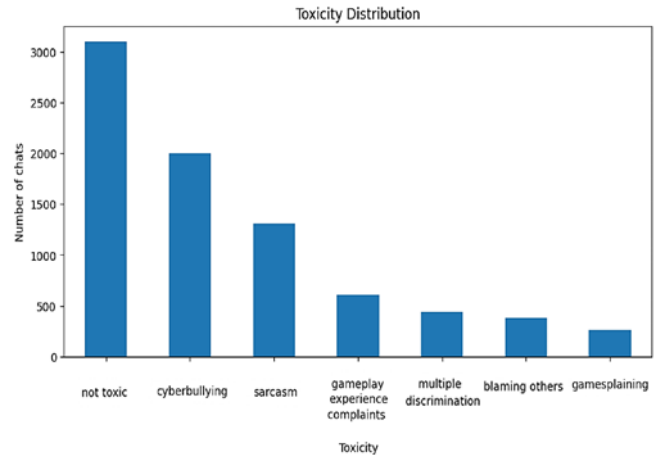


Figure 3: Toxicity Distribution After Data Labelling

instances, as depicted in Figure 3. These efforts yielded a dataset comprising 8,274 rows, up from the original 8,086.

3.3.3 Data Preparation. Preparing the chat data was the next step, which utilized some of the NLP techniques including the removal of duplicates and removing some of special characters that might unrelated such as underscores, quotation marks, etc., however question mark and exclamation point was not removed due to its context in the game like it can be a disbelief in the in-game chat.

After the cleaning and augmenting, the dependent variables such as the emotions and toxicities undergone one-hot encoded to make categorical data possible to work with in the model. This one-hot encoded makes the categories into a finite set of label values, this consists of 1 and 0s, the position of 1 in the set determines the category. Using the Scikit-learn library for one-hot encoding facilitated the process, providing the flexibility to transform the encoded data back to its categorical values when needed. The text inputs were prepared for the model using the BERT Tokenizer. The completed processed dataset was split into two parts: 80% for training and 20% for testing.

3.4 Model Architecture

The model consists of 4 major layers, first was first input layer which accepts the tokenize version of the chat data more specific is the input ids, then the label data in a one-hot encoded version. The second layer was the pre-trained BERT Layer, which is used for feature extraction which gives additional feature as the backbone of the model, receives data from input layer then produces a pooler output that is used in the next layer, the Classifier layer. This was split into two tasks connected by a trunk, the toxicity classification and emotion classification, using Bi-LSTM as the classifier for each task, this helps to retain information without duplicating the context using its memory [15].

Lastly, the output layer or the dense layer that accepts an input from the Bi-LSTM layers because there’s two task two output layer was made, first was for emotion that have a unit of 6, second was for toxicity which have a unit of 7. Both output layers have an activation

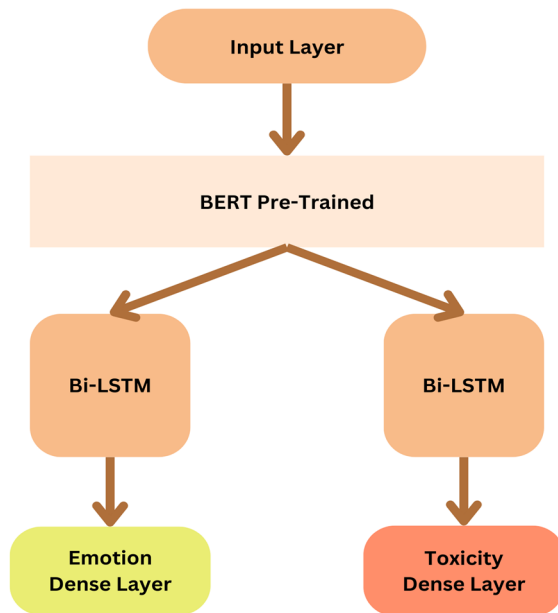


Figure 4: Multitask Classification Model

function SoftMax. The optimizer used was Adam with a learning rate of $1e-4$ with a learning decay of 0.01. Then a categorical cross entropy loss function for both toxicity and emotion were used as par multiple categories was used. The metrics used were discussed in Section 3.6.

3.4.1 Regularization Techniques. To prevent overfitting, the model employed dropout regularization with a rate of 0.6 up to 0.7. Additionally, task specific L1 and L2 regularization were applied. For the Emotion task, L1 and L2 coefficients were 0.0045 and 0.03 respectively, while for the Toxicity task, they were 0.006 and 0.03. These regularization techniques were calibrated manually during experimentation to optimize performance.

3.4.2 Hyperparameters Tuning. The following hyperparameters included: epoch which start from 30 until 20, then batch size from 32 to 70, also number of LSTM layers that have a value of 40 down from 50 after calibrations. Learning decay also applied which discussed on Section 3.4. The hyperparameter tuning done was a manual trial and error to find better results in the model. It was inefficient, due to time constraint resorting to this method has been chosen.

3.5 Model Implementation

The model was implemented using the python libraries such as keras for deep learning packages based on Tensorflow for the pre-built layers for the model like the Bi-LSTM, Input and Dense Layer, also the metrics is already built in because of Tensorflow, then Scikit-learn for preprocessing and metrics computation of the data like the one-hot encoder for the emotion and toxicity and the confusion matrix. For the pre-trained BERT, transformers library from HuggingFace was used, also includes the BERT tokenizer used for the tokenization of the text. Two varieties of BERT were used:

BERT-base-uncased is for general use and BERT-base-multilingual-cased for multi-language that can provide features from different languages.

3.6 Model Evaluation and Validation

In the model evaluation and validation phase, our primary objective is to ensure the accuracy and reliability of our trained model. Using manual analysis using metrics was used during experimentation, when found one of the metrics was low or indicate a possible overfitting then adjusting of the hyperparameter was done, from dropout, L1 and L2 regularization, batch size and number of LSTM layers. This process somehow helps mitigate overfitting and provides a robust assessment of the model's generalization ability. To quantify the model's performance, key metrics were used including precision, recall, F1 score, and accuracy. These metrics collectively offer insights into the model's predictive capabilities, balancing between correctness and completeness of predictions across the different classes.

Moreover, visual aid such as confusion matrices to gain a deeper understanding of the model's performance characteristics, identifying potential areas for improvement and assessing its discriminative power effectively. Through these comprehensive evaluation strategies, the aim is to ensure that our model not only learns accurately but also generalizes well to unseen data, thereby enhancing its utility and reliability in practical applications. After getting these results, another model was created with different BERT, a BERT with multilanguage capability.

4 RESULTS AND DISCUSSION

4.1 Model Performance

After conducting several experiments, the proponents compared the performance of two models using different BERT variants, this was shown on Table 3. The BERT Base model achieved an accuracy of 86.74%, a recall of 82.23%, and a precision of 91.17%. Its AUC is 98.85%, and the F1-Score is 63.50%. The BERT Multilingual model showed slightly better performance, with an accuracy of 87.70%, a recall of 84%, and a precision of 91.23%. Its AUC is 98.67%, and the F1-Score is 60.46%.

The model training several hyperparameters were used, but the best hyperparameters that were found, also been used in the model were 50 epochs, a batch size of 70, learning rate in Adam optimizer of 0.0001, then a LSTM (Long Short-Term Memory) layer of 0.75, also an L1 and L2 regularization was applied for both emotion and toxicity tasks with a value of 0.0045 and 0.03.

The utilization of Bi-LSTM as the classifier, supported by BERT from Transformers, demonstrates strong performance in identifying both toxicity and emotion especially the BERT Multilingual. However, toxicity exhibits superior performance compared to emotion. This is attributed to certain words or sentences lacking sufficient context to accurately determine the conveyed emotion; for instance, "GG" may signify happiness or sadness depending on the in-game situation. The performance disparity is evident in the confusion matrices depicted in Figures 3 and 4, which illustrate the classifier's predictive capabilities across various categories. Furthermore, the multitask architecture efficiently deployed yields dual outputs, enhancing overall output efficiency.

Table 3: Model’s Performance Score

Model BERT	Task	Accuracy %	Recall%	Precision%	AUC%	F1-Score
With Base	Toxicity	91.81	90.38	92.90	99.6	81.28
	Emotion	86.74	82.23	91.17	98.85	62.83
With Multilingual	Toxicity	93.88	92.85	94.77	99.63	77.96
	Emotion	87.70	84	91.23	98.67	60.46

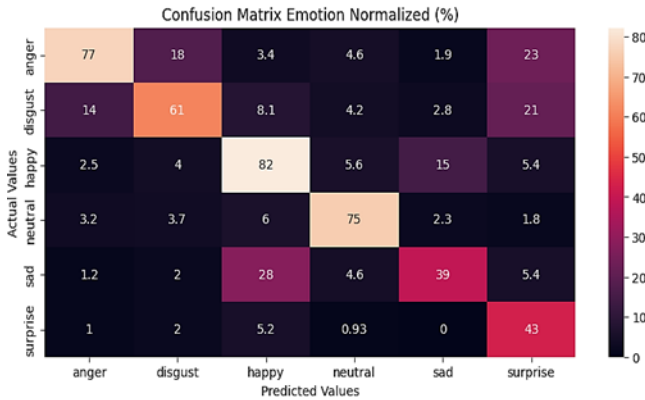


Figure 5: Confusion Matrix Emotion Normalized

Two confusion matrices were created to demonstrate the model with BERT Multilingual has capability in determining the toxicity and emotion of the chat in each category. These matrices were normalized for clarity and ease of interpretation. Figure 5 illustrates the normalized confusion matrix for emotion classification, depicted as a heatmap, where brighter colors denote higher values and darker colors indicate lower values. Figure 5 visually compares actual predictions with predicted values for each emotion classification, revealing successful identification rates. Specifically, the accuracy rates are 77% for anger, 81% for disgust, 82% for happy, 75% for neutral, 39% for sad, and 43% for surprise. However, 28% of instances where actual sadness is present are misclassified as happy due to the ambiguity of certain expressions like "GG", which could signify happiness or sadness depending on the context.

Figure 6 illustrates the confusion matrix for toxicity classification, presented as a heatmap. Brighter colors signify higher values, while darker colors represent lower values. The toxicity classification demonstrates effective prediction of actual values, with accuracy rates of 81% for blaming others, 88% for cyberbullying, 78% for gameplay experience complaints, 91% for gamesplaining, 82% for multiple discrimination, 60% for sarcasm, and 89% for not toxic. However, it is worth noting that there is a 33% chance of misclassifying actual non-toxic instances as sarcasm, likely due to the nuanced nature of sarcasm detection and the contextual complexity involved in distinguishing between genuine non-toxic communication and sarcastic remarks. Overall, the model performs well, achieving a high accuracy rate of 81.28% in predicting the actual classification.

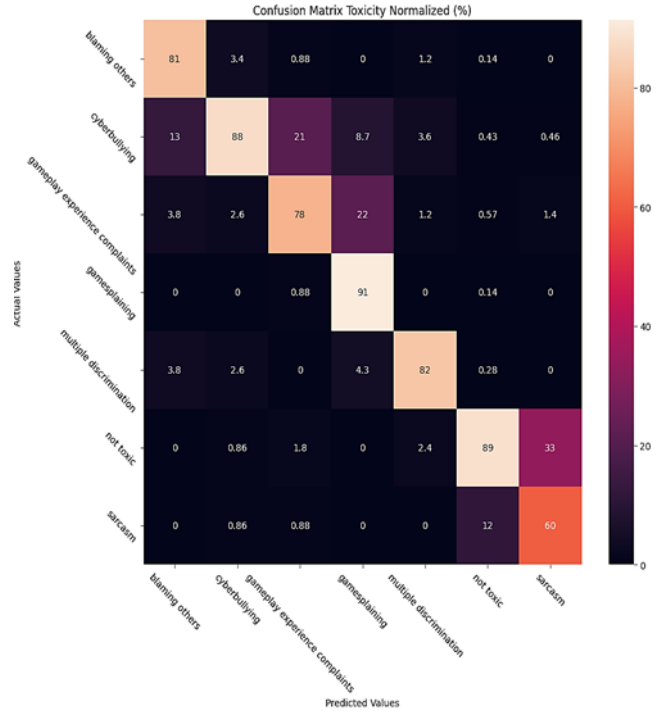


Figure 6: Confusion Matrix Toxicity Normalized

4.2 Evaluation Discussion

ChattyTicket’s evaluation reveals diverse user preferences and positive reception for its functionality, usability, performance, interactivity, and UI design. While most users are satisfied, valuable feedback suggests enhancements for future versions, including improved analytics display, more visual content, additional emotions in the model, and enhanced accuracy.

5 CONCLUSION AND FUTURE WORKS

5.1 Conclusion

In summary, the multi-task learning architecture, which integrates Bi-LSTM classifiers for toxicity and emotion with a BERT backbone, demonstrates significant effectiveness in predicting multiple classes for both tasks. The model achieved an accuracy of 93.88% in classifying toxicity and 87.7% in classifying emotion, highlighting the refined performance of the model with bert base multilingual. Despite accuracy improvements, challenges persist, particularly in discerning sarcasm within the ‘not toxic’ category due to contextual nuances.

Moreover, the development of an accessible API has facilitated the evaluation of chat text for both emotion and toxicity, showcasing accurate and efficient performance. The examination of datasets underscores the prevalence of toxic chats in gaming environments like Valorant, encompassing cyberbullying, sarcasm, and discrimination. The evaluation of the web application revealed diverse user preferences and garnered high praise for its utility, usability, performance, and interactivity. Constructive criticism has guided subsequent revisions, ensuring continuous enhancement to meet customer expectations and elevate the overall user experience.

In conclusion, the multi-task learning architecture, in conjunction with an accessible API, demonstrates promising results in predicting toxicity and emotion in chat text. The exploration of diverse datasets and user feedback has provided valuable insights into the prevalence of toxic behaviour in gaming environments and the effectiveness of the web application in meeting user needs. Constructive criticism has been instrumental in refining the platform, ensuring its ongoing alignment with user expectations, and enhancing the overall user experience.

5.2 Future Works

Future research should aim to focus on observing players during Valorant gameplay to provide more accurate labeling and contextualization of emotions, either through in-game observation or post-game interviews. To achieve this, researchers are advised to allocate at least a month or longer for data gathering, given the typical game duration of around 45 minutes. Extending the data collection period enables the accumulation of a more comprehensive dataset. Furthermore, researchers should consider incorporating additional classifiers such as Naïve Bayes or Logistic Regression to compare with Bi-LSTM performance and explore alternative architectures for multi-task learning beyond the Vanilla Tree-Like Structure to assess potential performance improvements.

REFERENCES

- Jenna Abrera, Maria Laureta, Kent Miculob, Fatima Valencia, Justin David Pineda, Jayvee Cabardo, and Lorena Rabago. 2019. Harassment Exposure Model Using Sentiment Analysis on Facebook Pages. In *2019 International Conference on Information Management and Technology*. 175–179.
- Muhammad Zubair Asghar, Adidah Lajis, Muhammad Mansoor Alam, Mohd Khairil Rahmat, Haidawati Mohamad Nasir, Hussain Ahmad, Mabrook S. Al-Rakhami, Atif Al-Amri, and Fahad R. Albogamy. 2022. A Deep Neural Network Model for the Detection and Classification of Emotions from Textual Content. *Complexity* 2022, 1 (2022), 8221121. <https://doi.org/10.1155/2022/8221121> arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1155/2022/8221121>
- Shijie Chen, Yu Zhang, and Qiang Yang. 2024. Multi-Task Learning in Natural Language Processing: An Overview. arXiv:2109.09138 [cs.AI] <https://arxiv.org/abs/2109.09138>
- Debating Communities and Networks XI. 2020. “Shat up u noob” – The primary causes of toxicity in online gaming communities. <https://networkconference.netstudies.org/2020Curtin/2020/05/13/shat-up-u-noob-the-primary-causes-of-toxicity-in-online-gaming-communities/>
- Jan Christian Blaise Cruz and Charibeth Cheng. 2021. Improving Large-scale Language Models and Resources for Filipino. arXiv:2111.06053 [cs.CL] <https://arxiv.org/abs/2111.06053>
- Richard Davidson, Paul Ekman, Nico Frijda, Harold Goldsmith, Jerome Kagan, Richard Lazarus, Jaak Panksepp, David Watson, and Lee Clark. 1994. How are emotions distinguished from moods, temperament, and other related affective constructs? *The nature of emotion: Fundamental questions. Series in affective science* (01 1994).
- Flor Miriam Plaza del Arco, Sercan Halat, Sebastian Padó, and Roman Klinger. 2022. Multi-Task Learning with Sentiment, Emotion, and Target Detection to Recognize Hate Speech and Offensive Language. arXiv:2109.10255 [cs.CL] <https://arxiv.org/abs/2109.10255>
- Sasa Dzigurski. 2022. *Toxicity in the game World of Tanks: A participant observation ethnography, thematic analysis, content analysis and autoethnography*. Master’s thesis. Uppsala University, Department of Game Design.
- Beverly P. Ferrer, Crystelle T. Tomilas, Lorenz P. Mallare, Benedict D. Pineda, Jr., Ana F. De Guzman, Pamela N. Siapno, Kathleen B. Payang, Michael B. Yu, Allen D. Bolaños, and Zareena D. Lee. 2021. A machine learning model for the profanity detection in the Filipino language. *Journal of Engineering Science and Technology Special special issue* (10 2021), 37–46.
- Enrico Gandolfi and Richard Ferdig. 2021. Sharing dark sides on game service platforms: Disruptive behaviors and toxicity in DOTA2 through a platform lens. *Convergence: The International Journal of Research into New Media Technologies* 28 (07 2021), 135485652110288. <https://doi.org/10.1177/13548565211028809>
- Vicpher Garnada. 2020. ONLINE GAMING ADDICTION AND ACADEMIC ATTITUDES: THE CASE OF COLLEGE STUDENTS IN THE PHILIPPINES. *INTERNATIONAL REVIEW OF HUMANITIES AND SCIENTIFIC RESEARCH* (03 2020), 417–426. https://www.researchgate.net/profile/Vicpher-Garnada/publication/339687790_ONLINE_GAMING_ADDICTION_AND_ACADEMIC_ATTITUDES_THE_CASE_OF_COLLEGE_STUDENTS_IN_THE_PHILIPPINES/links/5e5fc5a46fcdceba1c5c61/ONLINE-GAMING-ADDICTION-AND-ACADEMIC-ATTITUDES-THE-CASE-OF-COLLEGE-STUDENTS.pdf
- Karen Gasper, Lauren A. Spencer, and Danfei Hu. 2019. Does Neutral Affect Exist? How Challenging Three Beliefs About Neutral Affect Can Advance Affective Research. *Frontiers in Psychology* 10 (2019). <https://doi.org/10.3389/fpsyg.2019.02476>
- Yucheng Huang, Rui Song, Fausto Giunchiglia, and Hao Xu. 2022. A Multitask Learning Framework for Abuse Detection and Emotion Classification. *Algorithms* 15, 4 (2022). <https://doi.org/10.3390/a15040116>
- Sidney V. Irwin, Anjum Naweed, and Michele Lastella. 2023. The AACTT of Trash Talk: Identifying Forms of Trash Talk in Esports Using Behavior Specification. *Journal of Electronic Gaming and Esports* 1, 1 (2023), jege.2022–0024. <https://doi.org/10.1123/jege.2022-0024>
- Beakcheol Jang, Myeonghwi Kim, Gaspard Harerimana, Sang-ug Kang, and Jong Wook Kim. 2020. Bi-LSTM Model to Increase Accuracy in Text Classification: Combining Word2vec CNN and Attention Mechanism. *Applied Sciences* 10, 17 (2020). <https://doi.org/10.3390/app10175841>
- Shengyi Jiang, Yingwen Fu, Xiaotian Lin, and Nankai Lin. 2021. *Pre-trained Language Models for Tagalog with Multi-source Data*. Springer Nature Switzerland AG 2021, 210–223. https://doi.org/10.1007/978-3-030-88480-2_17
- Nicolas T. Pujante, Jr. 2021. Speech for Fun, Fury, and Freedom: Exploring Trash Talk in Gaming Stations. *Asian Journal of Language, Literature and Culture Studies* 4, 1 (January 2021), 1–11. <http://science.sdpublishers.org/id/eprint/217/>
- Pranav Kompally, Sibi Chakkaravarthy Sethuraman, Steven Walczak, Samuel Johnson, and Meenalosini Vimal Cruz. 2021. MaLang: A Decentralized Deep Learning Approach for Detecting Abusive Textual Content. *Applied Sciences* 11, 18 (2021). <https://doi.org/10.3390/app11188701>
- Ryan Labana, Jehan Hadjisaïd, Adrian Imperial, Kyeth Jumawid, Marc Jayson, M Lupague, and Daniel Malicdem. 2020. Online Game Addiction and the Level of Depression Among Adolescents in Manila, Philippines. *Central Asian Journal of Global Health* 9 (12 2020). <https://doi.org/10.5195/cajgh.2020.369>
- Sanghyub John Lee, JongYoon Lim, Leo Paas, and Ho Seok Ahn. 2023. Transformer transfer learning emotion detection model: synchronizing socially agreed and self-reported emotions in big data. *Neural Comput. Appl.* 35, 15 (jan 2023), 10945–10956. <https://doi.org/10.1007/s00521-023-08276-8>
- E. Ong. 2022. *Building a rich digital lexicon for low-resource Philippine languages. FilWordNet*. <http://filwordnet.dlsu.edu.ph/in-focus/digital-lexicon-philippine-languages-overview/>
- W. Opinion. 2020. *Toxicity in Gaming Is Dangerous. Here’s How to Stand Up to It*. <https://www.wired.com/story/toxicity-in-gaming-is-dangerous-heres-how-to-stand-up-to-it/>
- Ria Sagum, Aldrin Ramos, and Monique Llanes. 2019. FICOBU: Filipino WordNet Construction Using Decision Tree and Language Modeling. *International Journal of Machine Learning and Computing* 9 (02 2019), 103–107. <https://doi.org/10.18178/ijmlc.2019.9.1.772>
- Lee Jungmin Shannen Tadana, Kim Shin-Jeong. 2021. Empathy, cyberbullying, and cybervictimization among Filipino adolescents. *Child Health Nurs Res* 27, 1 (2021), 65–74. <https://doi.org/10.4094/chnr.2021.27.1.65> arXiv:<http://e-chnr.org/journal/view.php?number=1730>
- Cuihua Shen, Qiusi Sun, Taeyoung Kim, Grace Wolff, Rabindra Ratan, and Dmitri Williams. 2020. Viral vitriol: Predictors and contagion of online toxicity in World of Tanks. *Computers in Human Behavior* 108 (2020), 106343. <https://doi.org/10.1016/j.chb.2020.106343>
- Zicheng Zhu, Renwen Zhang, and Yuren Qin. 2022. Toxicity and prosocial behaviors in massively multiplayer online games: The role of mutual dependence, power, and passion. *Journal of Computer-Mediated Communication* 27, 6 (09 2022), zmacc017. <https://doi.org/10.1093/jcmc/zmac017> arXiv:<https://academic.oup.com/jcmc/article-pdf/27/6/zmac017/45870068/zmac017.pdf>

Utilizing Machine Learning Methods to Predict Student Re-engagement via Minecraft Data

Mylene Villegas

Department of Information
Systems and Computer Science
Ateneo de Manila University
Quezon City, Philippines
mylenevillegas@gmail.com

ABSTRACT

This study focused on developing a model to predict student re-engagement (a student's voluntary re-participation in an educational intervention) using the data extracted from a Game-Based Learning called What-if Hypothetical Implementations in Minecraft (WHIMC), a game which offers engaging STEM learning experiences through Minecraft Java Edition. Quantifying re-engagement in environments like Minecraft can be considerably difficult because of its open-ended structure. In order to address this, the researcher harnessed machine learning methods to deliver a more advanced and thorough result, going beyond the traditional approaches of previous studies. The current research utilized data gathered from a group of Grade 8 students at a middle school in the Philippines who engaged with WHIMC. This collection of data includes student demographics, survey responses, and in-game features such as player positions, observations, and science tools usage. Leveraging the collected dataset, various machine learning classifiers, including Logistic Regression, Random Forest, Naïve Bayes, Support Vector Machine (SVM), and Multilayer Perceptron (MLP) algorithms, were evaluated using metrics such as ROC-AUC, Sensitivity, and Specificity. Notably, the MLP classifier, utilizing both survey and in-game data, exhibited the most optimal performance, yielding an ROC-AUC of .81, Sensitivity of 71%, and Specificity of 79%. Highlighting the significance of the Specificity metric, this study underscores its role in directing interventions towards students with a higher likelihood of disengagement.

KEYWORDS

Minecraft, whimc, machine learning, student re-engagement

1 INTRODUCTION

Re-engagement refers to a learner's spontaneous and self-motivated re-participation in an educational intervention. It is important as it serves as evidence of intrinsic motivation and intrinsic motivation leads students to demonstrate enhanced learning effectiveness [8, 9, 38]. This is one of the phases of the engagement process model [21]. This model is focused on deconstructing the concept of

engagement concerning individuals' experiences with technology. It is composed of four distinct phases: the point of initial engagement, a phase of sustained engagement, a subsequent disengagement phase, and, in some cases, a potential re-engagement phase.

Multiple studies indicate that when students engage in specific educational activities, they generally experience higher levels of satisfaction, better academic performance, persistence, and social engagement [2, 15, 16]. The level of student engagement can be impacted by various contextual factors, such as the specific learning environments they find themselves in or the instructional approaches employed by their teachers [6]. These external elements play a significant role in shaping how actively students participate in their educational experiences.

In parallel with this heightened interest on student engagement, the integration of digital technology has emerged as a central component of higher education, exerting its influence across every facet of the student experience [4, 12, 27]. A prominent example of the utilization of digital technology for learning is the integration of game-based learning (GBL). Implementing GBL offered students engaging learning experiences through interactive means. This approach granted students autonomy over tasks, amplifying their curiosity and involvement while preventing unfavorable emotional encounters [31].

This study focused on predicting student re-engagement using the data extracted from a GBL called What-if Hypothetical Implementations in Minecraft (WHIMC). WHIMC utilizes Minecraft Java Edition as an educational platform for students to actively delve into the scientific implications of alternate renditions of Earth via "what if" questions, such as "What if the Earth had no moon?" or "What if the Earth has a colder sun?" (As depicted in Figure 1). The project's primary goal is to engage, excite, and generate interest in STEM (Science, Technology, Engineering, and Mathematics) [35]. The data extracted from the game includes in-game data, student demographics, and answers to survey questions, which contain numerical, categorical, ordinal, and free-text data coming from the students who took the module.

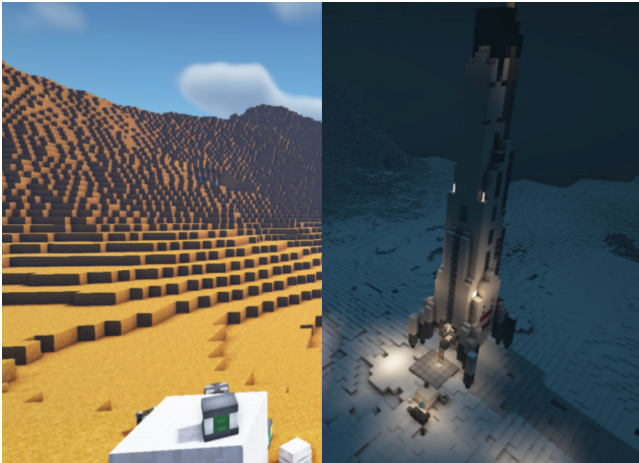


Figure 1: Earth with a Colder Sun in WHIMC. The world exhibits two distinct regions: the Western Area (left), characterized by a desert landscape, and the Eastern Area (right), covered in snow.

GBLs have proven to be beneficial for education across various fields, offering advantages such as fostering more genuine learning experiences and boosting student engagement. This is primarily due to their high level of interactivity and immersion [1]. In addition to this, the favorable impacts of GBLs encompass multiple areas such as student performance, skill acquisition, enthusiasm, and involvement. This motivated educators, game developers, funding entities, and researchers to employ games for instructing STEM subjects on various platforms [5]. Similar to these, WHIMC, as a GBL, shares the goal of engaging, nurturing enthusiasm, and sparking curiosity within the realm of science education [35]. As students immerse themselves in GBLs and other educational technologies like WHIMC, their learning potential expands. While re-engagement is not obligatory, it remains valuable in facilitating the learning opportunities and benefits mentioned in this chapter.

However, measuring re-engagement in environments like Minecraft can be considerably difficult because of its open-ended structure. For instance, the work of [37] tried to do an analysis on the impacts of WHIMC on student interest by making use of both quantitative and qualitative data such as coded interviews and notes, surveys on STEM interest, assessments of knowledge, and self-reported skill levels in Minecraft. On the other hand, the study by [9] attempted to quantify re-engagement by making use of additional in-game data on top of surveys, such as player positions, player observations and Science tools usage logs. Another study by [33] asked parents for help to check if students were playing Minecraft at home and to observe the students' level of concentration during their interaction with the game.

In the current research endeavor, the researcher attempted to quantify re-engagement, transcending the conventional approaches previously outlined and instead, utilizing machine learning methods, which have the potential to deliver a more sophisticated

and comprehensive outcome. Through this study, the researcher investigated the following: **RQ 1:** How can we develop a machine learning pipeline to effectively predict a student's likelihood of re-engaging with WHIMC? **RQ 2:** How effectively does the best-performing machine learning model predict re-engagement?

The rest of this paper is organized as follows. Included in the next section is the discussion of literature that provides background on the significance and potential benefits of utilizing machine learning methods to create a model that will be able to predict re-engagement using the survey and in-game data derived from WHIMC. This is then followed by a section that focuses on methods that the author used for data pre-processing, feature engineering, feature selection, modeling, validation, and post-modeling analysis. After, the results are shown by comparing the results of the various models. Finally, the author presents the conclusions and identify directions for future research.

2 REVIEW OF RELATED LITERATURE

To establish the groundwork for this research endeavor, the researcher conducted the literature review, thoroughly examining the existing body of work that employed machine learning techniques to quantify engagement across various platforms.

2.1. Existing studies about predicting engagement

Domina et al. [11] employed Machine Learning (ML) to predict student engagement during the COVID-19 pandemic, analyzing data from a survey of 10,000 parents in a southeastern U.S. school district. The resulting regression models highlighted the significance of technology accessibility: students with high-speed internet and devices showed greater engagement, even after socioeconomic factors were considered. Access to diverse learning opportunities also correlated with higher engagement levels. This discovery emphasizes that educators, despite their limited control over students' family backgrounds and household resources, retain the ability to make critical decisions regarding the instructional materials and resources they provide [11].

A different research conducted by Hussain et al. [13] employed machine learning algorithms to identify students with low engagement levels in a social science course from a virtual learning environment (VLE) at the Open University (OU) in the United Kingdom. To identify low-engagement students, the authors applied several types of ML algorithms to the dataset. They initially trained models using these algorithms and subsequently compared their accuracy, kappa values, and recall rates. The results indicated that the J48, decision tree, JRIP, and gradient-boosted classifiers demonstrated good performance in terms of accuracy, kappa value, and recall compared to other models. Based on these findings, the authors developed a dashboard to assist instructors at OU. These models can be seamlessly integrated into VLE systems, allowing instructors to assess student engagement across different activities

and materials and offer timely interventions to support students before their final exams. [13]

Another literature in line with the previously enumerated studies is the work by Ayouni et al. [3] that proposes an intelligent predictive system capable of forecasting students' engagement levels and providing them with feedback to enhance their motivation and commitment. A variety of data points such as number of logins, user participation, user activity, etc. were extracted from a Learning Management System (LMS). Based on these input features, the authors aim to create a predictor that could categorize students into three groups based on their engagement levels: "Not Engaged," "Passively Engaged," and "Actively Engaged." Three different machine learning algorithms, specifically the Decision Tree, Support Vector Machine, and Artificial Neural Network, were applied to analyze students' activities recorded in LMS reports. [3]

The findings of the study indicate that ML algorithms are effective in predicting students' engagement levels. Among these algorithms, the Artificial Neural Network displayed the highest accuracy rate at 85%, surpassing the Support Vector Machine (80%) and Decision Tree (75%) classification techniques. Based on these results, the intelligent predictive system delivers feedback to students and alerts instructors when a student's engagement level decreases. Instructors can then identify students' challenges within the course and motivate them through means such as email reminders, course messages, or scheduling online meetings. [3]

2.2 Attempts to quantify re-engagement

As of writing, there is only one study that aimed to quantify student re-engagement in the context of an open-world GBL. The study by Casano et al. [9] was initialized by gathering data through a survey to identify the most preferred features of WHIMC, followed by a systematic coding of these features to determine which aspects could be quantified and analyzed. Three potential re-engagement triggers were identified: social play, free exploration, and interactive learning components within WHIMC. The authors then used in-game data such as player position records, observations, and Science tool usage to develop simple heuristics to identify whether these triggers are exhibited outside designated testing hours. For instance, social play was defined as instances where player positions overlapped, indicating concurrent playtime outside of testing hours. The authors were able to utilize the module's design, as students had the option to access WHIMC beyond testing hours, enabling the analysis to consider these interactions as voluntary and unprompted. [9]

Furthermore, the study demonstrated ways to establish the extent to which specific elements may have contributed to re-engagement. For example, the degree of social play as a potential re-engagement trigger was described using the average number of concurrent users outside of testing hours. Free exploration as a re-engagement element was described by examining the ratio between the worlds

visited and the time spent exploring outside testing hours. Finally, the study illustrated how a creative parsing of available datasets enabled an understanding of the extent to which the interactive elements in WHIMC played a role in students' re-engagement with the game. [9]

3 METHODOLOGY

This section will highlight the methods employed by the researcher for data collection, data description, data cleaning, feature engineering, feature selection, modeling, and post-modeling analysis. It essentially outlines the approach and techniques used to address RQ1 and RQ2. Note that all these methodologies will be implemented using the Python programming language. Providing context for the collected data in the study, this section begins with details on navigating the WHIMC Server.

3.1 Navigating the WHIMC Server

WHIMC is a set of Minecraft worlds that aims to make learning STEM fun and engaging. These worlds are created and deployed in Minecraft Java Edition, and students can explore them to learn about different topics on STEM subjects. As the players progress in the game, they will have the opportunity to teleport to multiple worlds such as various versions of Earth which include normal baseline Earth, Earth with no moon, Earth with a different tilt, and Earth at a much cooler temperature. For instance, "Earth with No Moon" serves as an introductory experience for players as they begin exploring "what-if" scenarios. In this simplified alternate Earth, players gradually learn about various scientific concepts and tools, with a special emphasis on the observation tool, which is a crucial part of navigating the server. Completing this world equips players with a solid foundation for approaching other "what-if" scenarios more confidently. The objectives for this experience include gaining insights into how the absence of a moon impacts Earth's tides and winds, understanding the relationship between green energy sources and specific environmental conditions, and becoming proficient in using fundamental science variables and the `/observe` command—skills vital for success on the server. Similar to the "Earth with No Moon" world, every other world within the WHIMC server is thoughtfully designed to offer a diverse array of activities and valuable learning opportunities that cater to the players' educational and exploratory needs. [36]

3.2 Data Collection

The data used for this study came from a collective effort of Tablatin, Casano, and Rodrigo for previous studies focusing on WHIMC [9, 30]. The research team partnered with Philippine schools and teachers to develop learning modules targeting parts of the curriculum where WHIMC could be integrated. The data that was utilized for [30] was collected from the entire Grade 8 school population consisting of 8 class sections of a middle school in the Philippines. The students were tasked to work on two learning

modules, with ecosystem as the theme for Module 1 and biodiversity and evolution for Module 2. Prior to utilizing WHIMC, students completed a pre-test called Stem Interest Questionnaire (SIQ) to gauge their interest in various subject areas. Subsequently, following their engagement with WHIMC, the students participated in knowledge assessments, the Game Experience Questionnaire (GEQ), and the post-SIQ as a post-test evaluation. The respondents rate their level of agreement for both SIQ and GEQ using a 5-point Likert scale format (1 – *strongly disagree/not at all*, 2 – *disagree/slightly*, 3 – *neutral/moderately*, 4 – *agree/fairly*, 5 – *strongly agree/extremely*). After the activities mentioned, the researchers were able to come up with a collection of data which includes in-game features such as player positions, observations, and science tools used. In addition to this, student demographics and survey responses which includes their responses to the SIQ, GIQ, and open-ended questions were also compiled. In the current study, the researcher consolidated these data and performed the necessary steps enumerated in the next sections.

3.3 Consolidated Data Overview

In total, there were 211 prospective participants from the Grade 8 school population, representing 8 class sections of a middle school in the Philippines. After refining the data to include only those students who completed all the surveys and actively participated in WHIMC gameplay, 116 students, aged 13-15, remained. Among these 116 participants, a significant majority (93 out of 116 or 80%) reported being either familiar or very familiar with Minecraft. Another 20 students (17%) considered themselves somewhat familiar, while only 3 students (3%) admitted to having no prior familiarity.

A custom WHIMC plugin tracked player positions from a top-view perspective within the virtual worlds, capturing an average of 2,867 position counts per student. Additionally, each student generated an average of 10 observations and made use of science tools an average of 11 times during their gameplay.

3.4 Data Cleaning

It is important to acknowledge that the collected data may inherently contain noise, which can have a negative impact on the model's training. Hence, the process of data cleaning was carried out as deemed necessary [10]. In this sub-section, the researcher provides the methodology on how the data was refined, sanitized, and structured for the subsequent steps of the study.

3.4.1 Correcting Data Types

The initial data cleaning phase involved aligning data fields with appropriate types for compatibility with Python libraries. Utilizing functions from the Pandas Library [22], IDs across various dataframes were standardized to string format, ensuring seamless merging. Survey timestamps were transformed into datetime format, while in-game datasets with epoch timestamps were

converted to human-readable datetime values using a lambda function.

3.4.2 Handling Missing Values

Since the study is primarily focused on data derived from survey responses, instances (rows) representing students who were unable to complete all the surveys were filtered out across the datasets. On the other hand, the target variable for the model will be based on the instances of the students who will re-engage with WHIMC; therefore, students who did not actually participate in the game were excluded from the datasets. These steps were done through DataFrame indexing and boolean indexing in Pandas Library [22].

3.4.3 Data Standardization

Survey responses, primarily structured on the Likert Scale, required standardization due to varying interpretations. A mapping dictionary was created to flip values in columns with inverse responses. This ensured a consistent representation of the Likert scale, facilitating straightforward data analysis and interpretation.

3.4.4 Textual Data Clean-up

The survey responses were stripped of white spaces, unicode characters, and punctuation marks and lastly converted into lowercase. The Natural Language Toolkit (NLTK) [20] package was then used to remove English stop words.

The textual data is then normalized through lemmatization. Lemmatization is a method employed to reduce words to their base or root form, known as the lemma [14]. It utilizes a dictionary in its implementation. Put simply, it identifies the root word from various text variations. For instance, if the dataset contains words like “changes,” “changed,” and “changing,” all these words will be transformed to their root word, “change.” The Python implementation of this algorithm was accomplished by utilizing functions from NLTK [20].

3.5 Feature Engineering

The results of a closely related study [9] successfully identified three specific elements that emerged as potential drivers for re-engagement: social play, free exploration, and interactive learning elements within WHIMC. These findings were obtained through the systematic coding of in-game features, including player positions, observations, and Science tools usage. This could indicate that the relevant features for predicting student re-engagement may be found within the in-game features. To confirm this, the current study also focused on engineering features from the in-game dataframes.

3.5.1 Categorical Data Encoding

In this study, three techniques were utilized to convert categorical variables into numerical formats: One-Hot Encoding, Ordinal Encoding, and Binary Encoding. One-Hot Encoding was applied to variables without ordinal relationships, transforming them into binary variables indicating the presence (1) or absence (0) of specific categories. Ordinal Encoding preserved the order or ranking of categories, assigning numerical values based on their hierarchy. For example, "Familiarity" categories were encoded from 1 to 5. Binary Encoding captured the presence or absence of specific categorical values, setting binary indicators to 1 if a category was present and 0 otherwise.

3.5.2 In-game Data Aggregation

Aggregation techniques for feature engineering involve creating new features by grouping information from existing features within a dataset. These techniques are common concepts in descriptive statistics, used to capture higher-level patterns and information that can be valuable for machine learning models [24]. In the context of this study, the in-game data includes multiple rows for each participant. To create features that describe the details of the participant's gameplay, the researcher grouped the in-game data at the student level, utilizing standard aggregation functions such as mean, sum, count, minimum, and maximum.

3.6 Additional Features from Open-Ended Survey Responses

Analyzing linguistic data through computational methods is the essence of Natural Language Processing (NLP). Generally, the objective is to construct a representation of the text that imparts structure to the inherent unstructured nature of natural language [34]. NLP methods were employed on the survey responses to engineer features. Some key techniques utilized in this sub-section include lemmatization, bag-of-words, TF-IDF, and named entity recognition. The methods described herein focus specifically on the free-text survey responses to the questions: "*What aspects of WHIMC contributed to finding the topic enjoyable, interesting, and/or easy to learn?*" and "*What aspects of WHIMC contributed to finding the topic uninteresting and/or challenging to learn?*"

3.6.1 Bag-of-words

Bag-of-words model is a straightforward and predominantly used method to extract features for machine learning models. This technique is easily adjustable and can be used in a lot of ways to extract features from texts. In this model, the collection of each text data instance will be represented as the bag (multiset) of its words. The grammar and word order will no longer be considered, but the frequency of the words will still be recorded [23]. Specifically, a bag-of-words is a representation of textual data that describes the occurrence of words within a dataset. It only has two components:

the dictionary of all the unique words existing in the dataset which will be transformed into a column, and the frequency of each of those words row-wise. For the python implementation of Bag-of-words, a vectorizer function from the scikit-learn library [26] was utilized.

3.6.2 TF-IDF

TF-IDF, short for Term Frequency-Inverse Document Frequency, is another method used for extracting features from textual data. It consists of two components: TF and IDF [25]. Term Frequency (TF) measures how frequently a word appears in a document relative to the total number of words in that document. In the context of this study, a document corresponds to a survey response. The TF value is calculated by dividing the number of times a particular word appears in a document by the total number of words in that document which implies that the TF value increases as a word occurs more within a document. Conversely, Inverse Document Frequency (IDF) computes the importance of rare words in the dataset. This value is calculated for each word by taking the logarithm of the ratio between the total number of documents and the number of documents containing that word. This implies that scarcer words have higher IDF values. The final TF-IDF value is obtained by multiplying TF by IDF. The computational process for TF-IDF was accomplished in Python using a vectorizer function from the scikit-learn library [26].

3.6.3 Named Entity Recognition

Named Entity Recognition (NER) is the method of identifying nouns and proper nouns in a given text and categorizing them into predefined categories such as names of people, organizations, locations, dates, percentages, and more [19]. These categories function as additional features in the dataset, with the feature values denoting the frequency of each category within every survey response. Implementing NER in Python was achieved using the spaCy library [29].

3.7 Feature Selection

The researcher utilized Recursive Feature Selection (RFE) to automatically select a subset of the most relevant features from a given dataset. RFE works by recursively fitting the model to the data with different subsets of features and ranking them based on their importance, typically through a model-specific metric like coefficients in a logistic regression model or feature importances in a tree-based model [28].

3.8 Model Training and Evaluation

To determine which classifier should be used to predict a student's likelihood of re-engaging, the researcher applied five different classification models to the resulting dataset from the previous steps. These models include Logistic Regression, Random Forest,

Naïve Bayes, Support Vector Machine (SVM), and Multilayer Perceptron (MLP) [7]. Each of these models employs its unique approach to analyze the data and generate predictions for the target variable based on the given input variables. The performance of these models was then compared in terms of Sensitivity, Specificity, and ROC-AUC. Sensitivity measures the model's ability to correctly identify positive instances among all actual positives in the dataset, while Specificity assesses the model's ability to correctly identify negative instances among all actual negatives. The ROC curve visually represents a classifier's performance across different sensitivity and specificity thresholds. While it doesn't provide a single performance measure, the Area Under the Curve (AUC) condenses overall performance into a single metric. AUC values range from 0.5 to 1.0, with higher values indicating better performance [7, 17, 18].

4 RESULTS AND DISCUSSION

After completing the initial steps of the machine learning pipeline, including Data Collection, Data Cleaning, and Feature Engineering, a corpus of features ready to be trained by the selected models was produced. The dataset comprises 116 rows representing students and 176 features derived from both the Survey data (excluding free text) and in-game data. Additionally, 115 features originated from the free-text survey column, specifically from responses to the question "What aspects of WHIMC contributed to finding the topic enjoyable, interesting, and/or easy to learn?". The combination of these features then underwent Recursive Feature Elimination (RFE) before each round of modeling experiments, as detailed in the subsequent sections of this chapter.

The researcher aimed to predict if the student will re-engage with WHIMC or not. The target variable is binary and is defined as follows:

Table 1. Target Variable Definition

Class	Definition
1	The participant played WHIMC again the week after the 2-day training period.
0	The participant did not play WHIMC again after the 2-day training period.

In creating the models to predict student re-engagement, the researcher decided on specific sets of features to be used, as outlined below:

Table 2. Sets of Features Used in Prediction Models

F1 – Survey data (without the free text data) and in-game data (positions, observations, and science tools usage).
F2 – Survey data, features from the free text data, and in-game data (positions, observations, and science tools usage).
F3 – Features from the free text data only.

For each set of features above, 5 types of machine learning classifiers were trained on the data of 69 students, then tested on 47 hold out instances. Specifically, the machine learning classifiers used in this study were: (1) Logistic Regression, (2) Random Forest, (3) Naive Bayes, (4) Support Vector Machine (SVM), and (5) Multilayer Perceptron (MLP). To ensure optimal model performance across both classes, the metrics of Sensitivity, Specificity, and ROC-AUC were employed for evaluation.

The modelling experiments involved applying the chosen classifiers to each of the set of features as enumerated above. In total there were 15 classifiers made and assessed for this section. The results below show the metrics of the top 5 models ordered by ROC-AUC.

Table 3. Performance of Top 5 Classifiers

Model	Features	ROC-AUC
MLP	F1	.81
Logistic Regression	F1	.76
Naïve Bayes	F2	.71
SVM	F1	.68
Naïve Bayes	F3	.66

The most effective model for classifying student re-engagement in WHIMC was identified as the MLP classifier, employing survey data (without the free text data) and in-game data (positions, observations, and science tools usage) as features (F1). MLP is a type of artificial neural network capable of learning complex non-linear relationships in the data [32]. This is particularly useful when dealing with intricate patterns and interactions among features, which may be present in the dataset related to student re-engagement. Notably, for the best performing models, most of the features that were utilized are F1, alluding to its importance in predicting the outcome.

The confusion matrix below provides a detailed breakdown of the classification outcomes for the best MLP classifier in predicting student re-engagement:

Table 4. Confusion Matrix of the top performing MLP Classifier

		Predicted Label	
		0	1
Actual Label	0	11	3
	1	10	23

This model exhibited exemplary performance with an ROC-AUC of 0.81, sensitivity of 70%, and specificity of 79%. The choice of the MLP classifier was substantiated by its performance in discerning both positive and negative instances, achieving a balance between identifying students who would re-engage (sensitivity) and those who would not (specificity).

In particular, the emphasis on Specificity is crucial in this study, as this indicates that the model could be beneficial in providing early

intervention to students who are less likely to re-engage. Achieving a higher Specificity ensures a better ability to correctly identify true negatives, i.e., students who will not re-engage with WHIMC. This holds significance for facilitators, enabling them to target interventions toward students at a higher risk of disengagement. The consideration of Specificity in the model's performance underscores its practical application in education, facilitating proactive measures for students requiring additional support or encouragement. For instance, akin to the study by Hussain et al. [13], the model could be integrated into a system or deployed as a tool, allowing facilitators to monitor student engagement and offer timely interventions.

On the other hand, the results of the modeling experiments show that the addition of features extracted from the free text data for feature sets F2 and F3 did not improve the model performance. This could indicate that the quality and relevance of the textual features may not align with the patterns in the dataset related to student re-engagement. As a recommendation, exploring different techniques, such as more advanced NLP methods, could enhance the effectiveness of utilizing textual features. Additionally, incorporating more data, such as information from other schools implementing the WHIMC module, could enhance the model's ability to generalize and improve overall performance.

5 CONCLUSION

Re-engagement is defined as the learner's voluntary re-participation in an educational intervention. It is important as it is an indicator of intrinsic motivation which leads students to demonstrate enhanced learning effectiveness [8, 9, 38]. In this study, the specific challenge lies in developing a classifier that effectively gauges and distinguishes instances of student re-engagement by utilizing the data from a GBL called What-if Hypothetical Implementations in Minecraft (WHIMC). The researcher addressed this problem by investigating the following: **RQ 1:** How can we develop a machine learning pipeline to effectively predict a student's likelihood of re-engaging with WHIMC? **RQ 2:** How effectively does the best-performing machine learning model predict re-engagement?

In order to approach RQ1, the researcher provided a comprehensive methodology that details the approach and techniques to develop a machine learning classifier for student re-engagement. The data used for the study came from the module implementation of WHIMC in a middle school in the Philippines composed of a batch of Grade 8 students. Extracted data included in-game features, student demographics, and survey responses. To refine the data, the researcher corrected data types, handled missing values, standardized scales, and cleaned textual data. Feature engineering involved categorical data encoding, aggregation, and textual feature engineering using NLP methods such as bag-of-words, Named Entity Recognition (NER), and TF-IDF. Recursive Feature Elimination (RFE) was then applied to reduce dimensionality, preparing the data for model training. Various machine learning

classifiers, including Logistic Regression, Random Forest, Naïve Bayes, Support Vector Machine (SVM), and Multilayer Perceptron (MLP) algorithms, were evaluated using ROC-AUC, Sensitivity, and Specificity.

In addressing RQ2, the researcher aimed to assess the efficacy of the models in predicting student re-engagement. This involved analyzing the relevant metrics from the modeling experiments to identify the best performing model. The researcher carefully selected specific sets of features for utilization. These feature sets, denoted as F1, F2, and F3, encompass different combinations of survey data, free text data, and in-game data, each tailored to capture distinct aspects. For each feature set, five types of machine learning classifiers were trained using the data from 69 students and subsequently tested them on 47 holdout instances to assess their predictive performance. The most effective model for classifying student re-engagement in WHIMC was identified as the MLP classifier, employing survey data (without the free text data) and in-game data (positions, observations, and science tools usage) as features (F1). This model exhibited exemplary performance with an ROC-AUC of 0.81, sensitivity of 70%, and specificity of 79%. The metrics of the resulting model highlight it has good performance in identifying true negatives, i.e., students who will not re-engage with WHIMC. This is important as this enables targeted interventions toward students at a higher risk of disengagement.

For future improvements on the study, the recommendation is to explore advanced NLP methods to enhance the effectiveness of using textual features. Additionally, incorporating more data, especially from other schools implementing the WHIMC module, could improve the model's generalization and overall performance.

REFERENCES

- [1] Alonso-Fernandez, C., Calvo-Morata, A., Freire, M., Martinez-Ortiz, I. and Fernandez-Majón, B. 2019. Applications of data science to game learning analytics data: a systematic literature review. *Computer Education*. 141, 103612–103619.
- [2] Astin, A.W. 1984. Student involvement: A developmental theory for higher education. *Journal of College Student Development*. 25, 297–308.
- [3] Ayouni, S., Hajje, F., Maddeh, M. and Al-Otaibi, S. 2021. A new ML-based approach to enhance student engagement in online environment. *PLoS ONE*. 16, (11) Nov. 2021, e0258788. DOI:<https://doi.org/10.1371/journal.pone.0258788>.
- [4] Barak, M. 2018. Are digital natives open to change? Examining flexible thinking and resistance to change. *Computers & Education*. 121, 115–123.
- [5] Bertozzi, E. 2014. Using Games to Teach, Practice and Encourage Interest in STEM Subjects. *Learning, Education and Games*. 23–36.
- [6] Bond, M. and Bedenlier, S. 2019. Facilitating student engagement through educational technology: Towards a conceptual framework. *Journal of Interactive Media in Education*. 1, (11) 2019, 1–14.

- [7] Bruce, P. and Bruce, A. 2017. *Practical Statistics for Data Scientists: 50 Essential Concepts*. O'Reilly Media, Inc.
- [8] Cairns, P. 2016. Engagement in digital games. In *Why engagement matters.*, 81–104.
- [9] Casano, J.D.L., Fuentes, M. and Rodrigo, M.M.T. 2023. Quantifying Re-engagement in Minecraft. *Artificial Intelligence in Education*. 1831.
- [10] Chu, X., Ilyas, I.F., Krishnan, S. and Wang, J. 2016. Data Cleaning: Overview and Emerging Challenges. *Proceedings of the 2016 International Conference on Management of Data (New York, NY, USA, Jun. 2016)*, 2201–2206.
- [11] Domina, T., Renzulli, L., Murray, B., Garza, A.N. and Perez, L. 2021. Remote or removed: Predicting successful engagement with online learning during COVID-19. *Socius: Sociological Research for a Dynamic World*. 7.
- [12] Henderson, M., Selwyn, N. and Aston, R. 2017. What works and why? Student perceptions of 'useful' digital technology in university teaching and learning. *Studies in Higher Education*. 42, (8) 2017, 1567–1579.
- [13] Hussain, M., Zhu, W., Zhang, W. and Abidi, S.M. 2018. Student engagement predictions in an e-learning system and their impact on student course assessment scores. *Computational Intelligence and Neuroscience*. 1–21.
- [14] Khyani, D. and B S, S. 2021. An Interpretation of Lemmatization and Stemming in Natural Language Processing. *Shanghai Ligong Daxue Xuebao/Journal of University of Shanghai for Science and Technology*. 22, 350–357.
- [15] Kuh, G.D. 2001. Assessing what really matters to student learning: Inside the national survey of student engagement. *Change*. 33, (3) 2001, 10–17.
- [16] Kuh, G.D., Cruce, T.M., Shoup, R., Kinzie, J. and Gonyea, R.M. 2008. Unmasking the effects of student engagement on first-year college grades and persistence. *Journal of Higher Education*. 79, (5) 2008, 540–563.
- [17] Melo, F. 2013. Area under the ROC Curve. *Encyclopedia of Systems Biology*. W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, eds. Springer. 38–39.
- [18] Melo, F. 2013. Receiver Operating Characteristic (ROC) Curve. *Encyclopedia of Systems Biology*. W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, eds. Springer. 1818–1823.
- [19] Mohit, B. 2014. Named Entity Recognition. *Natural Language Processing of Semitic Languages*. I. Zitouni, ed. Springer. 221–245.
- [20] NLTK :: Natural Language Toolkit. Retrieved November 10, 2023 from <https://www.nltk.org/>.
- [21] O'Brien, H. and Toms, E.G. 2008. What is user engagement? A conceptual framework for defining user engagement with technology. *Journal of the American Society for Information Science and Technology*. 59, (6) 2008, 938–955.
- [22] pandas documentation — pandas 2.1.1 documentation. Retrieved October 26, 2023 from <https://pandas.pydata.org/docs/>.
- [23] Qader, W., M. Ameen, M. and Ahmed, B. 2019. An Overview of Bag of Words; Importance, Implementation, Applications, and Challenges. (Jun. 2019), 200–204.
- [24] Ross, A. and Willson, V.L. 2017. *Descriptive Statistics. Basic and Advanced Statistical Tests: Writing Results Sections and Creating Tables and Figures*. A. Ross and V.L. Willson, eds. SensePublishers. 3–7.
- [25] Sammut, C. and Webb, G.I. eds. 2010. TF-IDF. *Encyclopedia of Machine Learning*. Springer US. 986–987.
- [26] scikit-learn: machine learning in Python — scikit-learn 1.3.2 documentation. Retrieved November 10, 2023 from <https://scikit-learn.org/stable/>.
- [27] Selwyn, N. 2016. Exploring university students' negative engagements with digital technology. *Teaching in Higher Education*. 21, (8) 2016, 1006–1021.
- [28] sklearn.feature_selection.RFE. Retrieved November 6, 2023 from https://scikit-learn/stable/modules/generated/sklearn.feature_selection.RFE.html.
- [29] spaCy · Industrial-strength Natural Language Processing in Python. Retrieved November 10, 2023 from <https://spacy.io/>.
- [30] Tablatin, C.L.S., Casano, J.D.L. and Rodrigo, M.M.T. 2023. Using Minecraft to cultivate student interest in STEM. *Frontiers in Education*. 8.
- [31] Taub, M., Sawyer, R., Smith, A., Rowe, J., Azevedo, R. and Lester, J. 2020. The agency effect: The impact of student agency on learning, emotions, and problem-solving behaviors in a game-based learning environment. *Computers & Education*. 147.
- [32] Taud, H. and Mas, J.F. 2018. Multilayer Perceptron (MLP). *Geomatic Approaches for Modeling Land Change Scenarios*. M.T. Camacho Olmedo, M. Paegelow, J.-F. Mas, and F. Escobar, eds. Springer International Publishing. 451–455.
- [33] Tromba, P. 2013. Build Engagement and Knowledge One Block at a Time with Minecraft. *Learning & Leading with Technology*. 40, (8) 2013, 20–23.
- [34] Verspoor, K. and Cohen, K.B. 2013. Natural Language Processing. *Encyclopedia of Systems Biology*. W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, eds. Springer. 1495–1498.
- [35] What-if hypothetical implementations in minecraft. from <https://whimcproject.web.illinois.edu/>.
- [36] WHIMC Teacher Guides. Retrieved October 24, 2023 from <https://whimcproject.web.illinois.edu/education-research/teacherguide/>.
- [37] Yi, S. 2021. The impacts of a science-based videogame intervention on interest in STEM for adolescent learners.
- [38] Zaccone, M.C. and Pedrini, M. 2019. The effects of intrinsic and extrinsic motivation on students learning effectiveness. Exploring the moderating role of gender. *International Journal of Educational Management*. 33, (6) 2019, 1381–1394.

EXPLORING PHOTOGRAMMETRY AND NEURAL RADIANCE FIELDS FOR PRESERVING BAGUIO STATUES AND MONUMENTS

Yohann C. Gayao

College of Information in Technology
and Computer Science
University of the Cordilleras

Governor Pack Road Upper Session
ycg4409@students.uc-bcf.edu.ph

Bill R. Malitao

College of Information in Technology
and Computer Science
University of the Cordilleras

Governor Pack Road Upper Session
brm1653@students.uc-bcf.edu.ph

Christella Marie R. Ocampo

College of Information in Technology
and Computer Science
University of the Cordilleras

Governor Pack Road Upper Session
cro4664@students.uc-bcf.edu.ph

ABSTRACT

The Cordillera Administrative Region in the Philippines has a rich cultural heritage, including cultural statues and monuments. However, these artifacts are facing threats due to deterioration and inadequate preservation measures. This study aims to address this concern by comparing Photogrammetry and NeRF; two advanced digitization techniques, to accurately preserve digitally these cultural artifacts. However, more data is yet to be found regarding the comparative efficiency of these techniques in handling detailed objects like cultural artifacts. Therefore, this research aims to conduct a quantitative comparative analysis of Photogrammetry and NeRF in terms of accuracy, processing time, and resource requirements. The accuracy of the generated 3D models has been assessed by comparing them to the physical artifacts using established evaluation metrics. Processing time and resource requirements, such as computational power, have been measured as well to evaluate the efficiency of each technique. By comparing the performance of Photogrammetry and NeRF, this study aims to identify the most efficient approach for the digital preservation of CAR's cultural heritage. The researchers' hypothesis posits that while Photogrammetry might excel in ideal settings, its cost intensity could be a bottleneck. Conversely, NeRF's versatility allows it to perform well when faced with limitations while remaining cost-effective, making it a good alternative. By identifying the optimal approach for digitizing the Cordillera's cultural treasures, the researchers pave the way for developing effective preservation strategies.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Categories and Subject Descriptors

[Digital Preservation]: 3D Model Reconstruction - *geometric accuracy, ground truth, plane fitting assessment, cloud-to-cloud comparison*,
[Artificial Intelligence]:

General Terms

Machine Learning, Deep Learning,

Keywords

Photogrammetry; Neural Radiance Fields; Splatfacto; Instant-NGP; Nerfacto

1. INTRODUCTION

One of the rising concerns of our society today is cultural preservation. Cultural preservation pertains to the restoration and conservation of artifacts and historical sites. These include paintings, sculptures, documents, and locations with historical significance. Cultural preservation is a challenging duty as there are numerous issues; financial constraints, climate change, and natural disasters that can cause irreparable damage to cultural treasures. Cultural heritage sites are legacies that must be preserved.

As it is known, Numerous conservation efforts include restoring historical buildings, passing on an ancient craft, or recording traditional tales. However, These techniques are antiquated and are not impervious to the passage of time as the artifacts and landmarks are still subjected to natural elements. Factors such as deterioration, climate change, and the challenges of limited resources require innovative solutions to protect these heritage sites continuously [1].

Storing them digitally through modern techniques is starting to become a reliable addition to cultural preservation efforts [2]. Numerous cultural institutions have also begun to digitize their collections to make them more accessible and protected. Traditional methods are proven inadequate when utilized alone as they do not address the main issue, which is the inevitability of any physical artifact's eventual deterioration. AI algorithms play a crucial role in identifying damage and formulating targeted restoration plans for deteriorated artifacts and structures [1].

Applying both traditional and digital methods together is preferable to address deterioration concerns.

By leveraging principles of image analysis and computational geometry, photogrammetry plays a vital role in transforming visual data into valuable three-dimensional representations, providing crucial insights [3]. Photogrammetry encompasses methods of image measurement and interpretation to derive the shape and location of an object based on the dataset of gathered photographs [4]. As of now, innovations in the field of photogrammetry have made it much more flexible in its application and present new opportunities in the representation of monuments [5].

Neural Radiance Fields, on the other hand, represent a powerful framework that excels in capturing complex lighting effects, intricate details, and realistic textures, making it a pivotal technology in the realm of computer-generated imagery [6]. However, each scene must be optimized individually, with no knowledge shared between the images. Unnecessarily time-intensive and in the limit of single or extremely sparse views, it is unable to make use of any prior knowledge to speed up the process of model creation [7]

accommodating dynamic elements. The neural network's ability to capture complex, non-linear relationships in the data contributes to generating visually compelling 3D reconstructions, showcasing the symbiotic relationship between photogrammetry, neural radiance fields, and the transformative potential of machine learning in computer vision applications.

However, the efficiency and processing speed of the methods are yet to be tested and compared. Certain metrics such as the accuracy and image clarity in regards to exposure to different kinds of lighting have not been tested on cultural statues and monuments and in the Cordilleran region specifically.

The researchers aim to find out how the emerging capabilities of Neural Radiance Fields compare to the established strengths of photogrammetry. Through meticulous data gathering and comparison of NeRF and photogrammetry 3D models, relevant data regarding their respective outputs have been compared to one another in terms of efficiency, speed, and overall image quality.

The results have been utilized to contribute meaningfully to the ongoing dialogue surrounding digital preservation, fostering advancements in the field and supporting informed

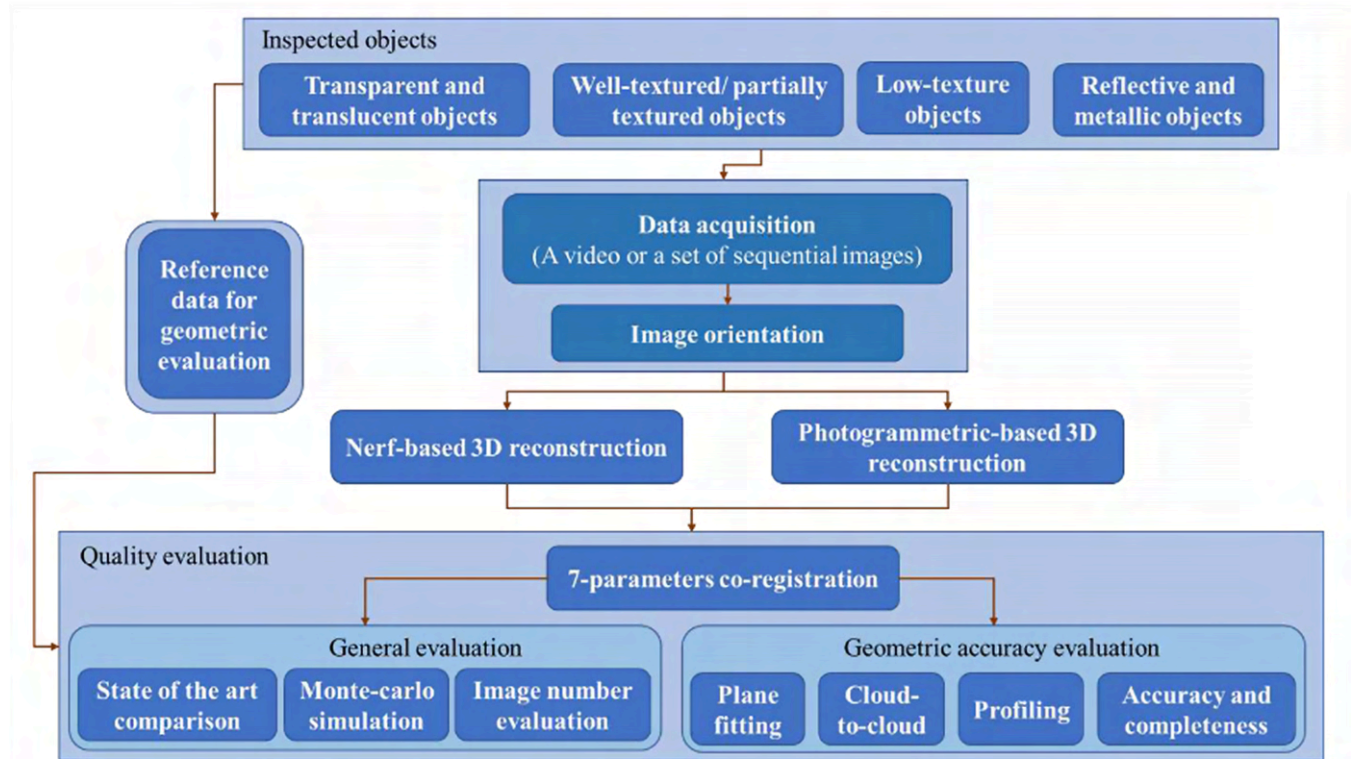


Figure 1. Overview of the proposed procedure to assess the performance of NeRF-based 3D reconstruction with respect to conventional photogrammetry [4]. Excerpt from A Critical Analysis of NeRF-based 3D Reconstruction. Remote Sensing.

The utilization of machine learning within Neural Radiance Fields is instrumental in achieving their remarkable capabilities. These networks are trained on extensive datasets containing diverse scenes and lighting conditions, enabling them to generalize and infer intricate 3D structures from 2D images. During the training process, the neural networks learn to predict the radiance or color of a point in 3D space based on its position and appearance in the input images. This learned representation allows NeRF to generate novel views of a scene with unprecedented realism, even

decision-making for the conservation of cultural artifacts in urban settings.

2. METHODOLOGY

This study employs a methodical and experimental quantitative design to explore digital preservation methods for statues and monuments in Baguio City. The process begins with a well-defined research question, focusing on evaluating the current methods or environmental impact on deterioration. Following a

thorough literature review, a suitable study design is chosen. Methodical studies involve interviews and observations, while experimental setups manipulate variables and utilize control groups to test preservation techniques. Representative statues are then selected for sampling.

Figure 1 employs a three-phased methodological framework. Initially, meticulous textural assessment of the objects is conducted to establish a benchmark for subsequent geometric evaluation. This data serves as a crucial reference point for the later stages. Subsequently, a set of sequential images is acquired and utilized as datasets for both NeRF-based and photogrammetric-based 3D reconstruction techniques. Finally, a comprehensive quality evaluation is performed, encompassing general and geometric accuracy assessments. Notably, the geometric accuracy evaluation leverages reference data as ground truth to ensure the validity and reliability of comparisons.

3. RESULTS AND DISCUSSIONS

Our study explored photogrammetry and Neural Radiance Fields for preserving Baguio statues and monuments. The researchers hypothesized that photogrammetry performs better in an ideal setting, but its cost intensity is an issue, and that NeRF is a viable alternative. Our findings partially supported this hypothesis. Models of NeRF showed similar quality to the models of photogrammetry while using fewer images.

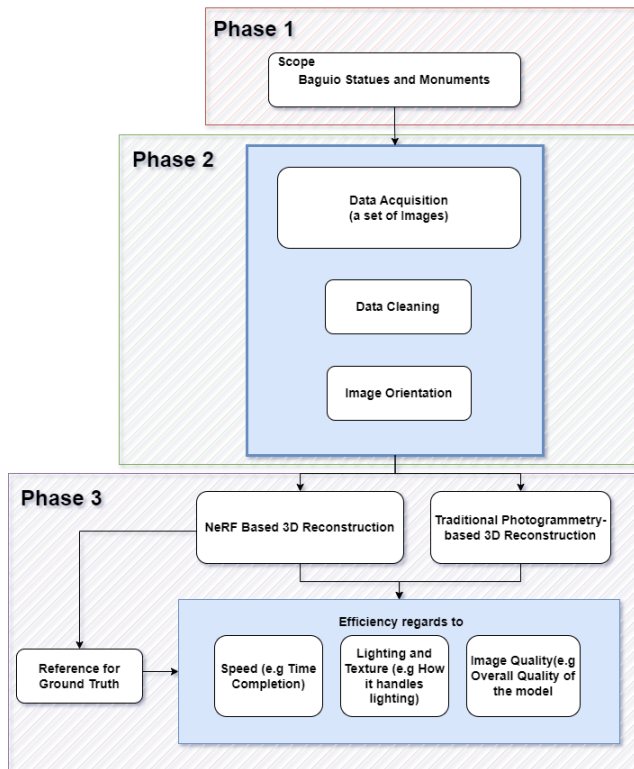


Figure 2. Model to compare NeRF to Traditional Photogrammetry in regards to efficiency, speed, lighting, and image quality

Figure 2 was constructed in reference to Figure 1. Displaying a set of phases and procedures to be conducted to properly assess and compare photogrammetry and Neural Radiance Fields digital preservation techniques. The model also employs a three-phased

methodological framework. Respectively, documentation, data acquisition, and 3D model reconstruction.



Figure 3. Statue Images taken to be used as Datasets

Phase 1, in Figure 3, the statues and monuments in Baguio were found and documented. Each statue or monument is classified based on the object’s texture/material used (e.g. metal, cement, wood, glass, or mixed). Detailed documentation included notes on material composition, and surrounding lighting conditions at multiple times of the day. A table containing the statues’ primary data can be observed below.

Name	Lighting	Material	Surface Finish
Crucifix Statue	Harsh Directional Lighting	Painted	Smooth
Manuel L. Quezon Statue	Diffused Lighting	Painted Concrete	Rough
Centennial Marker	Harsh Directional Lighting	Marble	Smooth
Burnham Bust	Harsh Directional Lighting	Painted Concrete/Marble	Smooth
Emilio Aguinaldo Monument	Diffused Lighting	Concrete	Rough

Figure 4. Table of Classification of the Statues

It is shown in Figure 4 that three of the statues were documented with harsh directional lighting due to sunlight, while the other two were documented during a calmer clime. The materials of the statues were predominantly concrete, or painted concrete, while

one statue was made from stone. Regarding the surface finish, on the other hand, only 2 statues were smooth while the rest were rough. These listed attributes were taken into consideration in the evaluation of the outputted 3D models.

For data acquisition, the researchers made sure to employ certain image capture guidelines. Particularly, image overlapping, taking more photos than required, effective use of the frame, good lighting, and avoiding blind spots. These guidelines helped ensure that the quality and formatting of the images were uniform.

However, since the researchers were only able to use phones for data collecting, consequently, a few issues were encountered. Specifically, limited storage life and battery capacity, lower resolution, and lack of configuration options for the camera.

Furthermore, the researchers meticulously documented the hardware specifications, particularly those relevant to photogrammetry and NeRF dataset processing. The chosen hardware plays a critical role in evaluating the comparative performance between these two techniques, as both are inherently dependent on hardware capabilities and input data. Due to limitations imposed by a mid-tier GPU with only 8GB of VRAM, the researchers implemented a pre-processing step for the photo dataset to reduce VRAM consumption. This pre-processing also facilitates a more efficient rendering process within the photogrammetry and NeRF software, focusing computational resources solely on the object of interest.



Figure 5. Creation of the Dataset for the Statues

Then, in Phase 2, Figure 5, the researchers proceeded with the creation of datasets to push the boundaries of 3D reconstruction using Neural Radiance Fields (NeRF). The researchers chose cultural artifacts like statues and monuments, considering their size, intricate shapes, and diverse materials. To make things even more in-depth, the statues and monuments were photographed under different lighting conditions.



Figure 6. Images masked for use

As seen in Figure 6, the images for the dataset have undergone data cleaning, and unnecessary details such as their backgrounds have been covered. The researchers manually masked the individual images through the use of Adobe Photoshop 2024 and its mask layering feature. This process was used to hide parts of the images without deletion. The images were then exported as PNGs to maintain the mask layering.



Figure 7. The Photogrammetric Reconstruction Process of the Statues

After gathering and cleaning images for the dataset, the researchers proceeded with Phase 3. In Figures 7 and 8, a comparative reconstruction of the statues and monuments is shown. In this phase, the researchers used the constructed datasets to apply Nerfacto which is an implementation of NerfStudio. Alongside this process, photogrammetric models were reconstructed using Agisoft Metashape. The evaluation focused on reconstruction speed, accuracy (both large-scale shape fidelity and detail preservation), and the influence of lighting variations.



Figure 8. NeRF Reconstruction Process of the Statues

This phase delves into the granular analysis and optimization of the chosen NeRF-based reconstruction workflow. A battery of metrics, encompassing error calculations (e.g. STD, Mean_E, RMSE, MAE), cloud-to-cloud comparisons, and performance profiling have been employed to rigorously quantify the strengths and weaknesses of the used NeRF approach.

Metrics comparing the results of the Neural Radiance Fields method against ground truth photographs guided our refinement

Burnham Statue	769	1215
Manuel L, Quezon Bust	347	1022
Centennial Marker	330	990
Crucifix Statue	327	1064
MEAN	423	1065.4

Figure 10. Table of Total Time Processed

In Figure 10, Nerfacto took a considerably longer amount of time compared to its more traditional counterpart, photogrammetry. With a mean of “1065.4”, larger than twice the photogrammetry’s mean of “423”. The researchers hypothesize that NeRF’s utilization of machine learning has caused this difference in processing time.

Through proper surveying, and the compilation of relevant images with varying viewpoints, lighting, and time of the day taken; the researchers have managed to create a dataset that satisfies the requirements of NeRF and photogrammetry, allowing the



Image

Photogrammetry

Nerfacto

Figure 9. Comparative Images between Captured Image, Photogrammetry, and NeRF

process. Our analysis also compared results from advanced mesh optimization features within Agisoft Metashape.

Statue / Monument	Photogrammetry Time(Seconds)	Nerfacto Time (Seconds)
Emilio Aguinaldo Statue	342	1036

researchers to properly utilize variants of both digital preservation techniques to recreate the statues to an adequate degree. Through the utilization of both the models that were produced by NeRF and photogrammetry techniques, the researchers performed a comparative reconstruction where the models were judged with set metrics and compared to the ground truth image of the statues. The metrics were measured via numerous error calculations, to determine its efficiency in terms of speed, lighting, and image quality.

Notably, photogrammetry required a significantly larger image set, suggesting its inherent labor-intensiveness [8] and the diminishing final product accuracy when analyzing an object from afar without multiple control ground points [9]. While NeRF achieved model generation with fewer images, its quality suffered accordingly, highlighting its dependence on a substantial dataset as well for accuracy and fidelity. Also, NeRF models its input using infinitesimally small 3D points along a ray, which causes aliasing when rendering views of varying resolutions [10].

In Phase 3, Figure 5, the researchers were able to maximize the capabilities of both NeRF and photogrammetry and their techniques within hardware constraints. This involved reprocessing incomplete or inaccurate models until obtaining the optimal outcome.

While quantitative results are pending, the study's findings temporarily position NeRF as a viable alternative to photogrammetry, albeit not a direct replacement. Both techniques demonstrate unique strengths and weaknesses, rendering their suitability context-dependent.

This research study centered on a comparative analysis of the efficiency of Neural Radiance Fields-based 3D reconstruction techniques versus established photogrammetry methods for the digital preservation of cultural heritage objects. Specifically, for the scope, the focus is on statues and monuments located within Baguio City. Only Nerfacto, an implementation of NeRFStudio, and established photogrammetry methods have been evaluated to pinpoint the most appropriate approach for accurately capturing these heritage assets. The researchers employed quantitative metrics. These include error calculation, cloud-to-cloud comparison, and profiling in assessing geometric accuracy and fidelity. This study prioritized object materials (stone, metal, wood), and varying lighting conditions, alongside analysis of factors like rendering time and image texture quality.

4. CONCLUSIONS AND RECOMMENDATIONS

This study lays the groundwork for further exploration of NeRF and photogrammetry in digital preservation. To advance the understanding and utilization of these techniques, several avenues for future research can be pursued. The digital landscape is actively advancing, and both NeRF and photogrammetry are no exceptions. It is crucial to delve into recent and versatile variants of these methodologies. Evaluating their strengths and limitations in the context of cultural heritage preservation can offer valuable insights. Another way is by developing robust comparative frameworks that go beyond basic error calculations, for it is essential. Metrics should encompass factors such as geometric accuracy, material authenticity, textural fidelity, and visual realism. Ground truth data obtained from high-precision scanning technologies should be used for comparative evaluation, ensuring the reliability and objectivity of comparisons. Establishing standardized benchmarks and data sets would enable researchers to share and compare their findings more effectively.

Both techniques show ample potential in fields other than cultural preservation. 3D model reconstruction in architecture, gaming, filmmaking, and even forensics where a crime scene can be reconstructed in 3D to offer a different and perhaps more holistic point-of-view. These examples are only a few of the many directions that could be taken when utilizing photogrammetry and NeRF, highlighting their versatility even more.

Lastly, while NeRF and photogrammetry are two distinct techniques, exploring the possibility of integration or the creation of a hybrid may prove beneficial to the field of study.

5. ACKNOWLEDGEMENTS

To the panel members, for sharing their suggestions and remarks that have greatly helped us to develop the project. To our adviser Miss Lovely Jen Reformado, Ma'am Thelma Domingo-Palaoag, and the members of CITCS, for enthusiastically viewing and testing the project and providing feedback to the researchers. More importantly, the researchers are grateful for their friends and family who gave them wholehearted guidance, moral support, and encouragement that kept them going.

6. REFERENCES

- [1] Akyol, Gamze & Avci, Ali Berkay. (2023). AI APPLICATIONS IN CULTURAL HERITAGE PRESERVATION: TECHNOLOGICAL ADVANCEMENTS FOR THE CONSERVATION. 10.6084/m9.figshare.24077862.
- [2] Murtiyoso, A., & Grussenmeyer, P. (2023). INITIAL ASSESSMENT ON THE USE OF STATE-OF-THE-ART NERF NEURAL NETWORK 3D RECONSTRUCTION FOR HERITAGE DOCUMENTATION. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLVIII-M-2-2023, 1113-1118.
- [3] Magnani M, Douglass M, Schroder W, Reeves J, Braun DR. The Digital Revolution to Come: Photogrammetry in Archaeological Practice. *American Antiquity*. 2020;85(4):737-760. doi:10.1017/aaq.2020.59
- [4] Burdziakowski, P. (2020). Increasing the Geometrical and Interpretation Quality of Unmanned Aerial Vehicle Photogrammetry Products using Super-Resolution Algorithms. *Remote Sensing*, 12(5), 810. MDPI AG. Retrieved from <http://dx.doi.org/10.3390/rs12050810>
- [5] Guidi, G., & Russo, S. (2019). Photogrammetry for architectural and archaeological documentation. In *Digital heritage*. Cham: Springer International Publishing.
- [6] Remondino, F., Karami, A., Yan, Z., Mazzacca, G., Rigon, S., & Qin, R. (2023). A Critical Analysis of NeRF-Based 3D Reconstruction. *Remote Sensing*, 15(14), 3585. MDPI AG. Retrieved from <http://dx.doi.org/10.3390/rs15143585>
- [7] Yu, A. W., Park, J., Peng, S., & Ma, C. (2021). PixelNeRF: Neural Radiance Fields from One or Few Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://openaccess.thecvf.com/content/CVPR2021/html/Yu_pixelNeRF_Neural_Radiance_Fields_From_One_or_Few_Images_CVPR_2021_paper.html
- [8] C. Zhang, H. Zhou, S. Shah, and H. Yujiang, "Computer vision-assisted photogrammetry and one-image 3D modeling in marine mammals," *ResearchGate*, Feb. 2023, [Online]. Available:https://www.researchgate.net/publication/370202518_Computer_vision-assisted_photogrammetry_and_one-image_3D_modeling_in_marine_mammals
- [9] Chudley, T. R., Christoffersen, P., Doyle, S. H., Abellan, A., & Snooke, N. (2019). High-accuracy UAV photogrammetry of ice sheet dynamics with no ground control. *The Cryosphere*, 13(3), 955-968. <https://doi.org/10.5194/tc-13-955-2019>
- [10] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan and P. Hedman, "Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 2022, pp. 5460-5469, doi: 10.1109/CVPR52688.2022.00539.

AniFrame: A Programming Language for 2D Drawing and Frame-Based Animation

Mark Edward M. Gonzales
De La Salle University
Manila, Philippines
mark_gonzales@dlsu.edu.ph

Elyssia Barrie H. Ong
De La Salle University
Manila, Philippines
elyssia_ong@dlsu.edu.ph

Hans Oswald A. Ibrahim
De La Salle University
Manila, Philippines
hans_oswald_ibrahim@dlsu.edu.ph

Ryan Austin Fernandez
De La Salle University
Manila, Philippines
ryan.fernandez@dlsu.edu.ph

ABSTRACT

Creative coding is an experimentation-heavy activity that requires translating high-level visual ideas into code. However, most languages and libraries for creative coding may not be adequately intuitive for beginners. In this paper, we present AniFrame, a domain-specific language for drawing and animation. Designed for novice programmers, it (i) features animation-specific data types, operations, and built-in functions to simplify the creation and animation of composite objects, (ii) allows for fine-grained control over animation sequences through explicit specification of the target object and the start and end frames, (iii) reduces the learning curve through a Python-like syntax, type inferencing, and a minimal set of control structures and keywords that map closely to their semantic intent, and (iv) promotes computational expressivity through support for common mathematical operations, built-in trigonometric functions, and user-defined recursion. Our usability test demonstrates AniFrame’s potential to enhance readability and writability for multiple creative coding use cases. AniFrame is open-source, and its implementation and reference are available at <https://github.com/memgonzales/aniframe-language>.

KEYWORDS

Creative coding, programming language design, exploratory programming, domain-specific language, animation

1 INTRODUCTION

Recent years have seen an increased interest in creative coding, or programming with an artistic rather than a functional intent, especially in engaging artistic expression and computational thinking among novice programmers [9, 10]. It has been described as an expressive and exploratory activity [2], with creative coders having to translate high-level visual ideas into code and go through multiple rounds of incremental refining (or even switching to another idea altogether) depending on the resulting output.

The difficulty of repeatedly mapping mental models to computer-compatible code is the crux of programming. Hence, from a human-centered viewpoint, programming languages and, by extension, libraries are not only notations or frameworks for writing code but, more importantly, user interfaces purposely designed to facilitate this mapping and make the programming task easier [17].

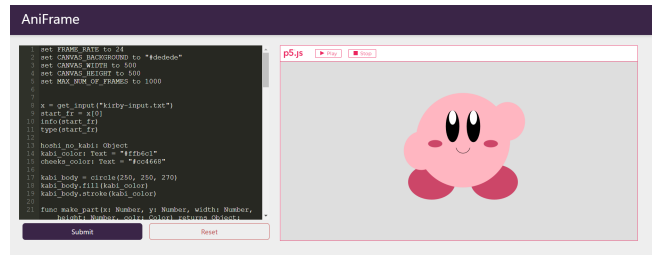


Figure 1: Sample AniFrame Code and Resulting Output. Designed for novice programmers, AniFrame supports a varied set of features that simplify the creation of composite objects and provide fine-grained control over animation, thus facilitating expressivity and exploration for creative coding.

To this end, libraries dedicated to creative coding [12–14, 21, 31] have been developed to provide an ecosystem and set of constructs for drawing and animation. However, using a library presupposes proficiency in the language for which it was created. Some syntactic and semantic aspects may also not be readily intuitive for novice coders. Examples include p5.js’ and Cinder’s use of braces for block demarcation, which they carry over from JavaScript and C++, respectively [13, 14, 24]; p5.js’ mechanics for layering drawn objects, which have been characterized as challenging [27]; and Cinder’s interfacing with OpenGL, which can pose a steep learning curve for programmers with no prior experience in graphics APIs [16].

Meanwhile, few domain-specific languages for animation have been designed, with most of them, such as ActionScript 3.0 [29] and Processing [26], following an object-oriented programming paradigm. Although this results in highly structured code, having to create classes and instantiate them even for simple programs impedes rapid prototyping and has been noted to increase the difficulty of learning for beginning programmers [20].

In this paper, we present AniFrame, an open-source domain-specific language for two-dimensional drawing and frame-based animation for novice programmers. The language’s core principles and features are as follows:

- *Ready Support for Animation-Specific Constructs.* AniFrame features animation-specific data types (e.g., for drawn objects

and colors), operations (e.g., for mixing colors and simplifying the layering of objects into composite objects), and built-in functions for shapes and affine transformations.

- *Fine-Grained Control Over Animation.* AniFrame adopts a frame-based strategy where programmers explicitly specify the object to be animated, along with the start and end frames for the animation sequence. Settings such as the frame rate and the total number of frames can also be configured.
- *Reduced Learning Curve.* AniFrame follows a Python-like syntax, limits the number of keywords and control structures to a minimum, and tries to use keywords that are close to their semantic intent (e.g., `Text` instead of `string`). Specifying data types is optional since type inferencing is enforced.
- *Computational Expressivity.* AniFrame supports common mathematical operations, built-in trigonometric functions, and user-defined recursive functions. Their utility is demonstrated in creating self-similar patterns, such as fractals.

2 RELATED WORKS

Similar to traditional programming, creative coding involves translating ideas into a sequence of instructions for a computer to process. However, it differentiates itself in that its goal is the expression of art rather than the development of functional software [9]. It thus opens a variety of functions, ranging from recreational (e.g., making digital sketches and installations [4]) to practical (e.g., introducing coding concepts to novice and non-programmers [10]) intents.

With the increased interest in creative coding in recent years, various tools have been developed to ease the coding process and support creativity among programmers and artists alike [4]. These tools can be broadly categorized into domain-specific languages, desktop-based libraries, and web-based libraries [27].

Among the few domain-specific languages that have been developed, the most widely used are Processing [26] and ActionScript [29]. Processing is a simplification of Java for animation. ActionScript was created to accompany the now-discontinued Adobe Flash, and, with its latest release (version 3.0), it became a superset of ECMAScript. Both follow an object-oriented paradigm (OOP), thus promoting reusability and modularity; however, this may be overly complicated for simple drawing and animation use cases, as well as cognitively demanding for novice programmers [20, 33].

Libraries for creative coding were developed later on, as languages and browsers extended support for more features and improved their portability. Desktop-based libraries include `vvvv` [5], which supports more sophisticated functionalities such as 3D rendering and machine learning but runs only on .NET ecosystems; `C4` [12], which provides a simplified API for mobile-specific features but runs only on iOS; and `Cinder` [13], which is cross-platform but can be difficult to learn for those without prior experience with OpenGL [27]. Web-based libraries, such as `p5.js` [14] and `Sketch.js` [31], are mostly written in JavaScript [27] and, therefore, presuppose knowledge of JavaScript, which may not be suited for beginners due to its global variable-based programming model [6].

While these tools have proved helpful in a variety of use cases, most domain-specific languages for creative coding assume familiarity with OOP, and libraries require experience with their base ecosystem or language, thereby increasing the learning curve for

beginners. AniFrame thus borrows design elements from these tools and also attempts to improve accessibility for novice coders, while still maintaining expressivity for more elaborate use cases.

3 LANGUAGE DESIGN

AniFrame’s design is grounded in (i) ready support for animation-specific constructs, (ii) fine-grained control over animation, (iii) reduced learning curve, and (iv) computational expressivity.

3.1 Animation-Specific Features

The domain-specificity of AniFrame derives from its animation-specific features, as reflected in its data types, operations, built-in functions, and frame-based approach to animation.

3.1.1 Data Types and Operations. AniFrame features three standard data types: (i) `Text` for strings, (ii) `Number` for floating-point and integer values, and (iii) `List` for collections of data that are possibly heterogeneous (i.e., of different data types). Moreover, it supports three domain-specific types: (i) `Object` for shapes and composite objects, (ii) `Color` for colors (which can be initialized in either hex or RGB), and (iii) `Coord` for coordinate pairs.

Selected operations between these domain-specific types are also permitted. For example, adding two objects creates a composite object, where the second operand is layered on top of the first operand. Adding two colors is equivalent to color mixing, while subtracting two colors is equivalent to color subtraction. Operations on coordinates are also defined in a component-wise fashion.

On a syntactic note, the word choices for data types were selected to be as intuitive and close as possible to their semantic intent; for instance, `Text` was chosen instead of `string`, and `Number` instead of `float` or `int`, following the results of the empirical study conducted by Stefik and Siebert [30] among novice programmers.

3.1.2 Built-In Functions. AniFrame provides a varied set of built-in functions for 2D drawing and animation, as listed in Table 1.

The largest class of built-in functions comprises those for drawing, styling, and animating objects. To facilitate rapid prototyping, each animation function has a version that applies the affine transformation only on the x -axis, only on the y -axis, and on both axes. In terms of syntax, their names purposely depart from their formal mathematical terminologies (e.g., `move`, `turn`, and `resize` instead of `translate`, `rotate`, and `scale`) in order to make their semantic intent more readily understood even by programmers without specialized mathematical background.

Common mathematical functions (e.g., square root and pseudo-random number generation) and trigonometric functions, which are important in programmatic animation [22], are also available out of the box for computational expressivity.

Moreover, in order to assist in debugging, the built-in method `info()` can be called to display values or, in the case of objects, their internal representations (discussed in Section 4.3), while `type()` can be invoked to display the data types of variables.

3.1.3 Frame-Based Animation. AniFrame adopts a frame-based approach to animation, where (i) all the built-in functions have required parameters for specifying the start and end frames, and (ii) an animation is applied only to the target object calling it. For

example, to move an object x by 3 units to the right starting at frame 10 and ending at frame 20, the code is `x.moveX(3, 10, 20)`.

This approach affords the programmer fine-grained control over animation sequences while still allowing for rapid prototyping since the appearance and position of the object in the in-between frames are automatically computed under the hood. In this regard, AniFrame takes after the idea of keyframes and tweening in Flash.

AniFrame's principle of applying an animation only to the target object calling it attempts to address pain points in the stack-based approach of OpenGL (and later inherited by Processing and p5.js), where an animation is applied to all the objects created subsequent to the animation function call unless reset via `pop()` [14, 15, 26].

3.2 Control Structures

AniFrame is a Turing-complete language in that it has sequential, conditional, and iterative control structures. With regard to conditionals, two-way selection using `if...else` and multiple selection using `else if` are supported. The decision to have only a single multiple selection structure (i.e., not supporting `switch...case` for instance) is deliberate; AniFrame tries to limit the number of keywords and control structures to a minimum in order to maintain a low learning curve for novice programmers.

AniFrame features three classes of iterative control structures: (i) precondition-controlled loops using `while`, (ii) count-controlled loops using `repeat`, and (iii) collection-controlled loops using `for...in`. Including a dedicated count-controlled loop is an attempt to increase readability and writability for use cases such as repeating an object's movement a predefined number of times. A `break` statement is also provided to prematurely escape loops. However, unconditional branching (e.g., via `goto`) is not supported in order to discourage "spaghetti" control flows [32].

3.3 Type System and Scoping Rules

AniFrame follows a static type system with support for both explicit and implicit type declaration (the latter via type inferencing). The motivation for preferring a static over a dynamic type system is to promote increased reliability and code maintainability [11]. Empirical results [23] have also pointed towards the advantage of static type systems for performing tasks that involve working with previously unknown API functions; in principle, this may also apply to AniFrame since a significant part of the initial learning curve entails developing familiarity with its built-in functions.

In terms of scoping, AniFrame follows static (lexical) scoping. Creating user-defined functions is supported, but nested functions are not allowed. Any variable declared outside a function is considered a global variable. A function can read the value of a global variable but cannot modify it, except when it is an object; this exception makes it easier to modularize the creation of composite objects by eliminating the need to pass the base object as a parameter (as demonstrated in Section 5.1). When a local variable inside a function shares the same name as that of a global variable, the global variable cannot be read inside that function. Since the target users of AniFrame are novice programmers, these restrictions are set to simplify the semantics of global scoping and prevent often-unintended side effects when working with global variables [25].

Listing 1: Sample Code Snippet for Demonstrating AniFrame's Implementation

```

1 nose: Object = rectangle(-650, 250, 800, 15)
2 nose.fill("#C2B280")
3 frame = 1
4 repeat 3:
5     nose.moveX(20, frame, frame + 100)
6     frame += 101
7 pinocchio: Object = circle(0, 250, 200)
8 pinocchio += nose

```

Syntactically, blocks are demarcated via indentations; in this regard, AniFrame takes after Python, which is well regarded for its readability and writability [1].

4 LANGUAGE IMPLEMENTATION

AniFrame is an interpreted language. To maintain a strict separation of concerns, its implementation is divided into three stages: (i) lexical analysis, (ii) parsing, and (iii) semantic analysis. The handling modules, namely the lexer (for lexical analysis), parser (for parsing), and interpreter (for semantic analysis), were written in Python using the language recognition tool ANTLR 4 [18].

4.1 Lexical Analysis

First, the lexer performs lexical analysis, converting the source code into a list of tokens following a longest-match-wins strategy [18] and stripping out comments. For illustration, a sample code snippet and its partial token stream are given in Listing 1 and Table 2, respectively. The patterns for the tokens are defined using regular expressions. In addition, since AniFrame borrows Python's indentation-based block demarcation, special tokens for indentations and outdentations are generated following the stack-based algorithm described in the Python Language Reference [8].

4.2 Parsing

After the lexical analysis, the parser creates a parse tree (Figure 2). This parser is generated via ANTLR's adaptive LL(*) parsing strategy, a predictive approach that involves an arbitrary lookahead and the launching of multiple pseudo-parallel subparsers for efficiency [19]. AniFrame's grammar is expressed using a variant of extended Backus-Naur form.

4.3 Semantic Analysis and Code Generation

After the parse tree is generated, the interpreter traverses it in order to determine the semantic intent of the statements in the source code. This traversal can be performed using either ANTLR's listener or visitor interface [18], but since it is necessary to return the results of visiting certain nodes (e.g., to update the symbol table) and propagate certain values up the parse tree, AniFrame's implementation employs the visitor pattern.

As the parse tree is traversed and expressions are resolved, the interpreter also builds a JSON-like intermediate code that consists

Table 1: Built-In Functions in AniFrame

Category	Functions
Shapes	point(), line(), curve(), circle(), ellipse(), triangle(), rectangle(), quad() (for quadrilaterals), polygon(), write() (for text boxes)
Styling	fill(), stroke()
Translation	move(), moveX(), moveY()
Rotation	turn(), turnX(), turnY()
Scaling	resize(), resizeX(), resizeY()
Shear	shear(), shearX(), shearY()
General Math	rand_num(), rand_int(), sqrt()
Trigonometry	sin(), cos(), tan(), asin(), acos(), atan(), atan2(), to_deg(), to_rad()
Miscellaneous	draw() (for placing objects on the canvas), info() (for displaying values or, in the case of objects, their internal representations), type() (for displaying data types)

Table 2: Partial Token Stream for Listing 1

Line #	Column #	Token	Lexeme
1	1	IDENTIFIER	nose
1	5	COLON	:
1	7	OBJECT_TYPE	Object
1	14	ASSIGNMENT_OP	=
1	16	RECTANGLE	rectangle
1	25	OPEN_PARENTHESIS	(
1	26	MINUS_OP	-
1	27	INTEGER_LITERAL	650
...

Listing 2: Intermediate Code Generated for Listing 1

```

1 "Drawing": {
2   "nose": [{"fill": "#C2B280",
3     "stroke": "DEFAULT_STROKE", "shape":
4     "rectangle(-650, 250, 800, 15)"}],
5   "pinocchio": [{"fill": "DEFAULT_FILL",
6     "stroke": "DEFAULT_STROKE", "shape":
7     "circle(0, 250, 200)", {"fill": "#C2B280",
8     "stroke": "DEFAULT_STROKE", "shape":
9     "rectangle(-650, 250, 800, 15)"}]
10 }
11 "Animation": {
12   "nose": [{"action": "moveX", "start": 1,
13     "end": 101}, {"action": "moveX", "start": 102,
14     "end": 202}, {"action": "moveX", "start": 203,
15     "end": 303}]
16 }

```

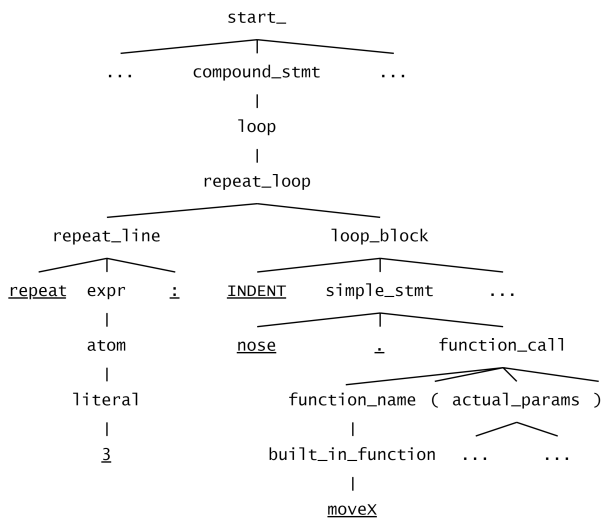


Figure 2: Partial Parse Tree for Listing 1. This figure focuses on the repeat loop (Lines 4 to 6). Terminals are underlined.

of two dictionaries: DRAWING and ANIMATION (Listing 2). DRAWING stores the drawn objects, their constituent shapes, and the styles (i.e., their stroke and fill colors) of each constituent shape. ANIMATION

stores the drawn objects, the animations applied to them, and the start and end frames for each animation.

The rationale for this internal representation is twofold. First, although programmers do not have to be aware of this under-the-hood representation, AniFrame provides a function to view it to facilitate debugging, especially for more experienced users. Hence, a conceptually intuitive and human-readable representation is preferable. Second, since the DRAWING and ANIMATION dictionaries are updated as nodes in the parse tree are visited, utilizing a data structure that supports $O(1)$ lookups and updates is critical to speeding up the intermediate code generation.

After the entire parse tree is traversed and the intermediate code is generated, the interpreter converts the intermediate code to the target code. The target code is in JavaScript and follows p5.js' semantics to allow the output to be displayed on web browsers using its player. To this end, each entry in the DRAWING dictionary is converted to a class. The entries in the ANIMATION dictionary are processed to yield a series of conditionals, each corresponding to a sequence of frames of continuous, identical animation.

4.4 Error Handling

Separation of concerns is also observed in error handling. Lexical errors (e.g., a token not matching any of the patterns in the lexer grammar) are caught by the lexer during lexical analysis. Syntax errors (e.g., mismatched parentheses) are caught by the parser during syntactic analysis. To complement ANTLR's default error recovery strategy and to output more tailored error messages, the parser grammar also includes special production rules for catching common errors, such as specifying an incorrect number of coordinates in a Coord pair. Semantic errors (e.g., incompatible operands) are caught by the interpreter during semantic analysis.

The impact of error messages on the programming experience of novice programmers has been emphasized in human-computer interaction studies [3, 7, 28]. To maximize the helpfulness and utility of error messages, AniFrame attempts to avoid terse, technical phrasing; for example, instead of "unsupported operand type(s) for +: 'int' and 'str'", AniFrame displays "+ operator between Number and Text is not supported." Line numbers of errors are displayed, along with column numbers for lexical and syntactic errors. AniFrame also takes advantage of ANTLR's adaptive LL(*) parsing and error recovery strategy, which are purposely designed to reduce cascading error messages [18, 19].

5 SAMPLE USE CASES

We demonstrate the utility and expressivity of AniFrame as a domain-specific language for creative coding through two sample use cases: animating a composite object and drawing a Sierpiński triangle via recursion.

5.1 Animating a Composite Object

The code for creating and animating a composite object in AniFrame and the output are provided in Listings 3 to 5 and Figure 3, respectively. As discussed in Section 3.1.1, the semantics of the addition operator simplifies the layering of objects into composite objects (Lines 9, 23, and 37 in Listing 3). It is also possible to create reusable object templates via user-defined functions (Listing 4 and Lines 17, 18, 20, and 21 in Listing 3).

The stacking of objects on the canvas is controlled by the order of the `draw()` statements in Lines 43 to 50 in Listing 3; the parameters of these statements pertain to the start and end frames of the objects' appearance on the canvas. Meanwhile, Listing 5 shows how custom animations can be created using AniFrame's built-in animation functions and control structures.

Other language constructs demonstrated in Listing 3 include explicit type declaration (Lines 7 and 8), type inferencing (Lines 1 and 2), and type coercion (Lines 1, 3, and 4; the inferred data type of `kabi_color` is `Text`, but it was implicitly converted to `Color` when passed as an argument to `fill()` and `stroke()`).

5.2 Drawing with Recursion

To show the computational expressivity of AniFrame, we use it to programmatically generate a Sierpiński triangle, a fractal with the overall shape of an equilateral triangle. It is recursively constructed by connecting the midpoints of an equilateral triangle, removing the central triangle formed, and repeating these steps for the remaining (smaller) equilateral triangles.

Listing 3: Sample Code for Creating a Composite Object

```

1 kabi_color = "#ffb6c1"
2 kabi_body = circle(250, 250, 270)
3 kabi_body.fill(kabi_color)
4 kabi_body.stroke(kabi_color)
5
6 cheeks_color = "#cc4668"
7 l_cheek: Object = ellipse(170, 260, 40, 20)
8 r_cheek: Object = ellipse(330, 260, 40, 20)
9 kabi_cheeks: Object = l_cheek + r_cheek
10 kabi_cheeks.fill(cheeks_color)
11 kabi_cheeks.stroke(cheeks_color)
12
13 # NOTE: Insert function definition of make_part(),
    as given in Listing 4
14
15 black = "#000000"
16 white = "#ffffff"
17 l_eye = make_part(220, 210, 38, 90, black)
18 l_eye_shine = make_part(220, 185, 20, 40, white)
19 l_eye += l_eye_shine
20 r_eye = make_part(280, 210, 38, 90, black)
21 r_eye_shine = make_part(280, 185, 20, 40, white)
22 r_eye += r_eye_shine
23 kabi_eyes: Object = l_eye + r_eye
24
25 smile: Object = curve(230, 150, 230, 272, 270,
    272, 270, 150)
26 smile.stroke(black)
27 smile.fill(kabi_color)
28
29 l_foot = make_part(160, 370, 135, 100,
    cheeks_color)
30 r_foot = make_part(340, 370, 135, 100,
    cheeks_color)
31
32 # NOTE: Insert function definition of make_hand(),
    as given in Listing 4
33
34 r_hand = ellipse(130, 280, 100, 110)
35 l_hand = make_hand()
36
37 hoshi_no_kabi: Object = r_hand + kabi_body
38 hoshi_no_kabi.fill(kabi_color)
39 hoshi_no_kabi.stroke(kabi_color)
40
41 # NOTE: Apply animation on l_hand (Listing 5)
42
43 l_foot.draw(1, 1000)
44 r_foot.draw(1, 1000)
45 l_hand.draw(1, 1000)
46 hoshi_no_kabi.draw(1, 1000)
47 l_eye.draw(1, 1000)
48 r_eye.draw(1, 1000)
49 kabi_cheeks.draw(1, 1000)
50 smile.draw(1, 1000)

```

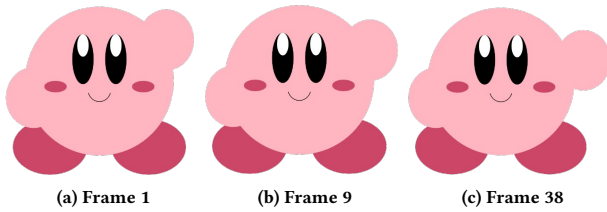


Figure 3: Composite Object Drawn and Animated in AniFrame. The figure (a character sprite waving its left hand) shows selected frames from the output of the code in Listings 3 to 5.

Listing 4: User-Defined Drawing Functions in Listing 3

```

1 func make_part(x: Number, y: Number, w: Number,
2   h: Number, color: Color) returns Object:
3   part: Object = ellipse(x, y, w, h)
4   part.fill(color)
5   part.stroke(color)
6   return part
7 func make_hand() returns Object:
8   sugoi = ellipse(360, 160, 100, 110)
9   sugoi.fill(kabi_color)
10  sugoi.stroke(kabi_color)
11  return sugoi

```

Listing 5: Animating the Left Hand (l_hand) in Listing 3

```

1 func wave_hand(frame: Number, delta: Number):
2   repeat(3):
3     l_hand.move(1, 2, frame, frame + delta)
4
5 func wave_hand_up(frame: Number, delta: Number):
6   repeat(3):
7     l_hand.move(-1, -2, frame, frame + delta)
8
9 frame = 1
10 delta = 12
11 repeat(3):
12   wave_hand(frame, delta)
13   wave_hand_up(frame + delta, delta)
14   frame += 2 * delta

```

The code for creating a Sierpiński triangle in AniFrame and the resulting output are provided in Listing 6 and Figure 4, respectively. Aside from recursion, this code also demonstrates other constructs in the language, including lists (Line 1 in Listing 6), support for common mathematical operations (`rand_int()` in Line 6 and `sqrt()` in Line 14), and global scoping for objects (Lines 7 and 13; global scoping is discussed in Section 3.3).

Listing 6: Sample Code for Creating a Sierpiński Triangle

```

1 colors = ["#CCE1F2", "#C6F8E5", "#FBF7D5",
2   "#F9DED7", "#F5CDDE", "#E2BEF1"]
3 func sierpinski(x1: Number, y1: Number,
4   x2: Number, y2: Number, x3: Number, y3: Number,
5   step: Number):
6   if step != 0:
7     shape = triangle(x1, y1, x2, y2, x3, y3)
8     shape.fill(colors[rand_int(0, 5)])
9     triangles += shape
10    sierpinski(x1, y1, (x1+x2)/2, (y1+y2)/2,
11      (x1+x3)/2, (y1+y3)/2, step-1)
12    sierpinski(x2, y2, (x1+x2)/2, (y1+y2)/2,
13      (x2+x3)/2, (y2+y3)/2, step-1)
14    sierpinski(x3, y3, (x3+x2)/2, (y3+y2)/2,
15      (x1+x3)/2, (y1+y3)/2, step-1)
16 triangles: Object = point(0, 0)
17 sierpinski(0, 0, 600, 0, 300*sqrt(3), 8)
18 triangles.draw(1, 100)

```

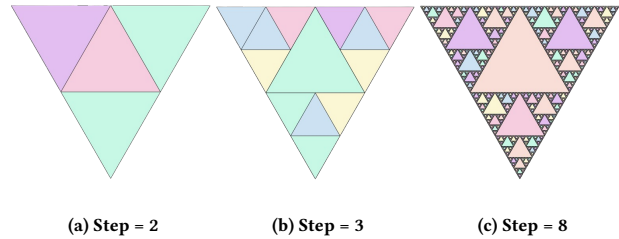


Figure 4: Sierpiński Triangle Drawn in AniFrame. The figure was generated programmatically using recursion. The code is given in Listing 6; the step is dictated by the last argument in Line 14.

6 USABILITY TEST

In order to have an initial assessment of AniFrame’s readability and writability, we conducted a preliminary usability test. Among the six respondents, three had taken an introductory programming course (covering variables, conditionals, and loops) but have been coding for less than four months; the other three have at least one year of programming experience.

The respondents were asked to complete five programmatic drawing and animation tasks (Table 3) in both AniFrame and p5.js. All six respondents had no previous exposure to AniFrame and p5.js and instead referred to their respective documentation while doing the tasks; in order to mitigate the influence of the documentation, we patterned AniFrame’s documentation after that of p5.js. The coding environment was a browser-based text editor without auto-complete. No time limit was set, but the respondents were allowed to give up if they felt that a task was too difficult.

Table 3: Tasks Given in the Usability Test. U1, U2, and U3 had taken an introductory programming course but have been coding for less than four months. U4, U5, and U6 have at least one year of programming experience. A tick mark (✓) indicates that the respondent was able to accomplish the task.

#	Task	Intent Tested	AniFrame						p5.js					
			U1	U2	U3	U4	U5	U6	U1	U2	U3	U4	U5	U6
1	Draw a circle and make it move from left to right	Drawing a shape and applying a single transformation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2	Display the text "Hello world" on a blue canvas	Working with Text, drawing a text box, and setting canvas-level configurations	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3	Draw a triangle and make it move up and down thrice at 5 frames per second	Drawing a simple shape and applying multiple transformations	✓	✓		✓	✓	✓					✓	✓
4	Draw a traffic light by creating red, green, and yellow signal lights and enclosing them in a rectangular border	Working with Color and drawing a composite object	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
5	Draw a simple face with eyes, nose, and mouth, and make the eyes (and only the eyes) move from left to right	Drawing a composite object and applying a transformation to only a component of it	✓	✓		✓	✓	✓			✓	✓	✓	✓

As seen in Table 3, all six respondents were able to complete Tasks 1 and 2 in both AniFrame and p5.js. Task 4 was completed by all six respondents in AniFrame and by five respondents in p5.js. On the other hand, Task 3 was accomplished by five respondents in AniFrame but only by two respondents in p5.js (both of which have at least one year of programming experience). Lastly, Task 5 was accomplished by five respondents in AniFrame and by four respondents in p5.js.

Task 3 involved making a triangle move up and down thrice. All six respondents first tried searching for a "move" function in the documentation but did not obtain any hits in p5.js since it uses "translate". In relation to this, U2, U4, and U5 mentioned that they appreciate AniFrame's use of the less technical term "move" since it is more accessible for beginners.

Coding in p5.js, U2 and U4 were able to move the triangle unidirectionally via `translate()` and `Vector` but were unable to implement the required change in direction. U5 did not use these constructs and instead opted for a frame-by-frame approach, manually adjusting the y -coordinate of the triangle per frame (Listing 7). U3 followed a similar approach but was unable to limit the motion to three times since they coded a loop for the counter whereas the semantics of p5.js require a conditional (Line 8 of Listing 7).

Meanwhile, coding in AniFrame, all six respondents utilized the built-in `moveY()` function (Listing 8). The respondents found the task to be easier in AniFrame (U1, U2, U4, U5, and U6), noting that it led to "significantly more readable and fewer lines of code" (U1) and required "less mental gymnastics since the repeat loop and `moveY()` work as expected compared to p5" (U2). However, one of the respondents (U3) was unable to finish the task since they did not adjust the `frame` parameter of the `moveY()` statement that corresponds to the change in direction.

Listing 7: p5.js Code of U5 for Task 3 of Usability Test (Moving a Triangle Up and Down Thrice)

```

1 function setup() {
2   createCanvas(720, 400);   frameRate(5);
3 }
4
5 y = 75;   delta = 5;   ctr = 3;
6
7 function draw() {
8   if (y <= 200 && y >= 0 && ctr + 1 >= 0) {
9     background(200);
10    triangle(30, y, 58, y-55, 86, y);
11    y += delta;
12  } else {
13    delta *= -1;   y += delta;   ctr--;
14  }
15 }

```

Task 5 involved drawing a simple face (with eyes, nose, and mouth) and animating only the eyes. Coding in p5.js, U2 was able to draw all the required facial features but, since they drew the eyes before the other features, the animation intended for the eyes cascaded down to the other features as a result of the stack-based semantics of p5.js (Section 3.1.3). This pain point was not observed in AniFrame since it was purposely designed such that an animation is applied only to the target object calling it. Moreover, the AniFrame code was described to be easier to write and trace since the target object is clearly specified when performing the animation function call (U2, U4, and U5).

Listing 8: AniFrame Code of U5 for Task 3 of Usability Test (Moving a Triangle Up and Down Thrice)

```

1 set FRAME_RATE to 5
2 i=0
3 repeat(3):
4     shape = triangle(30, 75, 58, 20, 86, 75)
5     shape.moveY(20, 1+i, 80+i)
6     shape.moveY(-20, 81+i, 160+i)
7     i += 160
8 shape.draw(1, 1000)

```

From the post-task semi-structured interview and the respondents' impressions, all six respondents cited AniFrame's readability as a major contributing factor to its suitability for beginning programmers. Its smaller set of functions also made it "less intimidating and a good starting point for animation" (U6).

With regard to the points for improvement, U3 and U5 noted that, although the frame-based approach allowed for fine-grained control, having to specify the start and end frames for every animation function call may not be readily intuitive for novice programmers and for those unfamiliar with the concept of frames in the first place. U2 suggested augmenting the coding environment with a visual representation of the frames vis-à-vis the animation timeline. Another recommendation was to allow for the canvas dimensions, canvas background, and frame rate to be accessible as special variables.

7 CONCLUSION

In this paper, we present AniFrame, an open-source domain-specific language for two-dimensional drawing and frame-based animation for novice programmers. Its design can be characterized as follows:

First, it features animation-specific data types, operations, and built-in functions for rapid creation and animation of composite objects. Second, it allows for fine-grained control over animation sequences through explicit specification of the target object, alongside the start and end frames. Third, it attempts to reduce the learning curve by adopting a Python-like syntax, supporting type inferencing, and using keywords that map closely to their semantic intent. Fourth, it promotes computational expressivity through built-in mathematical functions and support for recursion.

Our usability test points to AniFrame's potential to facilitate increased readability and writability for creative coding applications. Future directions include exploring syntactic and semantic improvements to enhance the intuitiveness of the frame-based approach, as well as providing support for three-dimensional graphics.

REFERENCES

- [1] Zahin Ahmed, Farishta Jayas Kinjol, and Ishrat Jahan Ananya. 2021. Comparative Analysis of Six Programming Languages Based on Readability, Writability, and Reliability. In *2021 24th International Conference on Computer and Information Technology (ICCIIT)*. 1–6.
- [2] Tyler Angert, Miroslav Suzara, Jenny Han, Christopher Pondoc, and Hariharan Subramonyam. 2023. Spellburst: A Node-based Interface for Exploratory Creative Coding with Natural Language Prompts. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Article 100.
- [3] Brett A. Becker, Paul Denny, James Prather, Raymond Pettit, Robert Nix, and Catherine Mooney. 2021. Towards Assessing the Readability of Programming Error Messages. In *Proceedings of the 23rd Australasian Computing Education Conference*. 181–188.
- [4] A. Benedetti, T. Elli, and M. Mauri. 2020. "Drawing with Code": The experience of teaching creative coding as a skill for communication designers (*12th International Conference on Education and New Learning Technologies*). 3478–3488.
- [5] Andrey Bo. 2012. Linear Spread: Illustrated Guide to vvvv for Newbies in Computer Arts. <https://visualprogramming.net/>.
- [6] Douglas Crockford. 2008. *JavaScript: The Good Parts*. O'Reilly Media, Inc.
- [7] Paul Denny, James Prather, Brett A. Becker, Catherine Mooney, John Homer, Zachary C Albrecht, and Garrett B. Powell. 2021. On Designing Programming Error Messages for Novices: Readability and its Constituent Factors. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Article 55, 15 pages.
- [8] Python Software Foundation. [n. d.]. The Python Language Reference - 2. Lexical analysis. https://docs.python.org/3/reference/lexical_analysis.html.
- [9] Ira Greenberg, Deepak Kumar, and Dianna Xu. 2012. Computational art and creative coding: teaching CS1 with Processing. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (North Carolina, USA)*. 660.
- [10] Ira Greenberg, Deepak Kumar, and Dianna Xu. 2012. Creative coding and visual portfolios for CS1. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (Raleigh, North Carolina, USA) (SIGCSE '12)*. 247–252.
- [11] Stefan Hanenberg, Sebastian Kleinschmager, Romain Robbes, Éric Tanter, and Andreas Steflk. 2014. An empirical study on the impact of static typing on software maintainability. *Empirical Softw. Engg.* 19, 5 (Oct 2014), 1335–1382.
- [12] Travis Kirton. 2013. C4: creative coding for iOS. In *International Conference on Tangible, Embedded, and Embodied Interaction*.
- [13] Rui Madeira and Dawid Gorny. 2013. *Cinder Creative Coding Cookbook*. Packt.
- [14] L. McCarthy, C. Reas, and B. Fry. 2015. *Getting Started with p5.js: Making Interactive Graphics in JavaScript and Processing*. Make Community, LLC.
- [15] Microsoft. [n. d.]. OpenGL - glPopMatrix function. <https://learn.microsoft.com/en-us/windows/win32/opengl/glpopmatrix>.
- [16] James R. Miller. 2014. Using a software framework to enhance online teaching of shader-based OpenGL. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education (Atlanta, Georgia, USA) (SIGCSE '14)*. 603–608.
- [17] Leonel Vinicio Morales Diaz. 2010. Programming languages as user interfaces. In *Proceedings of the 3rd Mexican Workshop on Human Computer Interaction (San Luis Potosí, Mexico) (MexIHC '10)*. 68–76.
- [18] Terence Parr. 2013. *The Definitive ANTLR 4 Reference*. Pragmatic Bookshelf.
- [19] Terence Parr, Sam Harwell, and Kathleen Fisher. 2014. Adaptive LL(*) parsing: the power of dynamic analysis. *SIGPLAN Not.* 49, 10 (Oct 2014), 579–598.
- [20] Nicolás Passerini and Carlos Lombardi. 2020. Postponing the Concept of Class When Introducing OOP. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education (Trondheim, Norway)*. 152–158.
- [21] Denis Perevalov. 2013. *Mastering openFrameworks: Creative Coding Demystified*. Packt Publishing.
- [22] Keith Peters. 2006. *Trigonometry for Animation*. Apress, Berkeley, CA, 41–68.
- [23] Pujan Petersen, Stefan Hanenberg, and Romain Robbes. 2014. An empirical comparison of static and dynamic type systems on API usage in the presence of an IDE: Java vs. Groovy with Eclipse. In *Proceedings of the 22nd International Conference on Program Comprehension (Hyderabad, India) (ICPC 2014)*. 212–222.
- [24] Yizhou Qian and James Lehman. 2017. Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review. *ACM Trans. Comput. Educ.* 18, 1, Article 1 (Oct 2017).
- [25] Anthony Ralston. 2003. *Side effect*. John Wiley and Sons Ltd., GBR, 1573–1574.
- [26] C. Reas and B. Fry. 2007. *Processing: A Programming Handbook for Visual Designers and Artists*. NBER.
- [27] Emil Sandberg. 2019. Creative Coding on the Web in p5.js: A Library Where JavaScript Meets Processing.
- [28] Eddie Antonio Santos. 2022. What makes a programming error message good?. In *Proceedings of the 2022 Conference on United Kingdom & Ireland Computing Education Research (UKICER '22)*. Article 21, 1 pages.
- [29] Rich Shupe and Zevan Rosser. 2010. *Learning ActionScript 3.0*. O'Reilly Media.
- [30] Andreas Steflk and Susanna Siebert. 2013. An Empirical Investigation into Programming Language Syntax. *ACM Trans. Comput. Educ.* 13, 4, Article 19 (Nov 2013), 40 pages.
- [31] Justin Windle. [n. d.]. Sketch.js. <https://soulwire.github.io/sketch.js/>.
- [32] W. A. Wulf. 1979. *A case against the GOTO*. Yourdon Press, USA, 83–98.
- [33] Stelios Xinogalos. 2015. Object-Oriented Design and Programming: An Investigation of Novices' Conceptions on Objects and Classes. *ACM Trans. Comput. Educ.* 15, 3, Article 13 (Jul 2015), 21 pages.

Comparative Analysis of Heuristics for the Offline Food Delivery Problem on Starlike Graphs

Jose Alfonso Barreiro
Ateneo de Manila University
Quezon City, Philippines
jose.barreiro@student.ateneo.edu

Mikael Giannes Bernardino
Ateneo de Manila University
Quezon City, Philippines
mikael.bernardino@student.ateneo.edu

Erick Gabriel Lopez
Ateneo de Manila University
Quezon City, Philippines
erick.lopez@student.ateneo.edu

John Paul Vergara
Ateneo de Manila University
Quezon City, Philippines
jpvergara@ateneo.edu

ABSTRACT

This paper explores the offline version of the Food Delivery Problem (oFDP) on starlike graphs. In oFDP, customer requests are sent to the central hub/restaurant (depot) which has a single server that has to decide where and when to deliver customer requests (orders). The server is required to return to the depot to pick up new orders before serving requests. The objective is to minimize the maximum flow time, i.e., the maximum time between the submission and completion of a request. Thus, this problem attempts to reduce the longest waiting time that a single customer may experience. The researchers study the oFDP on star-like graphs to develop several heuristics and compare their performance using generated experimental data. Four heuristics—FIFO, MRT, SAT, and OAT—were evaluated and benchmarked. OAT consistently performed well across different scenarios while SAT showed higher average approximation ratios, especially on certain graph types. FIFO and MRT exhibited acceptable performance but were sensitive to graph density. OAT emerged as a strong contender, balancing competitiveness and efficiency. Additionally, two properties were established: one highlighting the efficiency gain from batch delivery options, and the other emphasizing the strategic advantage of allowing waiting.

KEYWORDS

Offline food delivery problem (oFDP), starlike graphs, maximum flow time, heuristics, simulation, scheduling

1 INTRODUCTION

In the past decade, the landscape of purchasing and consuming foods has witnessed a substantial transformation [5], propelled by the rapid growth of online food ordering platforms. This trend experienced an unprecedented acceleration, with the recent global pandemic exerting a profound influence on the relevance of these services. The pandemic catalyzed a shift in lifestyles, prompting individuals to embrace remote work and confinement to their homes, thereby elevating the appeal of convenient online food delivery[6]. In consequence, traditional avenues for purchasing meals have faced limitations within this evolving context[1]. This transformative shift has ushered in a paradigm where online food

delivery has become a common choice, garnering favor among both consumers and businesses alike[6].

Nonetheless, this surge in demand for online food delivery services comes hand in hand with the potential challenge of optimizing delivery operations. The imminent risk of service overload necessitates a proactive approach to ensure seamless customer experiences. In this context, the need to manage delivery orders efficiently and mitigate potential delays assumes of utmost importance. Among the array of factors that collectively influence service quality of online food delivery, the duration elapsed between order placement and food delivery – commonly known as flow time – emerges as a critical dimension. This temporal aspect holds the potential to substantially shape the overall customer satisfaction for online food delivery. To further explore the complexities of optimizing delivery operations in the context of offline food delivery, this research aims to address the following questions: 1. What heuristics are applicable as solutions to the offline food delivery problem and how do they compare to one another in terms of efficiency, approximation ratio, robustness, and minimal traversal distance? 2. What properties can be derived from this problem, and how do they contribute to the performance of the algorithms?

In this study, we evaluate four heuristics in offline food delivery, comparing their efficiency and robustness. Additionally, we identify two key properties that contribute to their performance: the efficiency gain from batch delivery and the strategic advantage of allowing waiting. These findings advance understanding and optimization strategies in food delivery operations.

2 REVIEW OF RELATED WORK

In contrast to flow time scheduling problems that have primarily focused on preemptive solutions, the Food Delivery Problem presents a distinct scenario where preemption lacks practicality due to the preference for deploying additional servers over preempting those already in transit. This underscores the significance of the Food Delivery Problem (FDP) as an intriguing research area, driven not only by its real-world applicability, but also by its rarity as a flow time routing problem with the potential for non-trivial competitive algorithms[20].

The offline version of the food delivery problem (oFDP) shares close ties with numerous vehicle routing problems. For instance, the uncapacitated version (where $c = \infty$) with a single vehicle ($k = 1$) FDP, where all requests emerge at time 0, encompasses a variation of the path problem for the traveling salesman. In cases where the vehicle's capacity is finite ($c \neq \infty$), the previously mentioned FDP transforms into the Capacitated Vehicle Routing Problem (CVRP). A broader context is explored through the Dial-a-Ride Problem (DaRP), where not only the delivery destination but also individual pickup points can be specified by each customer. The vehicle is tasked with transporting the customer from their pickup location to their designated delivery spot. While much of the research on offline CVRP/DaRP has concentrated on the objective of minimizing total travel distance, these variations involve intricate problem nuances[21].

While the offline and online versions of the food delivery problem are closely linked, they have a clear distinction rooted in temporal dynamics. In the offline context, fixed parameters are known ahead, while the online scenario deals with incremental delivery requests over time. When deciding to start a delivery at time t , adherence to planned routes becomes crucial. This is formalized by scheduling k vehicle trips based solely on requests up to time t , connecting offline route design to online adaptability, where efficient decision-making in response to new requests is pivotal[20]. In the context of online food delivery optimization, Smith et al. proposed an innovative approach employing a capacitated multi pickup formulation and a branch-and-cut algorithm to efficiently determine least-cost vehicle routes, taking into account time windows, pickup and delivery constraints, as well as fleet capacity. Their method demonstrated significant success in solving benchmark problem instances with varying node sizes, addressing challenges previously reported as unsolved in the literature [22].

A similar problem to the OFDP is one referred to as the Online traveling salesman problem (Online TSP). In this problem, the server or salesman must navigate a graph to fulfill requests while trying to minimize the makespan all while only having limited information about future requests. Most solutions utilize a myopic approach to make dynamic decisions when creating the makespan. There is also a special case when the graph, which is a metric space, is on a real line and has received much attention in recent years [7, 10]. Several papers have observed that proper waiting could help improve the competitiveness of simple myopic algorithms that are implemented for Online TSP. Proper waiting is quite a useful concept as it allows for the mitigation of risks that arise from a myopic approach to decision making within the algorithm. The possibility of a better decision becoming available if a salesman were to merely wait for a given period of time is quite high yet only has minor overhead when utilized correctly [8, 14]. Currently, there are only competitive solutions for the minimal make-span variation of the Online TSP. To the best of our knowledge, if one were to change the objective to minimizing maximum flow time, there would be no algorithms currently that provide a competitive solution to the Online TSP with the goal of minimizing maximum

flow time. This provides an opportunity for further studies to explore this variation.

A related (Offline) TSP problem is the deadline TSP in which every vertex i in the graph can only be accessed by the traveling salesman within given time-windows $[R_i, D_i]$ of time that is set by the vertex [7]. These types of problems are quite difficult in general due to the constraints of the time-window as configuring a traveling sequence that matches the schedule while minimizing waiting times can be very complex. This principle can be applied to other problems that may recursively schedule trips by selecting longer trips ahead of time. The deadline TSP problem can also be applied to graphs that have a star or star-like topology which accurately represent the structure of companies that wish to deliver products to their clients with the guarantee of same day delivery [15, 16]. In this variation, a delivery driver is located at a single facility and must regularly return to the facility before completing delivery orders. Therefore, it closely mimics the problem in this paper but with additional restriction. Other studies have found that compiling multiple order requests on singular delivery trips within an arm of the star graph before returning to the depot is an effective measure at reducing the total delivery time; thus, ensuring that all orders are delivered [15,27].

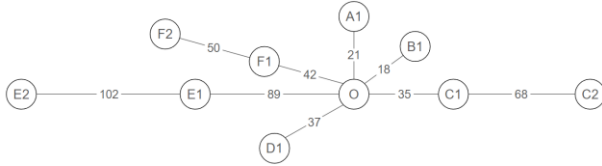
In addition, certain papers have expounded on these types of problems to focus on the aspect of fairness in real world systems as certain cities have different geographical areas with varying wealth and that greatly affects the performance of same day delivery to neighborhoods that are less affluent which causes political and class tension[9]. Researchers have attempted to use machine learning techniques to create a learning framework that can aid companies in adapting their delivery dispatching to have a fairer distribution of requests. This approach focuses on improving the regional service rate throughout the day by using Q-learning.

Another incentive for exploring the Offline Food Delivery Problem (FDP) emerges from an apparently unrelated challenge, the broadcast scheduling problem. In this distinct problem, a server maintains a collection of n pages, each with varying sizes, while requests for these pages are generated over time, each request being a query for one of the pages. The server can broadcast a page to all requests for that particular page, a process taking time equivalent to the size of the page. The overarching goal is to minimize the maximum flow time in this context. Intriguingly, the researchers can map the broadcast scheduling problem onto an uncapacitated single-vehicle FDP scenario on a star-shaped network, where each page's size in the broadcast scheduling problem corresponds to an edge in the FDP problem with a length of $s/2$. It's worth noting that the FIFO strategy is known to achieve $O(1)$ -competitiveness for the broadcast scheduling problem[8, 11, 15].

3. PRELIMINARIES

The problem addressed by this study is formally presented as the offline version of the Food Delivery Problem (FDP). Given a weighted star-like graph $G = (V, E)$ with edge lengths $\ell : E \rightarrow \mathbb{R}_{>0}$, where $\ell(e)$ denotes the time needed to traverse the edge e in either direction. There is a special vertex $o \in V$ called the depot,

which represents the restaurant. Located at depot o is a single server which has a capacity $c \in \mathbb{Z}_{>0} \cup \{\infty\}$. The function of this server is to deliver products from the depot to the customers located on the vertices along the different branches of the star-like graph. Customer orders come in the form of a set of \mathbb{R} requests that are sent to the depot. Each request $\rho \in \mathbb{R}$ is denoted by $\rho = (r_\rho, v_\rho)$, where $r_\rho \in \mathbb{R} \geq 0$ and $v_\rho \in V$ are the arrival time and the delivery location of the request respectively.



The server has the limitation that it must travel back to the depot before fulfilling any successive customer requests. In addition, the server cannot preemptively halt an order mid trip. This variation of the problem is offline which means that the server is aware of all requests even before they are released when planning trips. The output of the offline food delivery problem contains k sequences of trips correspondent to the itinerary of the server. The following properties need to be satisfied. For every pair of adjacent trips in any of the k sequences, the starting time of the latter trip is at least the completion time of the former one. Moreover, each request in R is served by exactly one trip in the k sequences; in other words, the sets of served requests in all trips of the k sequences form a partition of R . Let tp be the time that a request $\rho \in R$ is served (by the unique trip that serves it). The researchers define the flow time of ρ to be $tp - r_\rho$. The goal of the problem is to minimize the maximum flow time, i.e., $\max(\rho \in R (tp - r_\rho))$ which is the longest waiting time across all requests that a customer will experience.

The scope of this study is limited to star and star-like graphs of the offline version of the Food Delivery Problem (FDP). The strategies deliberated in this paper will be evaluated with respect to the characteristics of this particular graph structure. Evaluations involving strategies within broader contexts are reserved for prospective research endeavors. Emphasis will be placed on empirically comparing the heuristics in this study, while theoretical proofs of competitiveness will be included whenever feasible.

3.1 Trip Selection Heuristics

3.1.1 FIFO(First in First out):

The heuristic that is most intuitive and commonly used, where the hub will always serve the nodes or customers based on the arrival of their request ($T_{arrival}$). The FIFO heuristic ensures that nodes or customers are serviced in the order of their arrival ($T_{arrival}$). Mathematically, for any two nodes i and j where i arrived before j (i.e., $(T_{arrival, i} < T_{arrival, j})$), the service to node i will start before the service to node j .

Criteria 1 (Fairness):The FIFO heuristic maintains fairness in service provision, as it adheres to the principle of serving requests in the order they are received.

3.1.2. MRT (Min Return Time):

This heuristic will prioritize serving the customer node that will return the soonest with consideration with its priority value. Calculation for the estimated return time will be $\max(T_{Current} - T_{arrival}) + 2d - \max((T_{Current} - T_{arrival}), 0)$. Where “ $\max((T_{Current} - T_{arrival}), 0)$.” pertains to the priority value assigned to each node to prevent starvation which is when some of the requests are ignored for an indefinite period of time while other requests are favored by the heuristic. On the other hand, the symbol d pertains to the time of travel to or from between the depot and the customer node.

Criteria 2 (Priority Calculation): The MRT heuristic calculates priority values for nodes based on the time they spend in the queue and the time needed to complete their service. The longer a node remains in the queue and the faster it can be completed, the higher its priority becomes. This ensures that nodes that are faster to complete and have been waiting longer are serviced earlier, preventing potential starvation and promoting efficiency.

3.1.3. SAT (Single Arm Trips):

This heuristic is based on the MRT but focuses on completing entire arm trips to minimize overhead costs. This heuristic will prioritize serving an entire arm that will result in the server returning the soonest with consideration of its priority value. Arrival time for the entire arm is the maximum arrival time of its nodes $T_{ArmArrival} = \max(T_{NodeArr})$, and distance is the total distance from the hub to the leaf node in that arm $d = \max(d_n)$. After calculating values for the whole arm, the calculation for the return time will be the same as MRT $\max(T_{Current} - T_{ArmArrival}) + 2d - \max((T_{Current} - T_{ArmArrival}), 0)$.

Criteria 3 (Overhead Reduction): SAT reduces overhead costs by completing arm trips in a manner that minimizes the total distance traveled from the hub to the leaf node in the arm, optimizing resource allocation and energy usage.

3.1.4. OAT (Opportunistic Arm Traversal):

OAT is a hybrid heuristic combining elements of FIFO, SAT and MRT. This heuristic prioritizes cost-efficient whole-arm trips like SAT but seamlessly switches to single-node trips during downtimes. Downtime is defined as the difference between the current time and the arrival time of the next arm $DT = T_{current} - T_{Arr, Arm+1}$. In sparse graphs, where total request costs are less than the time span between the earliest and latest arrivals, OAT employs FIFO for optimal performance. This adaptability optimizes overhead costs while addressing varying graph characteristics.

Criteria 4 (Adaptability): OAT exhibits adaptability by dynamically adjusting the traversal strategy based on the system's state, allowing for efficient use of resources while considering real-time dynamics.

3.1.5. BruteForce:

Brute Force is a straightforward approach where all possible combinations or sequences of servicing nodes are exhaustively examined without any specific heuristic or optimization. In the context of trip selection, this would involve considering every

possible order in which nodes or customers can be serviced, regardless of their arrival time, priority, or any other factors.

Criteria 5 (Exhaustive Search): The Brute Force approach ensures an exhaustive search through all possible combinations of servicing nodes. It explores every permutation without relying on specific heuristics, making it a comprehensive but computationally intensive method.

3.2 Testing and Evaluation of Heuristics

The research design adopted for this study employs a systematic approach to investigate, develop, and rigorously test new and existing heuristics for the offline version of the Food Delivery Problem (FDP). This study encompasses the evaluation of novel heuristics, as described in section 3.1, alongside a comprehensive analysis of other potential heuristics that could be applied to address the problem statement presented.

3.2.1 Data Collection Techniques

As this study tackles the offline version of the food delivery problem, a structured representation is used in the form of starlike graphs. It also uses comparative analysis to garner insights from preexisting studies involving the offline food delivery problem and its possible solutions. Such solutions introduce the usage of a number of heuristics which guide the selection of requests for the problem. After analyzing these performance metrics, optimization techniques are employed to minimize the maximum flow time.

To simulate the performance of these heuristics, the study conducts testing using test cases designed to represent different situations. The testing is divided into two segments based on the number of requests in the graphs: cases with 10 or fewer requests and cases with more than 10 requests. This differentiation is crucial, as it addresses the limitations of the exhaustive brute force approach, which becomes less effective with larger graphs due to its factorial time complexity.

In total, the study generates and tests 5000 cases, 1500 cases among which are used as default scenarios with 250 cases allocated for each graph type, explained in section 3.2.3, while the remaining 2000 cases are predefined test cases representing real-world scenarios and randomly generated cases with more than 10 requests. where the Brute-force approach is unable to provide timely solutions due to its factorial time complexity.

By systematically collecting and analyzing data through these techniques, the study aims to evaluate and compare the effectiveness of various heuristics in addressing the offline food delivery problem.

1. Default
2. Dense Graphs
3. Sparse Graphs
4. Graphs with Many Arms
5. Graphs with few arms
6. Dense Requests
7. Sparse Requests

3.2.2 Performance Metrics

This section outlines a structured approach for evaluating and assessing the effectiveness of the heuristics using the three performance metrics described below:

Approximation Ratio of Each Heuristic's Minimum Maximum Flow Time: This metric compares the solution produced by each heuristic to the optimal solution, providing a measure of how close each heuristic gets to the best possible outcome. The minimum maximum flow time is a critical factor in food delivery logistics, as it represents the time it takes for the last delivery to be completed, affecting overall customer satisfaction and operational efficiency.[4]

Time Complexity: This metric quantifies the computational resources required by each heuristic to find a solution. Lower time complexity indicates faster performance and more efficient resource utilization, which is desirable in real-world applications where time is a crucial factor. [25]

Total Traversal Distance: This metric measures the total distance traveled by delivery vehicles while fulfilling orders. Minimizing traversal distance is important for reducing fuel costs, vehicle wear and tear, and overall environmental impact. [24]

3.2.3 Generation of Test Cases

In addition to the predefined test cases that represent real world case scenarios, the research introduces the generation of additional test scenarios using a discrete probability distribution, specifically the Poisson distribution. This generation process comprises two elements: the graph and the requests.

The graph is generated using the following procedure:

1. Randomize the number of arms.
2. For each arm, randomize the number of nodes along it.
3. For each edge, assign a randomized value for distance.

Table 1: Graph Generation's Key Parameters per Graph Type

	Default	Sparse Request	Dense Request	Sparse Graphs	Dense Graphs	Few Arms	Many Arms
Minimum number of arms	2	2	2	2	2	2	5
Maximum number of arms	8	8	8	8	8	3	24
Average number of nodes per arm*	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Average distance between nodes**	20	30	10	20	20	20	20
Average number of requests per arm*	1.5	1.5	1.5	0.125	10	1.5	1.5

* Randomized using Poisson distribution

** Distance is in 1 unit of time = 1 minute

Numbers are approximated from real life delivery scenarios i.e. Shakeys

Table 1 enumerates several key parameters that influence each process of the graph generation procedure, which are modified to fit each test scenario. The number of arms is randomized within a range set by the minimum and maximum number of arms. In addition, the average number of nodes per arm is another key parameter which is used during the randomization of the number of nodes per arm. Finally, the weight of each edge is generated randomly which utilizes a given average distance between the nodes which affects the density of the generated graph.

Given this generated graph, the researchers can then generate a list of requests:

1. For each arm in the graph, use Poisson distribution to generate the number of requests along an arm
2. For each request, select a node and randomize the request arrival time within the set time interval

Similar to the graph generation procedure, there are key parameters responsible for influencing the generation of the requests on the graph. Maximum time is the total time span. 1 hour time intervals were selected as a standard for test scenarios for the research. Another key parameter is the average number of requests which is used to determine the number of requests per arm of the graph through the usage of Poisson distribution.

As shown in Table 1, graph types are classified according to the values of specific parameters. Sparse request graphs feature a low average distance between nodes, while graphs with dense requests feature a higher than default value. To generate sparse and dense graphs, the average number of requests per arm are altered from the default value. Graphs with few arms and many arms have proportional minimum and maximum numbers of arms. Given these parameters, certain heuristics tend to perform better depending on the graph type. For example, FIFO and MRT would be more efficient on a sparse request graph due to the low average distance between nodes, but SAT perform sub optimally as requests are less dense, making single-arm trips less efficient.

The modification of these key parameters allows for the researchers to generate test scenarios under the following different classifications:

4. EXPERIMENTAL RESULTS

As stated in section 3.2.1, the experimental results section is divided into two segments: the first pertains to graphs with 10 or fewer requests, while the second addresses cases involving more than 10 requests. This differentiation is essential as the Brute-force approach exhibits limited efficacy when confronted with graphs containing more than 10 requests, primarily attributed to its $O(n!)$ time complexity. Consequently, the subsequent analysis and discussions will navigate these distinct scenarios to provide a nuanced understanding of the heuristic's performance under varying request loads.

Section 1: Approximation ratio in Minimizing Maximum Flow Time on 10 or fewer requests

Table 2: Heuristics as to Overall Approximation Ratio

	R_{Best}	R_{Ave}	R_{Worst}
FIFO	1.00	1.49	6.47
MRT	1.00	1.56	5.89
SAT	1.00	2.89	36.55
OAT	1.00	1.24	2.89
Bruteforce	1.00	1.00	1.00

Table 2 compares the approximation ratios of various heuristics for minimizing maximum flow time on 10 or fewer requests. FIFO and MRT achieve optimal ratios in the best-case scenario but perform

less efficiently in average and worst cases. SAT exhibits good performance initially but suffers significantly in worst-case scenarios. OAT, on the other hand, maintains consistently good ratios across scenarios, making it a balanced choice among the 4. Bruteforce achieves optimal ratios but is computationally impractical due to its exhaustive nature.

Table 3: Heuristic as to average approximation ratio on Different Graph Types

	Sparse Request	Dense Request	Sparse Graphs	Dense Graphs	Few Arms	Many Arms
FIFO	1.23	1.69	1.16	1.57	1.82	1.31
MRT	1.34	1.73	1.27	1.64	1.91	1.35
SAT	4.72	1.53	7.74	2.4	1.56	2.1
OAT	1.15	1.25	1.12	1.31	1.27	1.19
Bruteforce	1.00	1.00	1.00	1.00	1.00	1.00

Table 3 outlines Approximation Ratios for various heuristics across different graph scenarios in the context of the Offline version of the Food Delivery Problem (FDP). SAT exhibits notably higher average approximation ratios across all scenarios, particularly on sparse graphs and on cases with sparse request distribution, indicating potential inefficiency in these specific situations. Conversely, OAT consistently maintains lower ratios across different graph types and request densities, suggesting its robust performance in various scenarios. Additionally, while FIFO and MRT perform adequately, they show increased ratios on dense graphs compared to sparse ones, highlighting the impact of graph density on the heuristic's efficiency.

Section 2: Approximation Ratio of Minimizing Maximum Flow Time on More Than 10 Requests

Table 4: Heuristics as to Overall Approximation Ratio relative to the Best output among all the heuristics

	R_{Best}	R_{Ave}	R_{Worst}
FIFO	1	2.05	5.84
MRT	1	2.06	5.84
SAT	1	2.04	18.47
OAT	1	1.02	1.81
Bruteforce	-	-	-

Table 4 highlights the relative performance of heuristics concerning their approximation ratios compared to the best performing heuristic at each particular case. OAT stands out by consistently maintaining approximation ratios close to the best case, indicating its robustness across different scenarios. Conversely, FIFO, MRT, and SAT show higher ratios relative to the best answer, particularly in worst-case scenarios, suggesting potential inefficiencies in these situations. This underscores the importance of algorithm choice, with OAT offering promising performance in minimizing maximum flow time.

Table 4: Heuristic's Performance Relative to Time Complexity

	$R_{Worst}(≤10)$	$R_{Worst}(10+)$	Time Complexity
FIFO	6.47	5.84	$O(n)$

MRT	5.89	5.84	$O(n^2)$
SAT	36.55	18.47	$O(n^2)$
OAT	2.89	1.81	$O(n^2)$
Bruteforce	1.00	-	$O(n!)$

As achieving a delicate equilibrium between competitiveness and computational efficiency is important, OAT stands out as the best and preferred choice. It maintains a worst case approximation ratio of 2.89 for smaller inputs and 1.82 for larger inputs, indicating effective performance with a quadratic time complexity ($O(n^2)$), making it more scalable than the factorial time complexity of the Brute force. But if scalability for considerably larger datasets is a primary concern, FIFO (First-In-First-Out) might be more suitable. Its approximation ratio of 6.47 for smaller inputs and 5.84 for larger inputs, paired with a linear time complexity ($O(n)$), indicates more efficient scaling with increasing input sizes compared to heuristics with quadratic or factorial complexities.

The time complexity of FIFO, denoted as $O(n)$, reflects its straightforward and linear handling of requests. Each request is processed sequentially without any nested loops or additional calculations, leading to a direct scaling with the number of requests, denoted as 'n'. In contrast, MRT, SAT, and OAT exhibit quadratic time complexity, $O(n^2)$, as their operations often involve nested loops or calculations that increase quadratically with the number of requests. These heuristics commonly integrate loops for priority calculation and starvation prevention, contributing to their increased computational load.

For the Brute force approach, characterized by a factorial time complexity of $O(n!)$, exhaustive exploration of all possible combinations is employed by considering every permutation of the input elements. This exhaustive search strategy entails exploring 'n * (n-1) * (n-2) * ... * 1 = n!' possible combinations, where the number of possibilities decreases by one at each step. Consequently, the computational effort grows factorially with the number of requests, making it impractical for larger datasets and rendering it inefficient for many real-world applications.

5. PROPERTIES

In addition to heuristic evaluations, we established two properties to deepen our understanding of the problem:

Property 1: Batch Delivery Option Enhances Efficiency

This property highlights the efficiency gain from batch delivery options, affirming that batch delivery generally outperforms individual deliveries in terms of overall efficiency. Leveraging batch delivery strategies can reduce travel time and enhance resource utilization effectively.

Property 2: Option For Strategic Waiting Is Advantageous

This property emphasizes the strategic advantage of allowing waiting in decision-making processes. By demonstrating that there exists at least one solution for any graph G and set of request R where waiting improves decision quality, this property underscores the importance of flexibility in logistical operations. Incorporating waiting options can lead to better outcomes, particularly in scenarios requiring optimization of delivery schedules.

5.1 Elaboration on Key Properties

Property 1: Batch Delivery Option Enhances Efficiency. Improving Efficiency through the Batch Delivery Option. The availability of an option to perform batch delivery, particularly when serving multiple customers on the same arm simultaneously, enhances overall delivery efficiency by reducing travel time and optimizing resource utilization.

Proposition: For any given set of delivery requests R_n , the overall delivery efficiency achieved through the batch delivery option E_{batch,R_n} is higher or equal to the efficiency of serving single customers individually E_{single,R_n}

$$E_{batch,R_n} \geq E_{single,R_n}$$

Definitions:

E_{batch,R_n} : Overall delivery efficiency when the batch delivery option is utilized for the set of requests R_n .

E_{single,R_n} : Overall delivery efficiency when single customers are served individually for the set of requests R_n .

T_{batch} : Total travel time for batch delivery.

T_{single} : Total travel time for individual deliveries.

Base Case: When addressing a single delivery request ($|R|=1$), the delivery efficiency achieved through the batch delivery option is equivalent to serving a single customer individually $E_{batch,R_n} = E_{single,R_n}$. Thus, the base case holds.

Inductive Step: For any set of delivery requests R_{n+1} , let's assume the proposition holds true for R_n i.e., $E_{batch,R_n} \geq E_{single,R_n}$

Case 1: Batch Delivery Option

When the option to perform batch delivery is allowed for the set of requests R_{n+1} , let $|R_{n+1}|=k$, where $k > 1$. Denote the total travel time for batch delivery as T_{batch} . Then, the overall delivery efficiency with batch delivery is:

$$E_{batch,R_{n+1}} = \frac{k}{T_{batch}}$$

Case 2: Serving Single Customers

If only serving single customers individually is considered, the overall delivery efficiency is determined based on individual deliveries: $E_{single,R_{n+1}}$. Without the option for batch delivery, the efficiency relies on the traditional approach of serving single customers separately. $\frac{1}{T_{single}}$

Comparing Cases:

To show that $E_{batch,R_n} \geq E_{single,R_n}$, we need to show

$$\frac{k}{T_{batch}} \geq \frac{1}{T_{single}}$$

Consider the time saved by batch delivery, denoted by $\Delta T = T_{single} - T_{batch}$. We can express T_{batch} in terms of T_{single} and ΔT as $T_{batch} = T_{single} - \Delta T$

$$\text{Therefore, } E_{batch,R_{n+1}} = \frac{k}{T_{single} - \Delta T}$$

Since ΔT represents time saved, $T_{single} - \Delta T \leq T_{single}$, which implies $\frac{1}{T_{single} - \Delta T} \geq \frac{1}{T_{single}}$

Thus, $E_{batch,Rn+1} \geq E_{single,Rn+1}$

Property 2: Option For Strategic Waiting Is Advantageous.

Proposition: Let G denote any graph and R denote any set of requests. Then, there exists a solution S , such that the decision quality when waiting is allowed denoted $q_{with_waiting}$, is greater than or equal to the decision quality when waiting is not allowed, denoted $q_{without_waiting}$.

In symbolic notation:

$$\forall G \forall R \exists S: q_{with_waiting} \geq q_{without_waiting}$$

This statement asserts that regardless of the specific graph or set of requests, there exists at least one solution where the decision quality achieved by waiting is greater than or equal to the decision quality obtained without waiting.

Proof: To prove this property, we need to show that there exists at least one solution where the decision quality achieved by strategic waiting is greater than or equal to the decision quality obtained without waiting.

Let's denote:

$Q_{no_wait}(G,R)$ as the decision quality without waiting.

$Q_{with_wait}(G,R,W)$ as the decision quality with strategic waiting, where W represents the waiting strategy.

The change in decision quality due to strategic waiting is given by:

$$\Delta Q(G,R,W) = Q_{with_wait}(G,R,W) - Q_{no_wait}(G,R)$$

Now, we need to prove that for any graph G and any set of requests R , there exists a solution S such that: $q_{with_wait} \geq q_{no_wait}$, where: q_{with_wait} and q_{no_wait} represent the decision qualities achieved with and without waiting, respectively. Mathematically, we can represent this as:

$$\forall G \forall R \exists S: Q_{with_wait}(G,R,W) \geq Q_{no_wait}(G,R)$$

To prove this statement, let's consider All possible cases:

Case 1: $Q_{with_wait}(G,R,W) = Q_{no_wait}(G,R)$. If the decision quality with waiting is equal to the decision quality without waiting, then the property holds trivially.

Case 2: $Q_{with_wait}(G,R,W) > Q_{no_wait}(G,R)$. If the decision quality with waiting is greater than the decision quality without waiting, then the property holds true.

Case 3: $Q_{with_wait}(G,R,W) < Q_{no_wait}(G,R)$. Let's denote the maximum flow time achieved with waiting as $Q_{max_with_wait}$ and without waiting as $Q_{max_no_wait}$. Then, we have:

$$Q_{max_with_wait} = \max_{\rho \in R}(tp - r\rho)$$

$$Q_{max_no_wait} = \max_{\rho \in R}(tp' - r\rho)$$

Where tp is the time at which request ρ is served when waiting is allowed, and tp' is the time at which request ρ is served when waiting is not allowed.

Given $Q_{with_wait}(G,R,W) < Q_{no_wait}(G,R)$.

we have: $Q_{max_with_wait} < Q_{max_no_wait}$

Let's consider the definition of $Q_{max_with_wait}$ and $Q_{max_no_wait}$ in terms of tp and tp' . Since both tp and tp' are arrival times plus the time taken for service, we can express them as:

$$tp = r\rho + \text{service_time}\rho$$

$$tp' = r\rho + \text{service_time}\rho'$$

where $\text{service_time}\rho$ and $\text{service_time}\rho'$ are the service times for request ρ with and without waiting, respectively.

Since waiting allows the server to optimize its route and potentially reduce service times, we can express $\text{service_time}\rho$ as:

$$\text{service_time}\rho = \min_service_time\rho$$

where $\min_service_time\rho$ is the minimum possible service time for request ρ when waiting is allowed. Therefore, we have:

$$tp = r\rho + \min_service_time\rho$$

Now, let's examine tp' . Since waiting is not allowed, the service time $\text{service_time}\rho'$ remains the same as the minimum service time when waiting is allowed. Hence:

$$tp' = r\rho + \min_service_time\rho$$

Given these expressions for tp and tp' , we can see that both $Q_{max_with_wait}$ and $Q_{max_no_wait}$ are equal, making the case of $Q_{with_wait}(G,R,W) < Q_{no_wait}(G,R)$ impossible.

Thus, the assumption that $Q_{with_wait}(G,R,W) < Q_{no_wait}(G,R)$ is invalid, and, therefore, $Q_{with_wait}(G,R,W) \geq Q_{no_wait}(G,R)$ holds true.

6. CONCLUSION

This study conducted a thorough evaluation and comparison of four distinct heuristics—FIFO, MRT, SAT, and OAT—against each other and the exhaustive brute force approach, as elaborated in Section 3.1.

The results reveal that SAT stands out with notably higher average approximation ratios across all scenarios, particularly on sparse graphs with many arms, suggesting potential inefficiency in these specific situations. Conversely, OAT consistently maintains lower ratios across different graph types and request densities, indicating its robust performance in various scenarios. Moreover, FIFO and MRT demonstrate acceptable performance, but they exhibit increased ratios on dense graphs compared to sparse ones,

highlighting the sensitivity of the heuristic efficiency to graph density variations.

In addition to heuristic evaluations, we established two properties to deepen our understanding of the problem: Property 1 highlights the efficiency gain from batch delivery options, affirming that batch delivery generally outperforms individual deliveries in terms of overall efficiency. Leveraging batch delivery strategies can reduce travel time and enhance resource utilization effectively. Property 2 emphasizes the strategic advantage of allowing waiting in decision-making processes. By demonstrating that there exists at least one solution for any graph G and set of request R where waiting improves decision quality, this property underscores the importance of flexibility in logistical operations. Incorporating waiting options can lead to better outcomes, particularly in scenarios requiring optimization of delivery schedules.

Overall, when considering both time complexity and average approximation ratio which are the key criteria in evaluating heuristics, the Opportunistic Arm Traversal heuristic emerges as a strong contender as it strikes a balance between competitiveness and computational efficiency, demonstrating effectiveness across different graph types and request densities.

ACKNOWLEDGMENTS

Dr. Andrei D. Coronel, PhD, Jose Alfredo A. de Vera III, Winfer Tabares, Felix II P. Muga: Our distinguished panelists whose critical role in shaping the quality of our work during defense sessions. We thank them for their insightful critiques and invaluable guidance, which significantly contributed to the refinement of our research.

Ateneo de Manila University: We extend our sincere appreciation to Ateneo for providing the conducive environment and resources that allowed this research to flourish. Family, Friends, Classmates, and Teachers: To our support network, we express gratitude for your unwavering encouragement, serving as a constant source of motivation throughout this journey. This study is a collaborative effort, and each acknowledgment represents a vital piece of the puzzle that has culminated in its completion. To everyone who played a part in this journey, we extend our deepest thanks.

REFERENCES

- [1] Aday, S., & Aday, M. S. (2020). Impact of COVID-19 on the food supply chain. *Food Quality and Safety*, 4(4), 167-180. <https://doi.org/10.1093/fqsafe/fyaa024>
- [2] Agnets, A., Cosmi, M., Nicosia, G., & Pacifici, A. (2023). Two is better than one? Order aggregation in a meal delivery scheduling problem. *Computers & Industrial Engineering*, 183, 109514. doi:10.1016/j.cie.2023.109514
- [3] ANON. (2023). Poisson Distribution. Retrieved September 24, 2023 from <https://www.sciencedirect.com/topics/mathematics/poisson-distribution>
- [4] Ausiello, G., Paschos, V. Th. (2006). Reductions, completeness and the hardness of approximability. *European Journal of Operational Research*, 172(3), 719-739. <https://doi.org/10.1016/j.ejor.2005.06.006>
- [5] Axon, S., Lent, T., Njoku A. (2023). Shifting sustainable lifestyle practices and behaviour during times of pandemic disruptive change: Implications for on-going socio-technical transitions. *Energy Research & Social Science*, 102, <https://doi.org/10.1016/j.erss.2023.103188>.
- [6] Balieva, G. (2023). Online food purchasing during COVID-19 pandemic. *Scientific Papers Series Management, Economic Engineering in Agriculture and Rural Development*, 23(2), 51-58. <https://managementjournal.usamv.ro/>
- [7] Bansal, N., Blum, A., Chawla, S., & Meyerson, A. (2004). Approximation algorithms for deadline-TSP and vehicle routing with time-windows. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing (STOC '04)* (pp. 1–10). doi:10.1145/1007352.1007385

- [8] Bansal, N., Charikar, M., Khanna, S., & Naor, J. (2005). Approximating the average response time in broadcast scheduling. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)* (pp. 215–221). Philadelphia, PA: Society for Industrial and Applied Mathematics.
- [9] Bienkowski, M., Kraska, A., & Liu, H.-H. (2021). Traveling repairperson, unrelated machines, and other stories about average completion times. *arXiv preprint arXiv:2102.06904*.
- [10] Bjelde, A., et al. (2000). Tight bounds for online TSP on the line. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 994–1005). New York: ACM.
- [11] Borodin, A., & El-Yaniv, R. (2005). *Online computation and competitive analysis*. Cambridge, MA: Cambridge University Press.
- [12] Chang, J., Erlebach, T., Gailis, R., & Khuller, S. (2011). Broadcast scheduling: Algorithms and complexity. *ACM Transactions on Algorithms*, 7(4), 1–14. doi:10.1145/2000807
- [13] Chen, P.-C., Demaine, E. D., Liao, C.-S., & Wei, H.-T. (2019). Waiting is not easy but worth it: The online TSP on the line revisited. *arXiv preprint arXiv:1907.00317*.
- [14] Chen, X., Wang, T., Thomas, B. W., & Ulmer, M. W. (2023). Same-day delivery with fair customer service. *European Journal of Operational Research*, 308(2), 738–751. doi:10.1016/j.ejor.2022.12.009
- [15] Chekuri, C., Im, S., Moseley, B., Fiat, A., & Sanders, P. (2009). Minimizing maximum response time and delay factor in broadcast scheduling. In *Algorithms - ESA 2009* (pp. 444–455). Berlin, Heidelberg: Springer.
- [16] Cosmi, M., Oriolo, G., Piccialli, V., & Ventura, P. (2019). Single courier single restaurant meal delivery (without routing). *Operations Research Letters*, 47(6), 537–541. doi:10.1016/j.orl.2019.09.007
- [17] Feuerstein, E., & Stougie, L. (2001). On-line single-server dial-a-ride problems. *Theoretical Computer Science*, 268(1), 91–105. doi:10.1016/S0304-3975(00)00261-9
- [18] Fiat, A., & Woeginger, G. J. (1998). *Competitive analysis of algorithms*. In *Online algorithms: The state of the art* (pp. 1–12). Berlin, Heidelberg: Springer.
- [19] Juris Hartmanis and Richard E. Stearns (1965). On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117(1965), 285–306.
- [20] Guo, X., Li, S., Luo, K., & Zhang, Y. (2021). Online food delivery to minimize maximum flow time. *arXiv preprint arXiv:2110.15772*.
- [21] Guo, X., Luo, K., Tang, Z. G., & Zhang, Y. (2022). The online food delivery problem on stars. *Theoretical Computer Science*, 928, 13–26. doi:10.1016/j.tcs.2022.06.007
- [22] Kohar, A., & Jakhur, S. K. (2021). A capacitated multi pickup online food delivery problem with time windows: A branch-and-cut algorithm. *Annals of Operations Research*. doi:10.1007/S10479-021-04145-6
- [23] Lopez, J. (2023). Examining the online food delivery problem on starlike graphs.
- [24] Nicola, D., Vetschera, R., & Dragomir, A. (2019). Total distance approximations for routing solutions. *Computers & Operations Research*, 102, 67-74. <https://doi.org/10.1016/j.cor.2018.10.008>
- [25] Reyes, D., Erera, A. L., & Savelsbergh, M. W. P. (2018). Complexity of routing problems with release dates and deadlines. *European Journal of Operational Research*, 266(1), 29–34. doi:10.1016/j.ejor.2017.09.020
- [26] Russell, S. J., Norvig, P., & Chang, M.-W. (2023). *Artificial intelligence: A modern approach* (4th ed.). Pearson.
- [27] Ulmer, M., Nowak, M., Mattfeld, D., & Kaminski, B. (2020). Binary driver-customer familiarity in service routing. *European Journal of Operational Research*, 286(2), 477–493. doi:10.1016/j.ejor.2020.03.037

Performance of various protein representations for predicting phage-host interaction

Franz Stewart V. Dizon
Bioinformatics Lab
Advanced Research Institute for
Informatics, Computing and
Networking
De La Salle University
Manila, Philippines
franz_dizon@dlsu.edu.ph

Jennifer C. Ureta
Bioinformatics Lab
Advanced Research Institute for
Informatics, Computing and
Networking
De La Salle University
Manila, Philippines
jennifer.ureta@dlsu.edu.ph

Anish M.S. Shrestha
Bioinformatics Lab
Advanced Research Institute for
Informatics, Computing and
Networking
De La Salle University
Manila, Philippines
anish.shrestha@dlsu.edu.ph

ABSTRACT

With the rise of antimicrobial resistance that decreases the effectiveness of antibiotics in treating bacterial infections, phage therapy is being studied as an alternative to antibiotics. Phage therapy is the use of phages to treat bacterial infections by letting the phages infect and lyse the bacterial pathogen at the site of infection. Phages are known to be able to infect a narrow range of hosts only, but laboratory experiments to verify an interaction between a phage and a bacterium are both costly and time-consuming. To mitigate this, several studies have explored the use of machine learning classifiers to predict whether a phage-host pair interacts or not. In this study, we formulated the prediction problem as a binary classification problem with the host and phage proteomes as input, and explored different kinds of protein representations, including protein embeddings that are generated by protein language models, that can serve as an input to machine learning classifiers. In our experiments, under a phylogeny-based train-test data split, protein embeddings did not necessarily improve classifier performance compared to using the conventional k-mer profile representation.

KEYWORDS

Phage, anti-microbial resistance, phage-host pair interaction prediction, protein language models, machine learning classifier

1 INTRODUCTION

Antimicrobial resistance (AMR) threatens the use of antibiotics by decreasing its effectiveness in treating bacterial infections [19, 24], which results in longer duration of illness, higher rates of mortality, increased costs of treatment, and inability to perform procedures that rely on effective antibiotics to prevent infection [19]. In 2019 alone, there were an estimated 1.27 million deaths globally directly attributable to drug resistance [27]. Resistance arises due to the excessive use of antibiotics, which exerts selective pressure that allows microorganisms that have developed resistance, to have a competitive advantage to survive and proliferate [14, 19].

With the emergence of antibiotic-resistant strains of life threatening microbes [4], phage therapy is being studied as an alternative to antibiotics [11, 15]. Phages are viruses that are capable of infecting and replicating within bacterial cells [13]. Phage therapy is the use of phages to treat bacterial infection, which is done by letting the phages infect and lyse the bacterial pathogen at the site of infection [13, 23]. A specific strain of phage is known to be able

to infect a narrow range of hosts only, which can be an advantage compared to a wider host range of antibiotics, that not only kills the target pathogenic bacteria but also other bacteria, some of which might be beneficial [13, 33].

Experimental methods used to identify phage-specific hosts require costly and time-consuming lab experiments to verify whether there is an interaction between the phage and the host [21]. To mitigate the cost and time consumed by lab experiments, computational approaches have been developed.

Recently, the use of machine learning classifiers for predicting phage-host interaction have been explored. One group of studies uses genomic information from only the phages. Young et al. [37] formulated the problem as taking the genome representation (DNA k-mer, amino acid k-mer, physio-chemical k-mer, protein domains) of the phage as input, and predicting a possible host. They trained a Support Vector Machine classifier separately for each host. Boeckaerts et al. [5] used only the phages' Receptor-Binding Proteins (RBP) as input. An RBP was represented by a vector composed of handcrafted DNA and protein features such as nucleotide frequencies, GC-content, codon frequencies, and others. The RBP vector representation served as the input to their multi-class classifier to identify the phage's possible hosts. Mark et al. [12] extended Boeckaerts et al.'s study by using Protein Language Models (PLM) to acquire vector representations, also called as protein embeddings, which serve as the input to their multi-class classifier.

There are also studies that formulate the phage-host interaction prediction problem as a binary classification problem that takes as input features derived from both the phage and the host, and predict whether the phage-host pair will interact or not. PredPHI [21] is a neural network classifier consisting of convolutional and fully connected layers. The input to PredPHI is a matrix that consists of derived protein features across the whole proteome of both the phage and the host, such as frequency of amino acids, abundance of each chemical component, and molecular weights. PHIAF [22], similar to PredPHI, incorporates an attention layer into its neural network, adds DNA-derived features for its input, and uses a Generative Adversarial Network (GAN) for data augmentation. PhageHostLearn [7] uses RBPs of phages and K-locus proteins as the input, and predicts whether the phage-host pair will interact or not. Embeddings for the protein sequence, obtained using the ESM-2 protein language model, served as an input to an XGBoost classifier.

In this study, we defined the phage-host prediction problem as taking the proteome (entire set of proteins) of both the phage and the host as the input, and deciding whether the phage-host pair interacts or not. We tested embeddings produced by different PLMs, namely ProtVec[2], Seq2Vec[18], ProtBert[10], and Prot5[10] to represent the protein sequences of both the host and the phage. We also tested non-PLM-based protein representations prior studies have used, such as k-mer profile and statistical profile[21], to determine if the use of protein embeddings does improve the prediction of phage-host interaction. In a departure from previous studies, we separated our training and test set according to the host’s phylum classification instead of just randomly splitting the dataset, so that the host sequences seen on the test set are as unrelated/independent as possible from the one found on the training set. We found that, under this strict train-test data split, protein embeddings did not necessarily improve classifier performance compared to using the conventional k-mer profile representation.

2 MATERIALS AND METHODS

Shown in Figure 1 is the overview of the methodology used in this study.

2.1 Problem definition

We defined the phage-host prediction problem as taking the proteome (entire set of proteins) of both the phage and the host as input, and deciding whether the phage-host pair interacts or not. We tested various protein representations to determine the best protein representation to use as an input to classifiers when predicting phage-host interaction. We used several machine learning classifiers to predict whether the phage-host pair interacts or not.

2.2 Data collection

We collected 5,257 phage-host interaction records from VirusHostDB [25] consisting of 780 distinct hosts and 4,854 distinct phages. We also acquired 4,830 proteomes of phages from VirusHostDB [25] and 308 proteomes of bacteria from NCBI [31]. Since there are phage-host interaction records where either the phage or the host’s proteome is not accessible online, we removed those phage-host interaction records in our dataset, which resulted in our final dataset containing 3,390 phage-host interaction records. We accessed VirusHostDB [25] last January 2023 and accessed NCBI [31] datasets last February 2023.

We also collected protein sequences from the Swiss-Prot database [9], containing 570,157 protein sequences, which we accessed last June 2023. We used it for training protein language models not available online such as ProtVec [2] and Seq2Vec [18].

2.3 Train-test split and negative pairs generation

Our test set consists of phage-host interaction pairs where the host’s phylum classifications are *Actinobacteria*, *Cyanobacteria*, *Spirochaetes*, *Tenericutes*, *Bacteroidetes*, *Chlamydiae*, *Fusobacteria*, and *Deinococcus-Thermus*, while the remaining phage-host interaction pairs not found on the test set, where its host’s phylum classification are *Firmicutes* and *Proteobacteria*, were included in

the training set. The split according to the host’s phylum classification was done so that hosts found on the training set are as unrelated as possible to the hosts found on the test set. After splitting the training and test set, negative pairs of phage-host were generated independently for both sets, where we randomly paired phage-host found in its respective set, and the pair had no previously recorded interactions. Lastly, the training set was split into 4 folds using sklearn’s GroupKFold [29] and was grouped according to the host’s order classification. The GroupKFold split is done to ensure that the same group is not seen in both training and validation sets. The overview of dataset composition is visualized in Figure 2. Looking at the train-test split at the species level, the test set contains 55 distinct species, while the training folds contain 38, 58, 51, and 53 distinct species respectively from fold 1 to fold 4.

2.4 Protein representations

In this study, we mainly explored the use of protein embeddings generated by protein language models to the phage-host interaction problem. The use of protein embeddings is motivated by the advancements of large language models for natural languages, which provide representations for words in the form of vectors, often called word embeddings [3]. Similar ideas have been recently implemented for biological sequences. [17]

Protein language models that provide protein embeddings have been recently applied to different bioinformatics problems [17]. These embeddings when applied to classification tasks, have been shown to provide better results compared to using hand-crafted features [34, 36].

2.4.1 k-mer profile. The k-mer profile of a protein sequence is a vector that contains the frequencies of all the possible substrings of length k found in the sequence. Since a proteome consists of multiple protein sequences, we calculated the summation of the k-mer profiles over all the protein sequences, and did this separately for the phages and the hosts. The value of k used in this study is set to 3.

2.4.2 Statistical profile. The PredPHI classifier built by Li et al. [21] proposed a proteome representation which extracted 27 features, where 21 of those are the frequency of amino acids (20 amino acids, 1 indicating unknown amino acid), 5 of those are the abundance of each chemical component in the sequences (carbon, hydrogen, oxygen, nitrogen, and sulfur), and the last 1 indicates the sum of molecular weights of all amino acids in the sequence. These 27 protein features across the entire proteome were combined using statistical measures such as mean, standard deviation, maximum, minimum, median, and variance. The final encoded features are in the form of a matrix with a shape of $6 \times 27 \times 2$, where 6 denotes the six statistical measures (mean, std, max, min, median, var), 27 denotes the 27 protein features, and 2 represents the features for the phage and its host. For convenience, we termed this representation as *statistical profile* in this study. Lastly, we flattened the statistical profile into a vector to serve as an input to machine learning classifiers.

2.4.3 ProtVec. ProtVec [2] is a type of protein language model based on Word2Vec [26], which outputs a 100-dimensional vector embedding given a 3-mer of amino acids. To generate a ProtVec

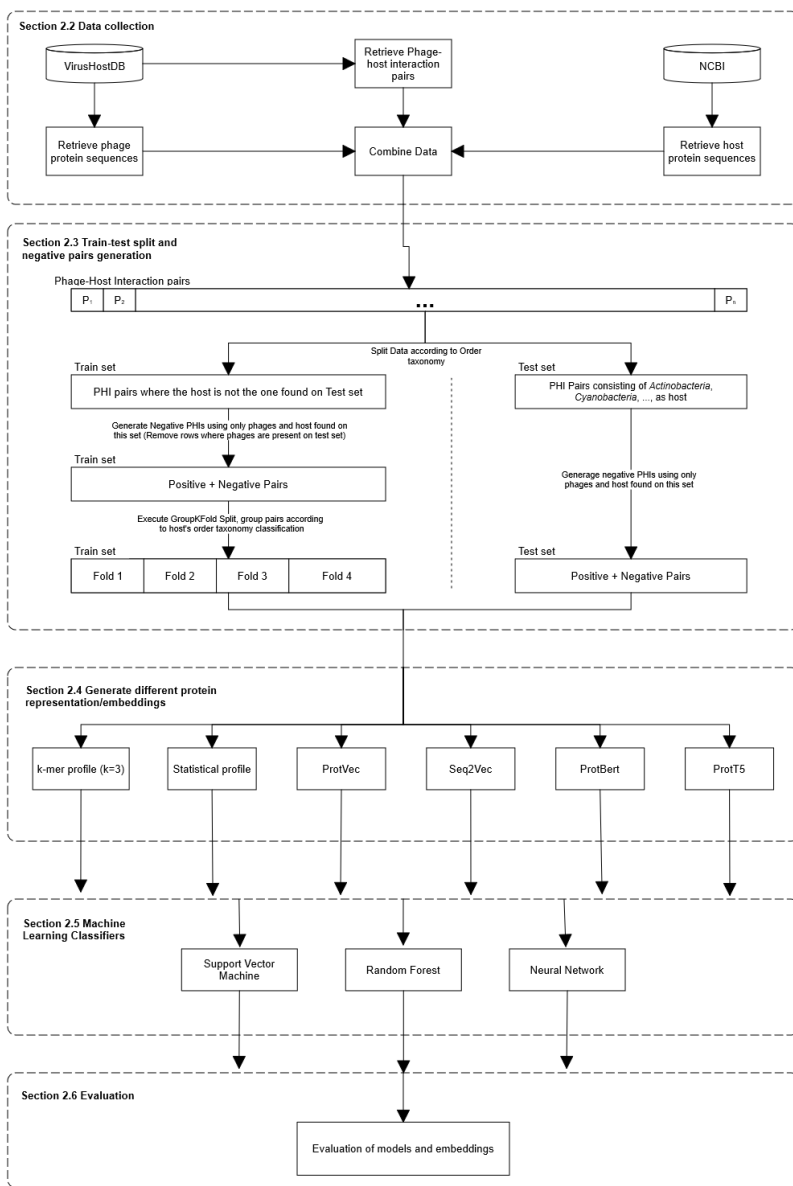


Figure 1: Flowchart summarizing the methodology of this study, which consists of 5 core components shown as dashed boxes. The first component is about collecting phage-host interaction records and protein sequences. The second component is about the train-test split method. The third component is about the generation of different protein representations. The fourth component is about the use of different machine learning classifier. And the last component is about the evaluation of protein representations and classifiers. Each component is discussed on Materials and methods section.

representation for a protein sequence, we generated ProtVec embeddings for each 3-mers that make up the protein sequence, summed up all the 3-mer embeddings element-wise, and then divided it by the number of 3-mer embeddings generated for that protein sequence. Since phages and hosts proteome consists of multiple proteins, to acquire the proteome representation, we summed up

element-wise all the ProtVec representation of each protein sequence found in the proteome, then divided it by the number of proteins found in the proteome.

The ProtVec model is not available online, so we built it by training a Word2Vec model available on the gensim python library [30]. We trained the model using the Swiss-Prot database [9], and used the hyper-parameters specified in ProtVec’s paper (*vector_size*: 100, *window*: 25, *sg*: 1).

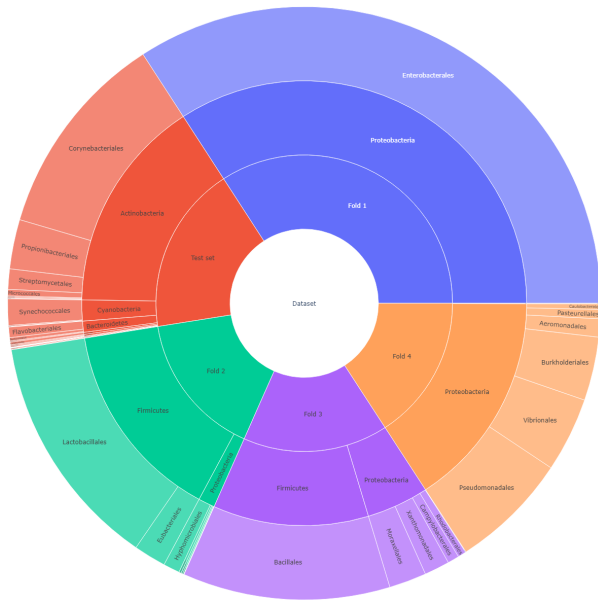


Figure 2: Sunburst chart visualizing the data for training and validation composition. The innermost pie chart visualizes the folds and the test set. The middle pie chart visualizes the hosts composition of each folds and test set at order level. The outermost pie chart visualizes the hosts composition of each folds and test set at species level.

2.4.4 Seq2Vec. Seq2Vec [18] is a type of protein embedding model based on Doc2Vec [20], where the main difference over ProtVec is the ability to generate a single embedding for the entire protein sequences regardless of its length, rather than having an embedding for each 3-mers. To acquire the proteome representation, we summed element-wise up all the embeddings of each protein sequence found in the proteome and divided it by the number of proteins found in the proteome. The Seq2Vec model is unavailable online, so we built it by training a Doc2Vec model available on the gensim python library [30]. We trained the model using the Swiss-Prot database [9], and used the hyper-parameters specified in Seq2Vec’s paper (*vector_size*: 250, *window*: 5).

2.4.5 ProtTrans. ProtTrans [10] developed several protein embedding models, two of which are used in this study namely ProtBert and Prot5-XL. Prot5-XL was considered as one of their top-performing models when tested on different tasks such as protein subcellular localization prediction and secondary structure prediction, and it is used in this study to generate embeddings for each phage. However, host protein sequences are very long and we lack computing resources to generate Prot5-XL embeddings for each host, so we used ProtBert instead to generate embeddings for each host. To acquire the final proteome representation, we summed up element-wise all the embeddings of each protein sequence found in the proteome and divided it by the number of proteins found in the proteome.

Pre-trained ProtTrans models are available on HuggingFace’s transformer library [35].

2.5 Machine learning classifiers

We used three classifiers: random forest, support vector machine, and a neural network. They take in the protein sequence representation of the phage-host pair as an input, and predict whether the phage-host pair interacts or not.

2.5.1 Random Forest. We trained a random forest classifier from sklearn [29], and used grid search to tune hyperparameters such as *n_estimators* to identify the best number of trees (the range was [25, 50, 100, 175, 250]), *criterion* to identify the best function to measure the quality of split (the range was [*gini*, *entropy*, *log_loss*]), and *max_features* to identify the best number of features to consider when looking for the best split (the range was [*sqrt*, *log2*, None]).

2.5.2 Support Vector Machine (SVM). We trained an SVM classifier using sklearn [29], and used grid search to tune hyperparameters such as *kernel* to identify the best kernel type to use (the range was [*linear*, *rbf*, *sigmoid*, *poly*]), *C* to identify the strength of the regularization (the range was [0.1, 1, 10, 50, 100]), *gamma* to identify the best value for the kernel coefficient (the range was [10, 5, 1, 0.1, 0.001], not applicable when *kernel*=*linear*), and *degree* to identify the best value for the degree of the polynomial kernel function (the range was [1,3,5], only applicable when *kernel*=*poly*).

2.5.3 Neural Network. Our neural network model accepts the combined vector representation of both the phage and the host, where the first half of the vector contains the representation of the host proteome, while the other half contains the representation of the phage proteome. The model consists of 3 fully connected hidden layers. The first layer contains 128 nodes, followed by a layer with 32 nodes, and followed by a layer with 8 nodes. The activation function used for the hidden layers is the Rectified Linear Unit (ReLU). The last layer in the neural network is a softmax layer containing 2 nodes, where the argmax function is used between the 2 nodes to predict whether an interaction between the phage-host pair is positive or not. The model was built using Pytorch [28]. We used the Adam function as our optimizer, with an initial learning rate of 0.75, which is multiplied by 0.75 every 75 epochs, to smoothen out the training loss as the number of epochs increases. We used a batch size of 256.

2.6 Evaluation

We evaluated our models on our test set. The performance metrics used to evaluate our models are accuracy, precision, recall, specificity, and F1 score.

Our data and codes are available at: <https://github.com/bioinfodlsu/phi-prediction>

3 RESULTS AND DISCUSSION

3.1 Evaluation result

The result of evaluating different protein representations as input to different machine learning classifiers is shown in Table 1. The ROC curve, together with its AUC value is shown in Figure 3.

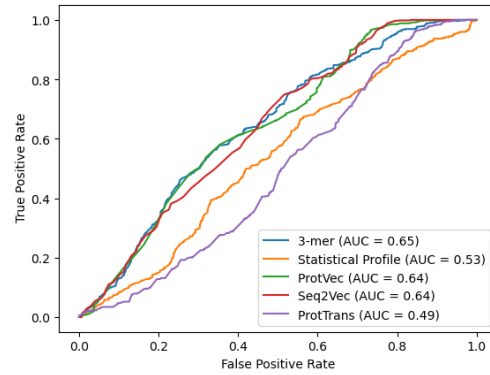
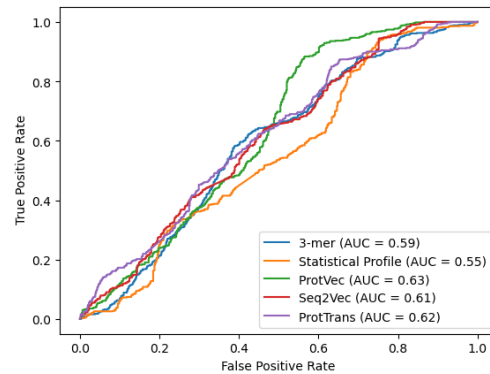
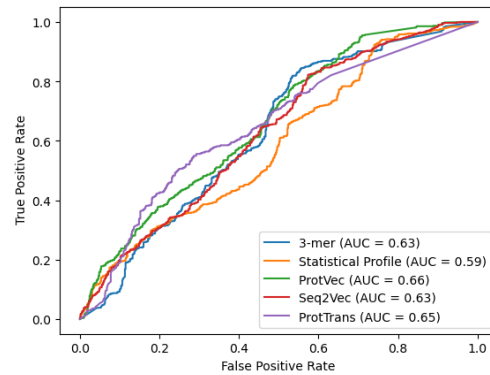
As can be observed from Table 1 and Figure 3, the different protein representations provide similar results across different classifiers (except for the statistical profile), with each representation outperforming the others in one kind of performance metric, but

Table 1: Results of evaluating different protein representations as input to random forest, SVM, and neural network classifiers, for predicting phage-host interaction.

Protein representation	Metric	Random Forest	SVM	Neural Network
3-mer	Accuracy	0.596931	0.598546	0.653473
	Precision	0.585960	0.592988	0.627346
	Recall	0.660743	0.628433	0.756058
	Specificity	0.533118	0.568659	0.550889
	F1 Score	0.621109	0.610196	0.685714
Statistics profile	Accuracy	0.492730	0.522617	0.423263
	Precision	0.484642	0.525180	0.413793
	Recall	0.229402	0.471729	0.368336
	Specificity	0.756058	0.573506	0.478191
	F1 Score	0.311404	0.497021	0.389744
ProtVec	Accuracy	0.590468	0.650242	0.627625
	Precision	0.564815	0.606651	0.586623
	Recall	0.788368	0.854604	0.864297
	Specificity	0.392569	0.445880	0.390953
	F1 Score	0.658125	0.709591	0.698890
Seq2Vec	Accuracy	0.596931	0.570275	0.608239
	Precision	0.567114	0.573854	0.573465
	Recall	0.819063	0.546042	0.844911
	Specificity	0.374798	0.594507	0.371567
	F1 Score	0.670192	0.559603	0.683214
ProtTrans (ProtBFD + Prot5)	Accuracy	0.489499	0.533118	0.627625
	Precision	0.341463	0.572438	0.589569
	Recall	0.022617	0.261712	0.840065
	Specificity	0.956381	0.804523	0.415186
	F1 Score	0.042424	0.359202	0.692871

being outperformed in another. For example, as shown in Table 1, 3-mer as input to the neural network slightly outperforms protein embeddings as input to the neural network in terms of accuracy, but as shown in Figure 3c, some protein embeddings such as ProtVec and ProtTrans slightly outperformed 3-mer in terms of AUC. Protein embeddings acquired from protein language models are shown to provide good results compared to the conventional k-mer profile representation when compared to other tasks found in bioinformatics such as predicting molecular function [34, 36]. However, protein embeddings do not necessarily improve classifier performance compared to a more conventional 3-mer profile in our phage-host interaction problem setting, possibly because protein embeddings on other tasks where it performed well are used individually as an input, whereas in this study, protein embeddings undergo averaging or mean-pooling due to the hosts and phages having multiple proteins. This averaging of embeddings may lead to the loss of information. Given our current results with the accuracy peaking only at about 65% and similarly low values of precision and recall, there is still room for improvement.

The statistical profile proposed by Li et al. [21] for their PredPHI classifier does not provide good results when used as an input to our classifiers as shown in Table 1 and Figure 3. A possible reason for the statistical profile not providing good results is that calculating

**(a) ROC Curves of different Protein representation as an input to a random forest classifier****(b) ROC Curves of different Protein representation as an input to a SVM classifier****(c) ROC Curves of different Protein representation as an input to a Neural Network classifier****Figure 3: ROC Curves of different protein representation on different classifiers.**

the statistical values of protein-derived features over a large number of protein sequences (such as the mean of amino acid 'A' frequency overall protein sequences of a phage), might not provide predictive signals for the machine learning models.

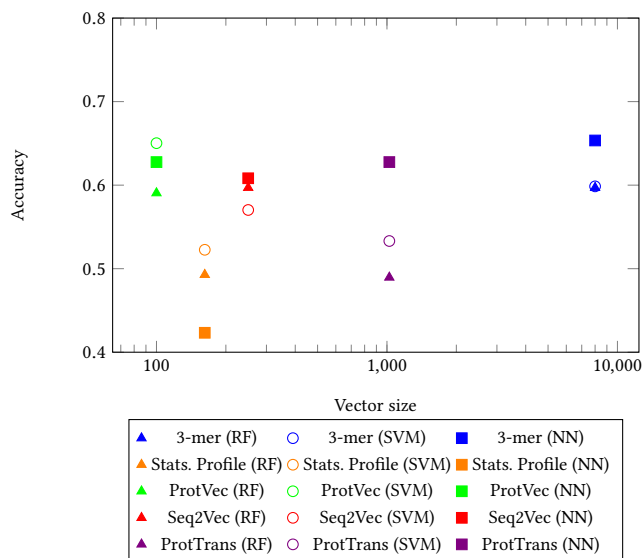


Figure 4: Visualization of the effect of different protein representation vector sizes in terms of the accuracy of classifiers for predicting phage-host interaction. RF indicates Random Forest, SVM indicates Support Vector Machine, NN indicates Neural Network

3.2 Host-specific evaluation

Shown in Table 2 is the performance of the SVM classifier using ProtVec protein embeddings as the input, for each type of host found in the test set. ProtVec embeddings and SVM were selected in this discussion since they achieved the highest accuracy when pairing a protein embedding with a machine learning model as shown in Table 1. For cases where a certain host only has a small number of recorded phage-host pairs, Young et al. [37] disregarded the hosts where the host does not have a minimum of 28 phages infecting it. However, in our current problem formulation, our classifier can still predict phages even for a host with a small number of recorded phages that infect it such as *Mycobacterium tuberculosis* with only 14 samples.

3.3 Effects of protein representation vector sizes

As can be observed from Figure 4, the vector size of protein representation does not correlate with the accuracy of the classifiers. For instance, ProtVec embedding is one of the top-performing representations even if it has the smallest vector size, outperforming the ProtTrans embedding, the largest vector size embeddings tested in this study, and even matching the performance of the 3-mer profile, which is 80 times larger in vector size. This suggests that the vector size does not directly affect the performance of the classifiers.

3.4 Future directions

Due to the large amount of proteins found on both the phage's proteome and the host's proteome, performing averaging or mean-pooling of protein representation for each proteome might lead to loss of information. It would be interesting to filter the proteins

that are related to the adsorption process, such as receptor-binding proteins of the phages or surface proteins of the hosts.

For filtering the phage's receptor-binding proteins, there has been previous work that filters the proteins based on the gene annotations [5, 12]. For cases where a gene does not have annotations, there are tools available for annotation such as Prokka [32]. There's also a recent study by Boeckaerts et al. [6], which aims to predict whether a protein is a receptor-binding protein or not, given the protein sequence. The authors proposed two approaches: the first one uses Hidden Markov Models that represent protein domains strictly related to phage RBPs, and the other one generates protein embeddings and uses them as input to a machine learning classifier.

For filtering the host's surface proteins, to the best of our knowledge, there are currently no studies that can directly identify whether a protein can be found on the bacterial surface or not. However, there are studies that can identify the subcellular localization of a given protein [1, 8, 16].

4 CONCLUSIONS

In this study, we defined the phage-host prediction problem as taking the proteome of both the phage and the host as the input, and deciding whether the phage-host pair interacts or not. We tested different protein representations and different classifiers. Protein embeddings acquired from protein language models are shown to provide good results compared to a more conventional feature such as a k-mer profile for various tasks in bioinformatics [34, 36]. However, protein embeddings did not necessarily improve classifier performance in our phage-host interaction problem setting. For future directions, it would be interesting to filter the proteins that are related to the adsorption process, such as receptor binding proteins of the phages or surface proteins of the hosts.

REFERENCES

- [1] José Juan Almagro Armenteros, Casper Kaae Sønderby, Søren Kaae Sønderby, Henrik Nielsen, and Ole Winther. 2017. DeepLoc: prediction of protein subcellular localization using deep learning. *Bioinformatics* 33, 21 (07 2017), 3387–3395. <https://doi.org/10.1093/bioinformatics/btx431> arXiv:https://academic.oup.com/bioinformatics/article-pdf/33/21/3387/50315453/bioinformatics_33_21_3387.pdf
- [2] Ehsaneddin Asgari and Mohammad R. Mofrad. 2015. Continuous distributed representation of biological sequences for deep proteomics and Genomics. *PLOS ONE* 10, 11 (2015). <https://doi.org/10.1371/journal.pone.0141287>
- [3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2014. Representation Learning: A Review and New Perspectives. arXiv:1206.5538 [cs.LG]
- [4] Juliano Bertozzi Silva, Zachary Storms, and Dominic Sauvageau. 2016. Host receptors for bacteriophage adsorption. *FEMS Microbiology Letters* 363, 4 (01 2016). <https://doi.org/10.1093/femsle/fnw002> arXiv:<https://academic.oup.com/femsle/article-pdf/363/4/fnw002/23927805/fnw002.pdf> fnw002.
- [5] Dimitri Boeckaerts, Michiel Stock, Bjorn Criel, Hans Gerstmans, Bernard De Baets, and Yves Briens. 2021. Predicting bacteriophage hosts based on sequences of annotated receptor-binding proteins. *Scientific Reports* 11, 1 (2021). <https://doi.org/10.1038/s41598-021-81063-4>
- [6] Dimitri Boeckaerts, Michiel Stock, Bernard De Baets, and Yves Briens. 2022. Identification of Phage Receptor-Binding Protein Sequences with Hidden Markov Models and an Extreme Gradient Boosting Classifier. *Viruses* 14, 6 (June 2022), 1329. <https://doi.org/10.3390/v14061329>
- [7] Yves Briens, Dimitri Boeckaerts, Michiel Stock, Celia Ferriol-González, Jesús Oteo-Iglesias, Rafael Sanjuan, Pilar Domingo-Calap, and Bernard De Baets. 2023. Actionable prediction of *Klebsiella* phage-host specificity at the subspecies level (Jul 2023). <https://doi.org/10.21203/rs.3.rs-3101607/v1>
- [8] Sebastian Briesemeister, Torsten Blum, Scott Brady, Yin Lam, Oliver Kohlbacher, and Hagit Shatkay. 2009. SherLoc2: A High-Accuracy Hybrid Method for Predicting Subcellular Localization of Proteins. *Journal of Proteome Research* 8, 11 (2009), 5363–5366. <https://doi.org/10.1021/pr900665y>

Table 2: Evaluation of SVM classifier using ProtVec protein embeddings as the input for some hosts found on our test set.

Host Name	# of samples	Accuracy	Precision	Recall	Specificity	F1 Score
<i>Leptospira noguchii serovar</i>	2	1.0000	1.0000	1.0000	1.0000	1.0000
<i>Microcystis aeruginosa</i>	2	0.5000	0.5000	1.0000	0.0000	0.6667
<i>Synechococcus sp.</i>	44	0.5227	0.4857	0.8500	0.2500	0.6182
<i>Prochlorococcus</i>	14	0.7857	0.7000	1.0000	0.5714	0.8235
<i>Prochlorococcus marinus</i>	8	1.0000	1.0000	1.0000	1.0000	1.0000
<i>Bifidobacterium asteroides</i>	4	0.7500	1.0000	0.5000	1.0000	0.6667
<i>Corynebacterium glutamicum</i>	6	0.6667	0.5000	1.0000	0.5000	0.6667
<i>Propionibacterium freudenreichii</i>	18	0.8333	0.8667	0.9286	0.5000	0.8966
<i>Cutibacterium acnes</i>	162	0.6667	0.6000	0.8961	0.4588	0.7187
<i>Mycobacterium avium</i>	2	0.5000	0.0000	0.0000	0.5000	0.0000
<i>Mycolicobacterium phlei</i>	6	0.6667	0.6667	0.6667	0.6667	0.6667
<i>Mycolicobacterium smegmatis</i>	200	0.6250	0.5870	0.8182	0.4356	0.6835
<i>Mycobacterium tuberculosis</i>	14	0.9286	0.9167	1.0000	0.6667	0.9565
<i>Rhodococcus rhodochrous</i>	2	0.5000	0.0000	0.0000	0.5000	0.0000
<i>Rhodococcus erythropolis</i>	24	0.6250	0.5714	1.0000	0.2500	0.7273
<i>Streptomyces griseus</i>	52	0.5577	0.5250	0.8400	0.2963	0.6462
<i>Streptomyces hygroscopicus</i>	2	0.5000	0.5000	1.0000	0.0000	0.6667
<i>Streptomyces scabiei</i>	8	0.7500	1.0000	0.7143	1.0000	0.8333
<i>Microbacterium paraoxydans</i>	16	0.7500	0.6667	0.8571	0.6667	0.7500
<i>Gordonia terrae</i>	352	0.6676	0.6230	0.8588	0.4743	0.7221
<i>Mycoplasma</i>	4	0.5000	0.3333	1.0000	0.3333	0.5000
<i>Rhodococcus opacus</i>	2	0.5000	0.5000	1.0000	0.0000	0.6667
<i>Rhodococcus hoagii</i>	16	0.8125	0.9091	0.8333	0.7500	0.8696
<i>Parabacteroides merdae</i>	2	0.5000	0.5000	1.0000	0.0000	0.6667
<i>Chlamydia psittaci</i>	6	0.8333	0.6667	1.0000	0.7500	0.8000
<i>Corynebacteriales</i>	6	0.6667	0.6667	1.0000	0.0000	0.8000
<i>Fusobacterium nucleatum</i>	4	0.7500	1.0000	0.5000	1.0000	0.6667
<i>Gordonia amicalis</i>	2	0.5000	0.0000	0.0000	1.0000	0.0000
<i>Flavobacterium psychrophilum</i>	30	0.6333	0.6087	0.8750	0.3571	0.7179
<i>Flavobacterium columnare</i>	4	1.0000	1.0000	1.0000	1.0000	1.0000

arXiv:<https://doi.org/10.1021/pr900665y> PMID: 19764776.

- [9] The UniProt Consortium. 2022. UniProt: the Universal Protein Knowledgebase in 2023. *Nucleic Acids Research* 51, D1 (11 2022), D523–D531. <https://doi.org/10.1093/nar/gkac1052> arXiv:<https://academic.oup.com/nar/article-pdf/51/D1/D523/48441158/gkac1052.pdf>
- [10] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghaliya Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. 2022. ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 10 (2022), 7112–7127. <https://doi.org/10.1109/TPAMI.2021.3095381>
- [11] Zhabiz Golkar, Omar Bagasra, and Donald Gene Pace. 2014. Bacteriophage therapy: a potential solution for the antibiotic resistance crisis. *The Journal of Infection in Developing Countries* 8, 02 (2014), 129–136. <https://doi.org/10.3855/jidc.3573>
- [12] Mark Edward Gonzales, Jennifer C. Ureta, and Anish M. Shrestha. 2023. Protein embeddings improve phage-host interaction prediction. *PLOS ONE* 18, 7 (Jul 2023). <https://doi.org/10.1371/journal.pone.0289030>
- [13] Fernando L. Gordillo Altamirano and Jeremy J. Barr. 2019. Phage Therapy in the Postantibiotic Era. *Clinical Microbiology Reviews* 32, 2 (2019). <https://doi.org/10.1128/cmr.00066-18>
- [14] Alison H Holmes, Luke S P Moore, Arnfinn Sundsfjord, Martin Steinbakk, Sadie Regmi, Abhilasha Karkey, Philippe J Guerin, and Laura J V Piddock. 2015. Understanding the mechanisms and drivers of antimicrobial resistance. *The Lancet* 387, 10014 (2015), 176–187. [https://doi.org/10.1016/s0140-6736\(15\)00473-0](https://doi.org/10.1016/s0140-6736(15)00473-0)
- [15] John N. Housby and Nicholas H. Mann. 2009. Phage therapy. *Drug Discovery Today* 14, 11 (2009), 536–540. <https://doi.org/10.1016/j.drudis.2009.03.006>
- [16] Annette Höglund, Pierre Dönnies, Torsten Blum, Hans-Werner Adolph, and Oliver Kohlbacher. 2006. MultiLoc: prediction of protein subcellular localization using N-terminal targeting sequences, sequence motifs and amino acid composition. *Bioinformatics* 22, 10 (01 2006), 1158–1165. <https://doi.org/10.1093/bioinformatics/btl002> arXiv:https://academic.oup.com/bioinformatics/article-pdf/22/10/1158/48838310/bioinformatics_22_10_1158.pdf
- [17] Hitoshi Iuchi, Taro Matsutani, Keisuke Yamada, Natsuki Iwano, Shunsuke Sumi, Shion Hosoda, Shitao Zhao, Tsukasa Fukunaga, and Michiaki Hamada. 2021. Representation learning applications in biological sequence analysis. *Computational and Structural Biotechnology Journal* 19 (2021), 3198–3208. <https://doi.org/10.1016/j.csbj.2021.05.039>
- [18] Dhananjay Kimothi, Akshay Soni, Praveesh Biyani, and James M. Hogan. 2016. Distributed Representations for Biological Sequence Analysis. *CoRR* abs/1608.05949 (2016). arXiv:1608.05949 <http://arxiv.org/abs/1608.05949>
- [19] Ramanan Laxminarayan, Adriano Duse, Chand Wattal, Anita K Zaidi, Heiman F Wertheim, Nithima Sumpradit, Erika Vlieghe, Gabriel Levy Hara, Ian M Gould, Herman Goossens, and et al. 2013. Antibiotic resistance—the need for global solutions. *The Lancet Infectious Diseases* 13, 12 (Nov 2013), 1057–1098. [https://doi.org/10.1016/s1473-3099\(13\)70318-9](https://doi.org/10.1016/s1473-3099(13)70318-9)
- [20] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. <https://doi.org/10.48550/ARXIV.1405.4053>
- [21] Menglu Li, Yanan Wang, Fuyi Li, Yun Zhao, Mengya Liu, Sijia Zhang, Yannan Bin, A. Ian Smith, Geoffrey I. Webb, Jian Li, Jiangning Song, and Junfeng Xia. 2021. A Deep Learning-Based Method for Identification of Bacteriophage-Host Interaction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 18, 5 (2021), 1801–1810. <https://doi.org/10.1109/TCBB.2020.3017386>
- [22] Menglu Li and Wen Zhang. 2021. PHIAF: prediction of phage-host interactions with GAN-based data augmentation and sequence-based feature fusion. *Briefings in Bioinformatics* 23, 1 (09 2021), bbab348.

- <https://doi.org/10.1093/bib/bbab348> arXiv:<https://academic.oup.com/bib/article-pdf/23/1/bbab348/42229736/bbab348.pdf>
- [23] Derek M Lin, Britt Koskella, and Henry C Lin. 2017. Phage therapy: An alternative to antibiotics in the age of multi-drug resistance. *World Journal of Gastrointestinal Pharmacology and Therapeutics* 8, 3 (2017), 162–173. <https://doi.org/10.4292/wjgpt.v8.i3.162>
- [24] Hilary D. Marston, Dennis M. Dixon, Jane M. Knisely, Tara N. Palmore, and Anthony S. Fauci. 2016. Antimicrobial Resistance. *JAMA* 316, 11 (09 2016), 1193–1204. <https://doi.org/10.1001/jama.2016.11764> arXiv:<https://jamanetwork.com/journals/jama/articlepdf/2553454/jsc160016.pdf>
- [25] Tomoko Mihara, Yosuke Nishimura, Yugo Shimizu, Hiroki Nishiyama, Genki Yoshikawa, Hideya Uehara, Pascal Hingamp, Susumu Goto, and Hiroyuki Ogata. 2016. Linking Virus Genomes with Host Taxonomy. *Viruses* 8, 3 (2016). <https://doi.org/10.3390/v8030066>
- [26] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. <https://doi.org/10.48550/ARXIV.1301.3781>
- [27] Christopher JL Murray, Kevin Shunji Ikuta, Fablina Sharara, Lucien Swetschinski, Gisela Robles Aguilar, Authia Gray, Chieh Han, Catherine Bisignano, Puja Rao, and Eve et al. Wool. 2022. Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis. *The Lancet* 399, 10325 (2022), 629–655. [https://doi.org/10.1016/s0140-6736\(21\)02724-0](https://doi.org/10.1016/s0140-6736(21)02724-0)
- [28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. arXiv:1912.01703 [cs.LG]
- [29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
- [30] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. <http://is.muni.cz/publication/884893/en>.
- [31] Eric W Sayers, Evan E Bolton, J Rodney Brister, Kathi Canese, Jessica Chan, Donald C Comeau, Ryan Connor, Kathryn Funk, Chris Kelly, Sunghwan Kim, and et al. 2021. Database Resources of the National Center for Biotechnology Information. *Nucleic Acids Research* 50, D1 (2021). <https://doi.org/10.1093/nar/gkab1112>
- [32] Torsten Seemann. 2014. Prokka: rapid prokaryotic genome annotation. *Bioinformatics* 30, 14 (03 2014), 2068–2069. <https://doi.org/10.1093/bioinformatics/btu153> arXiv:https://academic.oup.com/bioinformatics/article-pdf/30/14/2068/48924770/bioinformatics_30_14_2068.pdf
- [33] Saptarshi Sinha, Rajdeep K. Grewal, and Soumen Roy. 2018. Chapter Three - Modeling Bacteria-Phage Interactions and Its Implications for Phage Therapy. *Advances in Applied Microbiology*, Vol. 103. Academic Press, 103–141. <https://doi.org/10.1016/bs.aambs.2018.01.005>
- [34] Amelia Villegas-Morcillo, Stavros Makrodimitris, Roeland C H J van Ham, Angel M Gomez, Victoria Sanchez, and Marcel J T Reinders. 2020. Unsupervised protein embeddings outperform hand-crafted sequence and structure features at predicting molecular function. *Bioinformatics* 37, 2 (08 2020), 162–170. <https://doi.org/10.1093/bioinformatics/btaa701> arXiv:https://academic.oup.com/bioinformatics/article-pdf/37/2/162/50322153/btaa701_supplementary_data.pdf
- [35] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing. arXiv:1910.03771 [cs.CL]
- [36] Kevin K Yang, Zachary Wu, Claire N Bedbrook, and Frances H Arnold. 2018. Learned protein embeddings for machine learning. *Bioinformatics* 34, 15 (03 2018), 2642–2648. <https://doi.org/10.1093/bioinformatics/bty178> arXiv:https://academic.oup.com/bioinformatics/article-pdf/34/15/2642/48935464/bioinformatics_34_15_2642.pdf
- [37] Francesca Young, Simon Rogers, and David L. Robertson. 2020. Predicting host taxonomic information from viral genomes: A comparison of feature representations. *PLOS Computational Biology* 16, 5 (05 2020), 1–24. <https://doi.org/10.1371/journal.pcbi.1007894>

Coronavirus Herd Immunity Optimizer with Tent and Circle Chaotic Maps for the p -Center Facility Location Problem: The Case of Davao City, Digos City and Tagum City, Philippines

Hanna Mae Lina Limpag
University of the Philippines Mindanao
Davao City, Philippines
hllimpag@up.edu.ph

Ritchie Mae T. Gamot
University of the Philippines Mindanao
Davao City, Philippines
rtgamot@up.edu.ph

ABSTRACT

The Coronavirus Herd Immunity Optimizer (CHIO) is a relatively new nature-inspired algorithm that is inspired by the concept of herd immunity. The herd or population is said to be gradually affected by three different cases: the infected, susceptible, and immune cases. Improvements in the status of everyone in the herd is influenced by the algorithmic parameter, called the reproduction rate, or the rate at which a virus spreads. Performance of CHIO using benchmark optimization functions and comparison to well-known swarm-based algorithms have shown a good balance between exploration and exploitation. As CHIO is still in its infancy, this study attempted to use the original algorithm and its modified version to solve an NP-Hard problem, called the p -center problem. The modification was a simple addition of chaotic maps, tent and circle chaotic maps, to the original CHIO algorithm. Experimental and statistical results using household data sets from the Davao Region (Davao City, Tagum City and Digos City) showed that both CHIO and CHIO with chaotic maps can minimize the p -center problem with the former showing better results in most cases. Reproduction rate values were also varied, and results revealed that a higher basic reproductive of (BRr = 0.90) performed better than lower basic reproductive rate (BRr = 0.01).

KEYWORDS

Metaheuristics, Nature-inspired optimization, Facility location problem, p -center problem

*Article Title Footnote needs to be captured as Title Note

†Author Footnote to be captured as Author Note

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

PCSC2024, May 2024, Laguna, Philippines

© 2024 Copyright held by the owner/author(s).

<https://doi.org/10.1145/1234567890>

ACM Reference format:

FirstName Surname, FirstName Surname and FirstName Surname. 2024. Coronavirus Herd Immunity Optimizer with Chaotic Maps for the p -center Facility Location Problem: The case of Davao City, Digos City and Tagum City, Philippines. In *Proceedings of Philippine Computing Science Congress (PCSC2024)*. Laguna, Philippines, 8 pages.

1 Introduction

Facility location decisions show a prominent role in strategic planning of many firms, companies, and governmental organizations [1] as it involves the optimal placement of services such as education, health, commerce and many more. Such type of decision fall under the p -center minimax location-allocation problem.

The p -center problem is the location of facilities where the maximum distance between the nearest facility and its demand point is minimized [2]. Despite its simplicity and even in small-scale, the p -center problem is considered an NP-Hard problem [3].

The p -center in Euclidean space is in Equation 1:

$$\min_{(x_i, y_i)} \max_i \min_j [(a_i - x_i)^2 + (b_i - y_i)^2] \quad (1)$$

where (a_i, b_i) are the coordinates of the i th demand point and (x_i, y_i) are the coordinates of the i th facility. Deterministic and metaheuristic methods have been used in the past to solve the p -center problem.

The p -center problem was solved by Gaar and Sinnl [4] on a new integer programming formulation by means of branch-and-cut where cuts for customer demands points are iteratively generated. In the same study, they generated a way to use lower bound information to obtain stronger cuts which are at par with best known in literature. Computational studies with up to more than 700,000 customers and locations showed that, for many instances, the algorithm was competitive. Gaar and Sinnl [4] suggested hybridization techniques to alleviate time consuming calculations for some large-scale instances. Contardo, et al [5], on the other hand, solved the vertex p -center problem by proposing a row generation algorithm that iteratively solves smaller subproblems using subsets of the clients. Variable number of points are added in this subset at the end of each iteration. Computational experiments on up to 1 million clients and centers showed its scalability. Small values of p were solved to optimality whilst instances greater than 10^4 were solved faster than expected. Nematian and Sadati [6] introduced a p -center problem with demands treated as fuzzy random variables. The

problem was converted to a deterministic integer programming problem to be solved by new methods based on implementation of possibility theory and fuzzy random chance-constrained programming. Using the location of bicycle stations in Tabriz City, Iran, their method showed robustness and applicability to real cases with uncertainty.

Anonymous [7] solved the Euclidean p -center problem using a nature-inspired optimization algorithm called the Stud Krill Herd Algorithm (SKHA). It was applied, for the first time, to the allocation of early warning devices (EWD) in Digos City, Philippines. Algorithmic parameters were tested and showed the capacity of SKH to find possible best locations of EWDs for $p = 5, 6$ and 7 . Yin et al [8] used a greedy randomized solution and tabu search technique to solve the vertex p -center problem. The resulting solution is combined with one of the elite solutions by path-relinking, which consists in exploring trajectories that connect high-quality solutions. Their method showed competitive edge over state-of-the-art algorithms in literature both in solution quality and computational efficiency. Anonymous et al [9] also used the Moth-Flame Optimization Algorithm (MFOA) and Whale Optimization Algorithm (WOA), for the first time, on the Euclidean p -center problem. Experimental results using the Digos City, Philippines data alongside tests on algorithmic parameters showed the ability of both algorithms to find better locations for $p=5, 6$ and 7 as iterations progress. Between the two, MFOA generally showed better solution quality over WOA.

In this study, a relatively new nature-inspired optimization algorithm called the Community Herd Immunity Optimizer (CHIO), attempts to solve the p -center problem, for the first time, using household data locations of Digos City, Davao City and Tagum City, Philippines. Furthermore, chaotic maps were introduced to determine how the algorithm behaves from the traditional randomized generation initial solution generation and fatality cases. This study aims to contribute to alternative approaches to the Euclidean p -center problem.

2 Community Herd Immunity Optimizer

Herd immunity is the indirect protection from an infectious disease that occurs when a population has established immunity to disease through vaccination or previous infection. CHIO is a relatively new nature-inspired optimization algorithm inspired by herd immunity developed by Al-Betar et al [10]. This algorithm aims to provide solution, $x = (x_1, x_2, \dots, x_n)$, to the minimization/maximization problem $f(x)$, where n is the total number of genes per individual.

The seminal paper on CHIO compared the algorithm to seven well-known swarm-based algorithms [10]. Using 23 benchmarks functions. CHIO was able to obtain 16 out of 23 new results compared to the other methods. It was also used to solve real-world engineering optimization problems and compared with nine other set of methods. CHIO was also proven to be generally competitive in these kinds of problems. It is important to note that CHIO had a variation called Multi-Objective CHIO, or MOCHIO [11], to optimize the design of a brushless direct

current motor in the domain of magnetics. Results show that MOCHIO appears to be viable and dominant.

The implementation procedure algorithm:

1. Initialize algorithmic and control parameters to include the number of initial infected cases (C_0), maximum number of iterations (max_iter), population size or total number of cases/individuals (HIS) and number of genes per individual, (n), basic reproduction rate (BR_r) and maximum age of infected cases (max_age).
2. Generate the herd immunity population called HIP (Equation 2) of size HIP with dimension n . Each element in the matrix is generated randomly within the range of the lower bound and upper bound of the decision variables. Additionally, a one-dimensional status vector, S , of size HIS is initialized to 0. The S vector later will contain the values 0 (susceptible case), 1 (infected case) or 2 (immuned case). Another one-dimensional vector, A , of size HIS , called the age vector is also initialized to zero.

$$HIP = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_n^1 \\ x_1^2 & x_2^2 & \dots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{HIS} & x_2^{HIS} & \dots & x_n^{HIS} \end{bmatrix} \quad (2)$$

3. Evolve the population based on Equation 3. Depending on a generated random number, r , and the BR_r , each individual will either be updated using Equations 4, 5, or 6. The indices c , m and v are randomly chosen cases from the infected, susceptible and immune case, respectively.

$$x_i^j(t+1) = \begin{cases} C(x_i^j(t)) & r \in [0, 1/3 BR_r) \\ N(x_i^j(t)) & r \in [1/3 BR_r, 2/3 BR_r) \\ R(x_i^j(t)) & r \in [2/3 BR_r, BR_r) \\ x_i^j(t) & r \in [BR_r, 1) \end{cases} \quad (3)$$

$$C(x_i^j(t)) = x_i^j(t) + r \times (x_i^j(t) - x_i^c(t)) \quad (4)$$

$$N(x_i^j(t)) = x_i^j(t) + r \times (x_i^j(t) - x_i^m(t)) \quad (5)$$

$$R(x_i^j(t)) = x_i^j(t) + r \times (x_i^j(t) - x_i^v(t)) \quad (6)$$

4. Update the status and age vectors using Equation 7. The current solution will only be replaced by a new better solution (better immunity rate) else the age vector for this solution/case is incremented by 1.

$$(S_j, A_j) = \begin{cases} (1,1) & \text{if Equation 8} \\ (2,0) & \text{if Equation 9} \end{cases} \quad (7)$$

$$f(x^j(t+1)) < \frac{f(x^j(t+1))}{\Delta f(x)} \text{ and } S_j = 0 \text{ and is_Corona}(x^j(t+1)) \quad (8)$$

$$f(x^j(t+1)) > \frac{f(x^j(t+1))}{\Delta f(x)} \text{ and } S_j = 1 \quad (9)$$

$$\begin{aligned} & \text{is_Corona}(x^j(t+1)) \\ & = \begin{cases} 1 & \text{if new case is inherited from infected case} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (10)$$

5. Update fatality cases by regenerating a new random case if $A_j > max_age$ and $S_j = 1$. The status and age values for this case is set to 0 and 1, respectively.
6. The algorithm iterates steps 3 to 5 until max_iter iterations are reached.

For a more helpful understanding, these six primary steps are also represented as a flowchart using Figure 1.

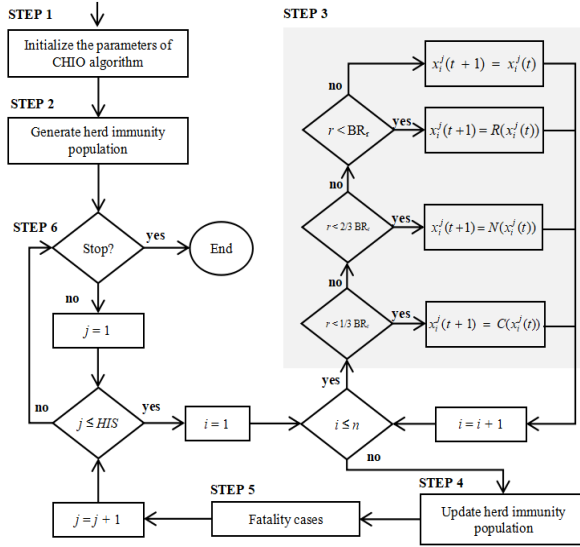


Figure 1: Flowchart of the Coronavirus Herd Immunity Optimizer (Lifted from Al-Betar et.al, 2020).

3 Chaotic Maps

Chaotic systems are rule-based systems with mathematically accessible input-output relationships. The output behaves randomly yet adhering to specific analytically grounded relationships. Its output is confined to two values making them suited for usage as a random factor to weaken particle movement paths and improve their exploring capabilities [12]. Studies in the past proved to show positive impact of using chaotic maps in optimization algorithms ([13],[14]).

In this study, two types of chaotic maps that have improved previous studies will be presented and used.

3.1 Tent Chaotic Map

The Tent Chaotic Map is defined in Equation 11 where x_k is a random value from 0 to 1 and μ is a value chosen between 0 to 2. The values from this chaotic map ranges from 0 to 1 and tends to generate a tent-like shape of the graph.

$$x_{k+1} = \begin{cases} \mu x_k & x_k < 0.5 \\ \mu(1 - x_k) & x_k \geq 0.5 \end{cases} \quad (11)$$

A study by Demidova and Gorchakov [14] comparing tent chaotic map to non-chaotic pseudorandom number generators showed that the former can yield more symmetrically and uniformly distributed real numbers. The tent chaotic map was used in place of the traditional pseudorandom number generators in implementing the Fish School Search Algorithm as tested on benchmark test functions. With the incorporation of exponential step into the algorithm, accuracy was improved and was proven

to be more effective compared to Particle Swarm Optimization and Genetic Algorithms.

3.2 Circle Chaotic Map

The Circle chaotic map or the sine circle map, as seen in Equation 12, is an iterated one-dimensional map [15]. The value x_k is a random number between 0 and 1 and a and b are the external applied frequency and strength of nonlinearity, respectively.

$$x_{k+1} = x_k + b - \frac{a}{2\pi} \sin(2\pi x_k) \text{mod}(1) \quad (12)$$

The same study by Demidova and Gorcothakov [14] showed that the circle chaotic map was competitive compared to other chaotic maps. It was, however, less superior than the tent chaotic map.

With the potential of CHIO in solving optimization problems and the proven effectiveness of chaotic maps to improve existing algorithms, this study investigated using the classical CHIO (CPC) and CHIO with Chaotic Maps (CCMPC) to solve the p -center problem.

4 CPC and CCMPC for the p -center problem

This study focuses mainly on, one, the performance of CHIO on p -center problems and, two, the effect of replacing the pseudo random number generators for the initial population generation and generation of fatality cases using tent and circle chaotic maps into CHIO. The classic CHIO will be labelled as CPC while any of the two CHIO with chaotic maps will be labelled as CCMPC.

4.1 Test Data and Parameter Settings

4.1.1 Datasets. The algorithms will be tested using datasets from the cities of Davao region in the Philippines. The Davao City dataset was obtained from the Phil-LIDAR2 project of the Anonymous (details omitted for double-blind reviewing) and has 186,244 household data. The Digos City dataset was lifted from the Phil-LIDAR1 project of the same university with a total number of 23,882 household data. Lastly, the geotagged dataset of Tagum City that was obtained from the City Planning Development Office of Tagum City with 60,000 household data.

In the interest of time and limited computing resources during the conduct of the study, the study utilized a sample of 30% of the total household data for each city. Thus, the number of household data used in the study for Davao, Digos, and Tagum datasets were 55876, 7165, and 18074, respectively.

4.1.2 Parameter Settings. Presented in Table 1 are the parameter settings including the algorithmic parameters for CHIO, and the parameters used for the chaotic maps. Furthermore, two values of the basic reproduction rate parameter were also experimented on when dealing with slow ($BR_r = 0.01$) and fast-spreading ($BR_r = 0.90$) infections. All in all, there are 18 different parameter configurations considering 3 data sets (Davao, Digos, and Tagum), algorithm versions (Classic Chio, Chio with Circle Chaotic Map and CHIO with Tent Chaotic Map) and basic reproductive rate (0.01 and 0.90). Each

parameter configuration is independently run 30 times to allow for statistical analysis.

4.2 Experimental Results

The goodness of the performance of CPC and CCMPC is initially gauged by the quality of the fitness values using Equation 1 (descriptive and inferential statistics), and then checks the algorithm performance based on CPU Time (time at which the algorithm first found the best solution) as a secondary criterion.

Table 1. Experimental parameters.

Parameter	Value
Basic reproduction rate (BRr)	0.01 and 0.90
Maximum age for infected cases (max_age)	100
Initial infected cases (C_0)	1
Maximum number of iteration (max_itr)	30000
Total number of solution (HIS)	30
Number of facilities/centers (n)	6
μ (Tent map)	2
a (Circle map)	0.5
b (Circle map)	0.2

4.2.1 *Basic reproduction rate 0.01.* Results for the different datasets and algorithms on using BRr = 0.01 are presented in this section. Statistical analyses were done on the fitness values, and results show that no significant difference was observed between the Classic CHIO (CC) and CHIO with Tent Chaotic map (CTC) for the Davao City dataset. Figure 2 (A and B) shows the convergence of the fitness values for these two algorithms as the iteration progresses. There is a drastic decrease/improvement in the fitness values on its first 10000 iterations and has been gradually declining upon further iterations. It is also important to note that improvements are still visible from iteration 25,000 onwards, where a possible decrease in the fitness value may still be achieved if iterations are to be increased.

Table 2 also shows the descriptive statistics associated with this experiment as well as the time at which the best fitness value was first observed (CPU time). Classic CHIO obtained the most desirable solution terms of best, worst, mean fitness values and CPU time. CHIO with Circle Map, however, generated very similar (stable) fitness values as shown by the lowest standard deviation value.

Graphs of the best performing algorithms in each dataset, showing the initial and final position of the centers were overlaid on Google earth. The radius of the circles indicates the fitness value of the solution generated. Final locations tend to be closer to the actual households of the city and are more within the boundary of the region shows a more desirable result. Figure 2 (A and B) shows that the radii of the red circles are smaller than the green circles, indicating improvement on the solutions generated by the algorithm as it progresses.

Statistical analysis for the different algorithms on the Digos City dataset showed that Classic CHIO was significantly better than the two CCMPCs while Classic CHIO and CHIO with Circle

chaotic map were not significantly different from each other when tested on the Tagum City dataset.

Table 2. Best, worst, mean, and standard deviation for Davao City dataset using BRr=0.01 in meters.

	Classic CHIO	CHIO w/ Tent	CHIO w/ Circle
Best Fitness [BF]	8426.3650	8949.5450	8790.3260
Worst Fitness [WF]	12941.3500	14457.9340	13728.8590
Mean Fitness [MF]	9429.0290	9695.9080	9911.0030
Standard Deviation [SD]	516.7280	492.1120	424.5360
Mean CPU Time [MCT] (s)	116406.0644	117120.9388	124436.3418

Value/s in bold character indicates the best value among the setup.

Tables 3 and 4 show the descriptive statistics of the different algorithms for Digos City and Tagum City, respectively. For the Digos City dataset, the Classic CHIO obtained the most desirable solution in terms of best, worst, and mean fitness values. CHIO with Tent Chaotic Map, however, generated very stable fitness values as well as faster CPU time. For the Tagum City dataset, on the other hand, the Classic CHIO was able to find the best fitness value and best mean fitness value. CHIO with Circle Chaotic Map was the most stable as shown by the least standard deviation and the quickest to finish among the three.

Table 3. Best, worst, mean, and standard deviation for Digos City dataset using BRr=0.01 in meters.

	Classic CHIO	CHIO w/ Tent	CHIO w/ Circle
BF	2051.7720	2299.6306	2370.4583
WF	3393.7780	3921.0970	5092.3250
MF	2451.7308	2601.8463	2660.9530
SD	151.7184	137.1814	153.01201
MCT	15498.9406	13063.4300	16704.6842

Value/s in bold character indicates the best value among the setup.

Table 4. Best, worst, mean, and standard deviation for Tagum City dataset using BRr=0.01 in meters.

	Classic CHIO	CHIO w/ Tent	CHIO w/ Circle
BF	5007.1218	5166.2807	5142.2310
WF	7586.5770	8169.4670	9426.3870
MF	5619.5590	5793.7950	5757.1058
SD	220.5652	273.1929	259.5690
MCT	36112.5090	38435.4468	33713.8444

Value/s in bold character indicates the best value among the setup.

Shown in Figure 2 (C) and Figure 2 (D and E) are the convergence graph and the initial and final positions for the best performing algorithms for Digos and Tagum datasets. Convergence behavior of the fitness values for the different algorithms as iterations progressed for Digos and Tagum City dataset were similar to that of the Davao City these graphs. It emphasizes that the algorithm can improve solutions even with a different basic reproduction rate, and improvements were evident by a tighter (smaller) radius of *p-centers* at the end of the iteration.

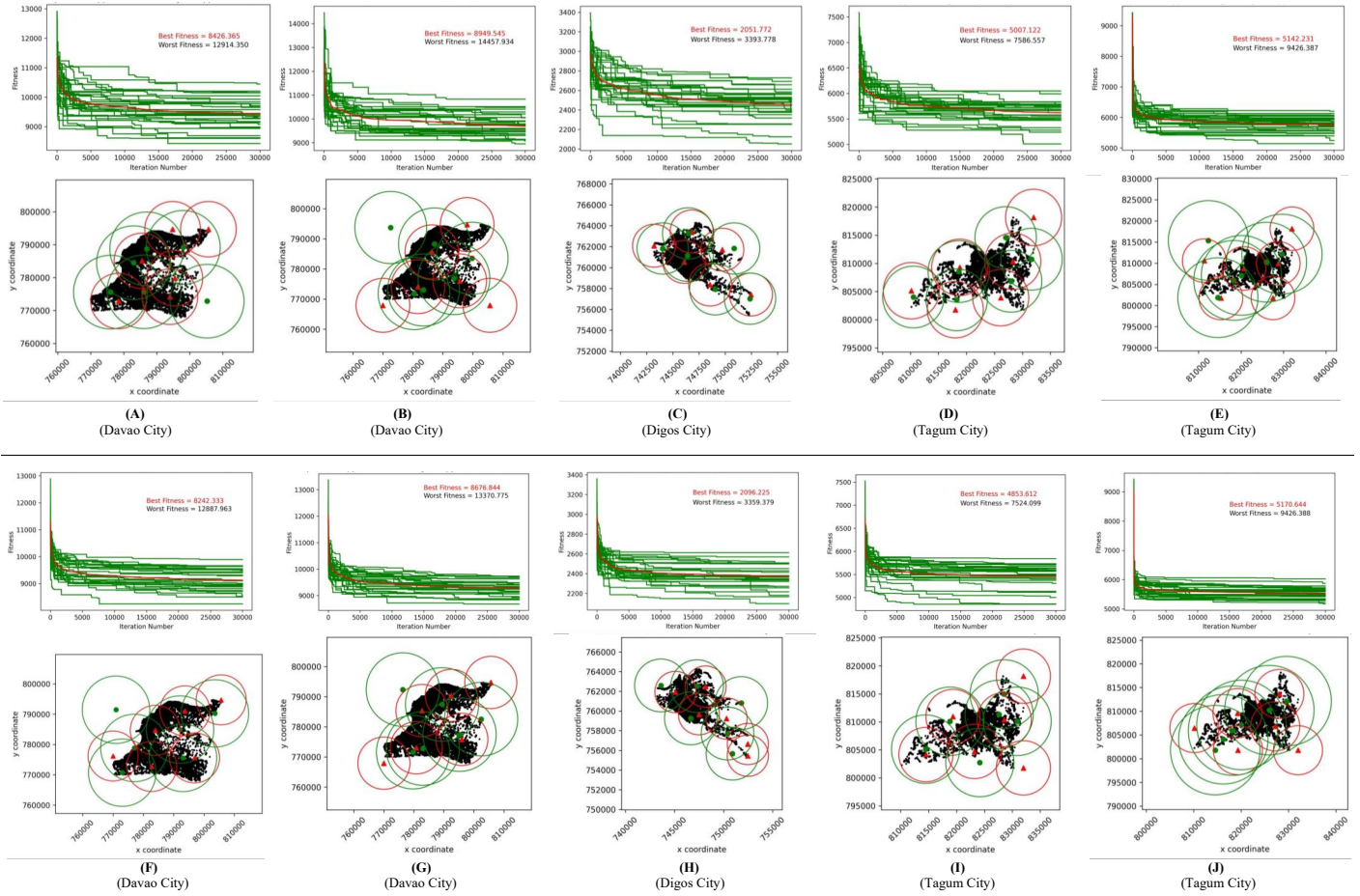


Figure 2: Convergence Graph(top) and Initial [green] and Final [red] (bottom) locations of the best solution/s using BR, = 0.01 (A through E) and BR, = 0.90 (F through J).

4.2.2 *Basic reproduction rate 0.90.* Results for the different datasets and algorithms on using $BRr = 0.90$ are presented in this section. It is worth noting that the best algorithms for a particular dataset and behaviour of convergence maps for this BRr had the exact similar results as with $BRr = 0.01$. The descriptive statistics resulting from $BRr = 0.90$, however, were different from that of $BRr=0.01$.

Tables 5, 6 and 7 show the descriptive statistics of the different algorithms for Davao City, Digos City and Tagum City, respectively. For the Davao City dataset, Classic CHIO acquired the most desirable value for the best and mean fitness, while CHIO with Tent Chaotic Map has the lowest worst fitness value. CHIO with Circle Chaotic Map, however, has least deviated fitness and the fastest to converge to the best solution. For the Digos City dataset, Classic CHIO obtained the most appealing results for the best, worst, and mean fitness values, as well as the most stable fitness as depicted from it having the lowest standard deviation among the three. All while, CHIO with Circle Chaotic Map is the quickest to arrive with the best solution. Lastly, for the Tagum City dataset, the best fitness value among the algorithms was acquired by CHIO with Tent Chaotic Map as well as the quickest to arrive at the best solution. While Classic CHIO has the best result for the values of the worst and mean fitness. CHIO with Circle Chaotic Map, however, is the most stable having the least deviated fitness values among the algorithms.

Statistical analysis for the different algorithms using $BRr=0.90$ for the different datasets produced similar results as that of the $BRr=0.01$. That is, Classic CHIO and CHIO with Tent Chaotic Map for the Davao City dataset; Classic CHIO for the Digos City dataset and Classic CHIO and CHIO with Circle Chaotic Map for the Tagum City dataset.

Plots of the solutions of the best performing algorithms for each dataset and its corresponding graphs are presented in Figure 2 (F through J). Convergence graphs for all these figures showed similar in behaviour as observed with $BRr=0.01$ where clear decrease in convergence trend can be observed. Plots of the p -centers also show improvement with the final (red) locations having tighter radii than the initial (green) locations. Again, it is to note that some final locations were in a body of water as seen in Figure 2 (F through J).

Table 5. Best, worst, mean, and standard deviation for Davao City dataset using $BRr=0.90$.

	Classic CHIO	CHIO w/ Tent	CHIO w/ Circle
	BF	8242.3331	8676.8444
WF	9893.9472	9734.0202	9893.1809
MF	9125.2997	9291.7594	9406.9496
SD	412.7521	292.3905	291.1344
MCT	129340.2153	117904.6340	93334.1110

Value/s in bold character indicates the best value among the setup.

Table 6. Best, worst, mean, and standard deviation for Digos City dataset using $BRr=0.90$.

	Classic CHIO	CHIO w/ Tent	CHIO w/ Circle
	BF	2096.2230	2153.8885
WF	3359.3790	3925.4880	5092.3180
MF	2373.0290	2541.3992	2586.2832
SD	123.0010	178.1307	152.7004
MCT	15309.2362	12273.7748	11485.1806

Value/s in bold character indicates the best value among the setup.

Table 7. Best, worst, mean, and standard deviation for Tagum City dataset using $BRr=0.01$.

	Classic CHIO	CHIO w/ Tent	CHIO w/ Circle
	BF	4853.6119	4811.3694
WF	7524.0990	8061.8010	9426.3880
MF	5460.7795	5645.6838	5540.4572
SD	247.0989	264.1345	199.0190
MCT	37433.6903	30490.1096	32811.7169

Value/s in bold character indicates the best value among the setup.

4.2.3 *BRr 0.01 vs BRr 0.90.* For better understanding of the comparisons on this section of the results, nine experimental setups were defined. Classic CHIO algorithm was used for the setups S1, S2, and S3, using Davao, and Tagum City datasets, respectively. Additionally, CHIO with Tent Chaotic Map, and with Circle Chaotic Map, were used for S4 through S6, and S7 through S9, respectively using the same dataset order as classic CHIO.

Corresponding best fitness values were recorded for each of the experimental setup and is presented using Table 9 for the different basic reproduction rates. It can be observed that 7 of the 9 setups performed better when the basic reproduction rate is 0.90.

Table 10 shows the summary of the statistical result using the best fitness values for all the algorithms. At a significance level of 0.05, 8 of the 9 setups have enough reason to believe that there is a significant difference between the two basic reproduction rates in terms of fitness value.

Table 9. Best fitness values and Mean Ranks using basic reproduction rates 0.01 and 0.90.

Setup	Fitness Value		Mean Rank	
	BRr = 0.01	BRr = 0.90	BRr = 0.01	BRr = 0.90
S1	8426.3654	8242.3331	28.5000	32.5000
S2	2051.7720	2096.2250	31.1000	29.9000
S3	5007.0532	4853.6119	29.7300	31.2700
S4	8949.5453	8676.8444	31.8300	29.1700
S5	2299.6306	2153.8885	30.8300	29.1400
S6	5166.2807	4811.3694	35.7700	25.2300
S7	8790.3264	8690.9238	36.7000	24.3000
S8	2370.4583	2252.4427	37.9300	30.2700
S9	5142.2311	5170.6444	30.8000	30.2000

Value/s in bold character indicates better value for fitness and mean rank comparing two basic reproduction rates.

Table 10. Mann-Whitney U-test result using at alpha = 0.05.

Setup (0.01 vs 0.90)	<i>p</i> -value
S1	0.018737
S2	0.011710
S3	0.018005
S4	0.001030
S5	0.074779
S6	0.000604
S7	0.000604
S8	0.000604
S9	0.000604

Value/s in bold character indicates significant difference.

Furthermore, the mean ranks of each comparison are also recorded and are also presented in Table 9. It can be observed that best fitness values using the basic reproduction rate value 0.90 performs better compared to basic reproduction rate value 0.01.

Based on the descriptive and inferential statistical evidence, using basic reproduction rate value of 0.90 provides better results. A larger basic reproduction rate (BRr) means that the generated solutions underwent frequent changes (refer to Equation 3), providing a larger solution space to explore.

5 Conclusion

CHIO showed a good balance between exploration and exploitation as concluded by Al-Betar et.al. With this and the algorithm being new, it presented itself as a good area for research to further explore its capabilities, hence, being used in this study to solve the *p-center* problem.

When tested, Classic CHIO resulted in improvement on the fitness values of the solution generated, indicating that the population has been acquiring herd immunity. This means the effectiveness of CHIO in solving the *p-center* problem.

Overall, any of the algorithms (CPC and CCMPC) are effective for all tested datasets. From which can be concluded from the statistical results on Davao and Tagum City showing no significant difference between algorithms. However, CPC emerged to be one if not the lone best performing algorithm for all three datasets on all BRr experimental values.

The value of the BRr also affects how the algorithms search the solution space. The larger BRr, in this case 0.90, the better algorithm performs compared to 0.01 with respect to the best fitness. The increased rate of exploring the solution space could have affected the results.

Further investigation on parameters used, and feasibility of the location of the centers still calls for improvement. The resulting location of the centers can also be subjected to land use land cover of each city to determine viability of putting up centers in those areas. This can also include elevation, nearness to highway, among others. The authors are also currently investigating the algorithm using full household data set used in this study.

The extent of the capabilities of this algorithm are still to be tested and can also be used to solve other optimization problems.

However, the methodology used in this study can already aid local government units in finding best initial locations for services such as emergency centers, early warning devices, fire stations, bicycle stations, and others.

Nevertheless, CHIO was found to be very capable and can be considered as an alternative to state-of-the-art methods in solving the *p-center* problem.

ACKNOWLEDGMENTS

The authors wish to acknowledge Anonymous for providing invaluable computational resources. The authors also wish to extend their sincerest gratitude to the ff: Anonymous for allowing the use of the real household datasets. *Details omitted for double-blind reviewing.*

REFERENCES

- [1] M Taghavi and H Shavandi (2012). The P-center Problem under Uncertainty. *Journal of Industrial and Systems Engineering*. 6 (1), 48-57.
- [2] A Kaveh and H Nasr (2011). Solving the conditional and unconditional p-center problem with modified harmony search: A real case study. *Scientia Iranica*. 18 (4), 867-877.
- [3] H Fayed and A Atiya (2012). A mixed breadth-depth first strategy for the branch and bound tree of Euclidean k-center problems. *Computational Optimization and Applications*. 54, 675-703.
- [4] E Gaar and M Sinnl (2022). A scaleable projection-based branch-and-cut algorithm for the p-center problem. *European Journal of Operations Research*. 303, 78-98.
- [5] C Contardo, M Iori and R Kramer (2019). A scaleable exact algorithm for the vertex p-center problem. *Computers and Operations Research*. 103, 211-220.
- [6] J Nematian and M Sadati (2015). New methods for solving a vertex p-center problem with uncertain demand-weighted distance: A real case study. *International Journal of Industrial Engineering Computations*. 6 (2), 253-266.
- [7] Anonymous (2019). Details omitted for double-blind reviewing.
- [8] AH Yin, TQ Zhou, JW Ding, QJ Zhao and ZP Lv (2017). Greedy Randomized Adaptive Search Procedure with Path-Relinking for the Vertex p-center problem. *Journal of Computer Science and Technology*. 32, 1319-1334.
- [9] Anonymous (2022). Details omitted for double-blind reviewing.
- [10] MA Al-Betar, ZAA Alyasseri and MA Awadallah (2020). Coronavirus Herd Immunity Optimizer. *Neural Computing and Application*. 33, 5011-5042.
- [11] C Kumar and T Gunasekar (2021). MOCHIO: A Novel Multi-Objective Coronavirus Herd Immunity Optimization Algorithm for Solving Brushless Direct Current Wheel Motor Design Optimization Problem. *Journal for Control, Measurement, Electronics, Computing and Communications*. 63 (1), 149-170.
- [12] M Mobaraki, R Boostani and M Sabeti (2020). Chaotic-based Particle Swarm Optimization with inertia weight for optimization tasks. *Journal of AI and Data Mining*. 8 (3), 303-312.
- [13] H Lu, X Wang, Z Fei, and M Qiu (2014). The Effects of Using Chaotic Map on Improving the Performance of Multi Objective Evolutionary Algorithms. *Mathematical Problems in Engineering*, 2014, 1-16.
- [14] LA Demidova and AV Gorchakov (2020). A study of Chaotic Maps Producing Symmetric Distributions in Fish School Search Optimization Algorithm with Exponential Step Decay. *Symmetry*. 12 (5), 784- 802.
- [15] GC Layek (2015). *An introduction to dynamical systems and chaos*. Springer, New Delhi, India.

Data-centric Advanced Knowledge Interface for Legal Archives

An NLP-Enhanced Tax Statute Text Search System in the Philippines

Matthew Roque

Department of Electronics and
Computer Engineering
De La Salle University
Manila, Philippines
matthew_roque@dlsu.edu.ph

Shirley Chu

College of Computer Studies
De La Salle University
Manila, Philippines
shirley.chu@dlsu.edu.ph

Melvin Cabatuan

Department of Electronics and
Computer Engineering
De La Salle University
Manila, Philippines
melvin.cabatuan@dlsu.edu.ph

Nicole Abejuela

Department of Electronics and
Computer Engineering
De La Salle University
Manila, Philippines
nicole_abejuela@dlsu.edu.ph

Michael Ng

Department of Electronics and
Computer Engineering
De La Salle University
Manila, Philippines
michael_stevens_ng@dlsu.edu.ph

Hans Nolasco

Department of Electronics and
Computer Engineering
De La Salle University
Manila, Philippines
hans_nolasco@dlsu.edu.ph

Jenine Valencia

Department of Electronics and
Computer Engineering
De La Salle University
Manila, Philippines
jenine_valencia@dlsu.edu.ph

ABSTRACT

This study presents an innovative and novel approach to section retrieval from the National Internal Revenue Code (NIRC) of the Philippines, leveraging advanced Natural Language Processing (NLP) techniques. The study shares the first documented dataset on manually annotated questions from the NIRC bar exam review materials, as well as a methodology for generating synthetic data using instruction-tuned Large Language Models such as Mistral 7b. Utilizing text embedding models, the research also explores the efficacy of preprocessing techniques, the impact of learning rates on model performance, and the computational considerations of using language models like Dense Passage Retrieval (DPR) and Jina Embeddings v2. The findings reveal that Jina Embeddings v2, trained on the combination of the original and synthetic datasets, delivers the highest accuracy, successfully retrieving a single relevant section out of 311, 70.52% of the time. A web application (DAKILA) was developed to house the system and serve as an interface between the retrieval pipeline and the user. Feedback was collected from students and practicing professionals from the fields of law and accountancy using the System Usability Scale (SUS), demonstrating strong user satisfaction and DAKILA's potential as a legal research tool.

KEYWORDS

Natural language processing, large language models, information retrieval, text embedding models, Philippine Tax Code, National Internal Revenue Code, statute law retrieval

1 INTRODUCTION

Natural Language Processing (NLP), a subset of Artificial Intelligence (AI) related to Machine Learning (ML) and Deep Learning (DL), has made significant strides in multiple industries within the last decade. Its impact on the legal sector is particularly noteworthy, offering innovative tools that serve both professionals and the public. NLP-driven predictive models have been instrumental in elevating the capabilities of legal practitioners, including lawyers and researchers. These models enhance case analyses and support the formulation of compelling legal arguments, marking a significant leap forward in legal practice [1, 26]. Moreover, advancements in automated summarization and question-answering technologies have dramatically improved the comprehension of complex legal documents. These tools efficiently distill extensive, intricate texts into manageable summaries, alleviating the burden of manual analysis. Consequently, they significantly reduce the likelihood of misinterpretation and variability in understanding, making legal information more accessible and navigable [13, 20, 21].

In the Philippine legal domain, various NLP methodologies have been applied, focusing on Supreme Court cases. One study predicted outcomes of criminal cases using text analysis with machine learning techniques like random forest classifiers and support vector machines [25]. Another explored semantic analysis for retrieving relevant case laws using Doc2Vec and cosine similarity measures [18]. Additionally, Juris2Vec, leveraging a mix of Word2Vec, GloVe, and fastText, created domain-specific embeddings tailored for Philippine legal texts, enhancing the precision and relevance of legal research in the Philippines [14].

While advancements in NLP have notably improved legal analysis and document accessibility in the Philippines, a significant

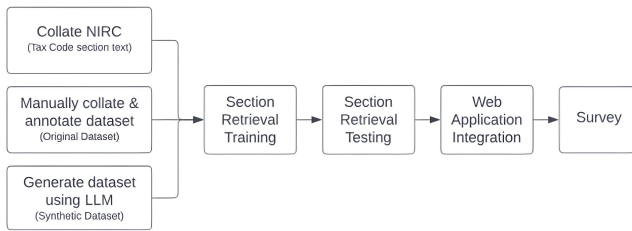


Figure 1: Main Process Flowchart

area for development remains in making statute laws more comprehensible to laypersons and more easily navigable for professionals. Current efforts focus on analyzing Supreme Court decisions, but there is a critical need for initiatives that simplify the statutory text for non-specialists. Enhancing NLP applications to decode complex legal terms into understandable language would not only empower the public with better legal understanding but also aid legal experts by facilitating faster access to essential statutory details. This highlights the importance of dedicated research towards creating tools specifically designed for the intricacies of Philippine statute law, aiming to eliminate the barrier posed by legal jargon.

This study aimed to pioneer a novel and credible section retrieval NLP dataset based on the National Internal Revenue Code (NIRC) as of 2023, also known as the Philippine Tax Code. The creation of NLP datasets tailored to the Philippine legal context is vital for capturing the unique linguistic nuances and sociocultural elements that influence legal terminology in the country [1, 3, 13, 21]. By aligning NLP resources with local contexts, this initiative aims to mitigate ambiguities and bridge interpretative divides that often arise when legal documents are analyzed from a purely foreign standpoint [2]. This localized approach not only enhances the precision of legal NLP applications but also broadens their usability and relevance across different cultural and legal frameworks. Moreover, this project sought to develop the accompanying NLP pipeline for statute law section retrieval. Various models and techniques were employed and evaluated within each task of the ensemble, i.e., dataset generation and relevant section retrieval. Integrating it into a web application, serving as the third objective of the study, facilitated the collection of essential feedback from its targeted primary users—Philippine professionals and students engaged with the law—via a System Usability Scale (SUS) survey.

2 METHODOLOGY

The methodology for this project adopts a comprehensive strategy to develop a robust section retrieval system for the NIRC. It starts with the assembly of a detailed dataset comprised of questions derived from the NIRC and corresponding relevant sections, done through both manual curation and the generation of synthetic data using advanced Large Language Models (LLMs) fine-tuned to follow specific instructions. This dual approach ensures a rich and varied corpus. The core of the system leverages text embedding models, meticulously fine-tuned to grasp the subtleties of Philippine legal texts, for the alignment of queries with pertinent sections. Subsequently, integration of the finalized system into a web application and its beta testing provides insights as to how utilizing NLP to

tackle the distinct challenges presented by legal documents can create a retrieval system that excels in accuracy, computational efficiency, and usability.

2.1 Datasets

The dataset comprised three key components: (1) questions based on the NIRC updated as of 2022, (2) relevant section number, and (3) the text of that relevant section, ensuring each question is matched to only one relevant section.

2.1.1 Manual Annotation for Original Dataset. The original dataset drew its content from reputable Philippine bar exam review materials from 1994 to 2022. Each item was thoroughly checked against the NIRC for validity. To address limitations in quantity and enhance the dataset’s quality, strategies such as negation and paraphrasing were employed, enriching the dataset while mitigating any potential biases [2]. Furthermore, to maintain a focus on the universal applicability, specific names in situational questions were substituted with generic identifiers.

2.1.2 Large Language Models for Generation of Synthetic Dataset. In the interest of further expanding the dataset to encourage diversity and better representation of the NIRC, the utilization of Large Language Models (LLMs) to generate a synthetic dataset was explored. The comparison of model capabilities and constraints shown in Table 1 informed the selection process. The Mistral-7B-Instruct-v0.2 model [7], referred to as Mistral, was chosen for its high performance on the MT-bench [26] and MMLU benchmark [6], indicating strong capabilities in multilingual translation and understanding across a range of subjects—key qualities for creating a diverse legal dataset.

Additional models within the computational scope were Llama-7-2b-chat [23] and Zephyr-7b-beta [24], both compatible with the hardware limits of Google Colab’s A100 GPU with 40 GB of VRAM. Zephyr is trained on the Mistral 7B model, ensuring a high level of performance similar to Mistral. Despite their potential, Mistral’s top-tier performance on benchmarks central to the project’s goals ultimately solidified its selection. In contrast, the larger SOLAR-10.7B-Instruct-v1.0 model [9] was not considered further due to exceeding the hardware’s computational capabilities. Thus, Mistral was chosen, balancing advanced language generation with the available resources to effectively enhance the dataset with synthetic legal text.

The synthetic dataset generation process harnessed the instruction-tuning capability of the Mistral model. Each section of the NIRC was inputted into the model sequentially, with a directive to craft five distinct questions based on the content of the section, along with corresponding answers. Additional instructions were given to prevent leakage of section numbers into the questions and to format the output as "Q: [Question] A: [Answer]". This structured prompt was crucial for the model to generate the desired output. Post-generation, regular expressions (RegEx) were employed to meticulously extract the data and store it in a CSV file. This streamlines the process of dataset creation and ensures ease of integration into the retrieval system. Fifty randomly sampled entries were sent to a lawyer who deemed it to be an adequate reflection of the NIRC.

Table 1: Open-source Large Language Model Comparison for Dataset Generation

Model	No. of Parameters	MT Bench	MMLU
Llama-2-7b-chat	6.74B	6.27	45.80
Mistral-7B-Instruct-v0.2	7.24B	7.60	60.80
Zephyr-7b-beta	7.24B	7.20	52.70
SOLAR-10.7B-Instruct-v1.0	10.70B	7.58	66.20

2.1.3 Dataset Splitting. The project created six distinct sets—three for training and three for testing. The original dataset underwent a stratified train-test split. A parallel process was applied to the synthetic dataset. The last pair of datasets involved combining the original and synthetic datasets prior to the stratified train-test split. Impacts of the LLM-fabricated data inclusion, and the placement of relevant section redistribution within the dataset preparation pipeline, were assessed.

2.2 Section Retrieval Models

The methodology for section retrieval from the NIRC considered several leading embedding models as shown in Table 2, evaluating them on key aspects such as sequence length handling, performance on benchmarks, and accessibility. BERT [4] set a foundational standard for deep bidirectional representation, while SBERT [19] refined this for sentence-level embeddings, and DPR [8] focused on dense passage retrieval. Despite their advancements, these models are constrained by a token limit of 512, which is insufficient for the detailed sections of the NIRC. Cohere AI’s cohere-embed-english-v3.0 model and OpenAI’s text-embedding offerings were also reviewed. They exhibit an ability to handle longer contexts, a necessity for the intricate legislative text. However, their proprietary and paywalled API-only nature limits their practicality for extended use, especially considering the potential scale of section retrieval tasks for the NIRC.

The jina-embeddings-v2-base-en model (herein referred to as Jina V2) emerges as a strong contender, primarily due to its impressive token limit of 8192 [5]. This is particularly relevant given the dense nature of legal documents; for instance, the longest section of the NIRC comprises 6390 words, resulting in 8060 tokens upon processing, as shown in Table 3.

In addition to Jina V2, the Dense Passage Retrieval (DPR) model was also considered for the premise of its specialization in retrieving long passages given short queries. For sections that exceeded its token limit of 512, a chunking approach was employed, dividing the text into overlapping chunks to maintain context continuity. These chunks were individually processed through DPR, and the resultant embeddings were aggregated to compute a representative mean for the entire section. For DPR, embeddings are derived from a small number of questions and passages using a dual-encoder framework, utilizing separate BERT-based models for processing both queries and context. This technique balances the model’s token constraints with the need to preserve the depth and meaning of the legal text, making DPR a valuable addition to the testing framework.

The decision to explore BERT and its derivatives for section retrieval within the NIRC was informed by their notable success in

legal AI competitions such as the Competition on Legal Information Extraction/Entailment (COLIEE) [10, 15–17] and Automated Legal Question Answering Competition (ALQAC) [22], as well as research findings outside of these competitions [3]. These sources collectively underscore the utility of BERT-based models in parsing and understanding complex legal texts, a capability that aligns with the project’s objectives. Moreover, recognizing that the legal AI field may not fully leverage the latest NLP and deep learning advancements, there was a deliberate effort to integrate more recent NLP technologies. This approach aims to harness underutilized deep learning advancements, potentially enhancing the performance and sophistication of legal document analysis beyond current legal AI applications.

Moreover, Jina V2’s architecture, which includes mechanisms such as Attention with Linear Biases and Gated Linear Units, provides a robust framework for the nuances of legislative language processing. The pre-training on the “Colossal Cleaned Common Crawl (C4)” dataset and further fine-tuning ensures that the model is well-adjusted to English, suitable for the language of the NIRC. The model’s capabilities are further validated by its performance on benchmarks like MTEB and LoCo, indicating significantly high results in embedding-related tasks, including long-context document handling. The high scores suggest that Jina V2 can create embeddings that capture deep semantic meanings, essential for accurately matching queries to the relevant sections of the NIRC.

The MTEB (Massive Text Embedding Benchmark) [12] is a multifaceted benchmarking suite that specifically gauges the performance of text embedding models on a variety of retrieval-related tasks. It encompasses a broad spectrum of challenges, including document retrieval, question answering, and semantic search, among others. The aim is to measure how effectively a model can understand and match query intent with relevant text content from a large dataset, a crucial capability for information retrieval applications.

The LoCo (Long Context) benchmark complements this by testing a model’s proficiency in managing and interpreting lengthy text inputs. It is especially pertinent for evaluating a model’s performance on extended passages, which is a common characteristic of legal documents like the NIRC. LoCo assesses whether a model can maintain its performance when dealing with long sequences, which is essential for the retrieval of comprehensive sections that contain the nuanced information required to accurately respond to complex queries.

In selecting the Jina V2 model for section retrieval within the NIRC, MTEB’s focus on retrieval-related tasks and LoCo’s focus on tasks for longer contexts provide a strong foundation for its appropriateness. These ensure that Jina V2 not only performs well in general language tasks but also excels in the specific domain of

Table 2: Embedding Model Comparison for Section Retrieval

Model	Sequence Length	MTEB	LoCo	Deployment
bert-base-uncased	512	-	-	Local
all-mpnet-base-v2	512	-	-	Local
dpr-single-nq-base	512	-	-	Local
Cohere embed-english-v3.0	512	64.47	66.60	API only
jina-embeddings-v2-base-en	8192	60.39	85.45	Local
text-embedding-3-small	8191	62.26	82.40	API only
text-embedding-ada-002	8191	60.99	52.70	API only

Table 3: Summary of NIRC Longest Section Lengths

Section Number	Number of Words	Jina V2 Tokens	BGE Reranker Tokens
34	6390	8060	8932
148	3070	4437	4851
144	2489	3286	3699
288	2199	3089	3466
22	2051	2684	2959

long context retrieval, which is central to the project’s objectives. The model’s high scores in these benchmarks signify its capability to create precise embeddings that facilitate the accurate matching of queries to relevant sections of the text, validating its use for this research.

Training and testing of the models were conducted with distinct approaches for each phase. Training was performed on the different datasets, with the model encoding question-relevant section pairs. The loss was calculated based on one minus the cosine similarity between the embeddings, aiming to minimize the distance between semantically related question and section embeddings. For the testing phase, a retrieval task approach was adopted where section text embeddings were pre-computed, and each question was dynamically encoded. The cosine similarity between the question embedding and all section embeddings was calculated to rank the sections according to relevance; whether the correct relevant section was the top-ranked by the model or not was the basis for calculating the accuracy. This process was facilitated using Google Colab and a V100 GPU to leverage high computational power. The training process was iteratively evaluated over various learning rates with three epochs each to identify the optimal model configuration. Memory constraints necessitated the use of smaller batch sizes, coupled with gradient accumulation techniques, to achieve an effective batch size of 40, which was determined to be ideal for stability [11]. Optimization was carried out using the AdamW optimizer, ensuring efficient training dynamics.

2.3 Deployment

2.3.1 Web Development. The development of the DAKILA web application encapsulates a comprehensive user journey, starting from a straightforward search page facilitated by a navigation bar for easy access to various sections like Search, About Us, and the

Tax Code. Even before receiving a query, upon setting up the server, the fine-tuned section retrieval model is pre-loaded, along with its tokenizer and pre-computed NIRC embeddings. Upon receiving a query, the DAKILA algorithm initiates a step-by-step process: first, it encodes this query using the tokenizer and fine-tuned model. Next, it calculates cosine similarity scores between the query embedding and all NIRC section embeddings, ranking them accordingly. The top relevant sections, up to a user-specified number ‘n’ (defaulting to 5), are then displayed to the user. This system, developed with Flask and utilizing pre-computed embeddings, ensures efficient and consistent retrieval of legal sections. This architecture ensures that user inputs are precisely processed, returning the most relevant legal sections, thereby showcasing the potential of advanced NLP techniques in making legal information more accessible and navigable not only for professionals in law and accountancy but also for the average layperson.

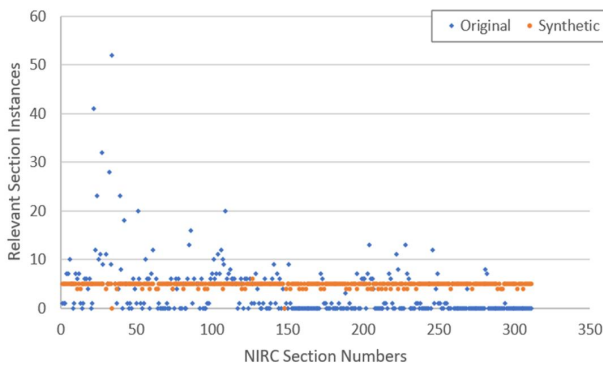
2.3.2 System Usability Scale Survey. Deployment of the DAKILA web application was conducted, with user testing providing practical insights about its functionality and user interface. Volunteers from law and accountancy fields interacted with the application. Their experiences were quantitatively measured through the System Usability Scale (SUS) survey accompanied by an additional Likert scale question inquiring about the perceived quality of its section retrieval output. The questions asks if the user prefers DAKILA over existing legal research tools, such as Google, CD Asia, and LawPhil, when navigating the NIRC. Through the SUS survey, we aim to evaluate the DAKILA web application’s ease of use, efficiency, and overall satisfaction in comparison to existing legal research platforms. Overall, the survey seeks to understand the application’s effectiveness in streamlining the search and retrieval process for tax laws, focusing on user-friendly navigation and the practical utility of the tool. Additionally, it will help highlight areas for improvement, ensuring that DAKILA meets the evolving needs of legal professionals and students by providing a more intuitive and resourceful alternative to traditional legal research methods.

3 RESULTS AND DISCUSSION

This section details experimentation results leading up to the determination of the best NLP pipeline for the purposes of this study. The Jina-V2 model trained on the training dataset combined prior to the stratified train-test split, at a learning rate of 1e-06 yielded the highest test accuracy score.

Table 4: Dataset Samples

Relevant Section Number	Dataset	Query
259	Original	Do you need a license to collect foreign payments?
172	Original	Is there a time limit on how long a package can be detained by a revenue officer without legal proceedings?
107	Original	Is an importer of flowers from abroad in 2011 liable for VAT?
60	Original	What are the various trusts subject to income tax?
24	Original	Is the 6% final tax rate imposed on all sales of real property classified as capital assets?
151	Synthetic	What is the definition of 'gross output' in the context of mineral taxation according to the Philippine Tax Code?
147	Synthetic	What are 'heated tobacco products' and how are they different from 'vapor products' as defined in this tax code?
183	Synthetic	What is the stamp tax for a life insurance policy with a coverage of PHP 1.2 million?
139	Synthetic	What happens to fermented liquor that is unfit for consumption due to damage?
283	Synthetic	How much of the excess collections from certain national taxes are distributed to local governments and how much is kept by the National Government?

**Figure 2: Relevant NIRC Section Distribution in the Original and Synthetic Datasets**

3.1 Datasets

The original dataset's composition hinged on the availability of bar exam reviewers. Figure 2 presents the original dataset's skewed distribution of the NIRC sections, highlighting areas deemed crucial for legal practice. This skew may align with common search queries but does not fully reflect the breadth of the NIRC. To address potential gaps in coverage, the Mistral 7B-generated dataset aimed to create a balanced representation, with a law professional confirming the synthetic dataset's quality. Table 4 shows samples from both datasets.

The combined datasets, inclusive of both original and synthetically generated data, were foundational to the superior performance of the Jina V2 model, despite the extended training time required as seen in Tables 5 and 7. A strategic approach to combining the two datasets and partitioning with stratification for training and testing purposes contributed to a modest yet noteworthy improvement in

Table 5: Summary of Datasets

Dataset	Training	Testing	Average Training Time (min)	
			DPR	Jina V2
Original	918	102	3.10	4.43
Synthetic	1333	149	2.13	6.42
Combined	2251	251 ^a	5.18	10.87

^a This is comprised of 97 entries from the original dataset and 154 from the synthetic dataset.

model accuracy. This performance uptick underscores the value of a more diverse and representative dataset that ensures entries from each NIRC section.

3.2 Section Retrieval Models

3.2.1 Training. In determining the best preprocessing techniques for both DPR and Jina V2 models, various methods were scrutinized individually for their impact on accuracy using the combined test set. For the DPR model, lowercasing, removal of footnotes, removal of special characters, and lemmatization emerged as beneficial, leading to improvements in accuracy. Contrastingly, the Jina V2 model performed optimally without any preprocessing. With these insights, the training and testing pipelines for each model were implemented accordingly. Experimentation with data partitioning showed that a 90:10 train-test split was optimal for the project needs.

Figure 3 showcases a plot of learning rate versus accuracy across three epochs for Jina V2 on the combined dataset, revealing that a learning rate of $1e-06$ was optimal, with higher epochs generally leading to reduced accuracy except at very slow learning rates. DPR and Jina V2 trained on the other variations of the dataset showed their peaks at different learning rates, which are shown in Tables 6 and 7, however, the observed decline in accuracy with

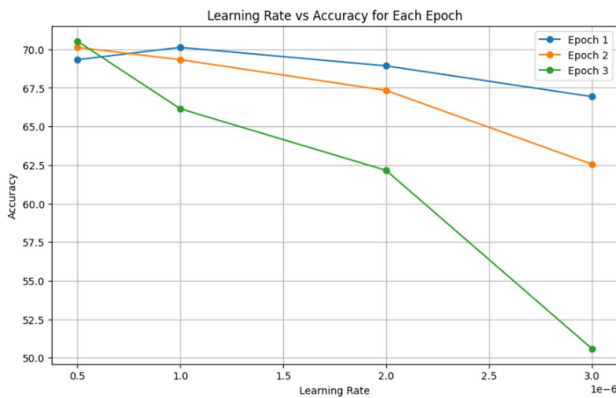


Figure 3: Learning Rate vs Accuracy for Each Epoch

an increase in the number of epochs ultimately holds true. The decrease in accuracy with additional epochs suggests that the models’ pre-existing language comprehension could be compromised by excessive fine-tuning, leading to overfitting.

To preserve the innate capabilities of the models and optimize accuracy, the final training iterations were conducted for a single epoch, without a validation set, thereby utilizing the full scope of the training data. Typically, a validation set is utilized between epochs during model training to monitor and adjust the model’s performance on data not seen during training. This intermediate evaluation helps in fine-tuning model parameters and preventing overfitting, ensuring the model generalizes well to new data. Given that training for a single epoch yielded the best results, the decision to bypass the validation set was made, eliminating the need for validation steps between epochs. This approach streamlined the training process, focusing on maximizing the efficiency and accuracy of the model.

3.2.2 *Testing.* For testing of the models fine-tuned on the combined training dataset, the combined testing dataset questions were first labeled as being either from the original or synthetic dataset and sorted accordingly. However, for testing the models fine-tuned on both the original and synthetic training datasets, the original and synthetic testing datasets were merged for the combined testing dataset. This was to ensure no overlap between the training and testing datasets.

Table 6 surmises that the DPR model attained its highest test accuracy of 34.02% when trained on the combined training dataset with a learning rate of 6e-07. This is significantly surpassed by Jina V2’s performance shown in Table 7, regardless of the testing dataset, when fine-tuned on the combined training dataset, sporting the best test accuracy scores of 63.92% on the original testing dataset, 74.68% on the synthetic testing dataset, and 70.52% on the combined testing dataset. This suggests a heightened model performance with the integration of the synthetic dataset. This substantial difference justifies the tradeoff in choosing the latter as the final section retrieval model despite its longer testing times documented in Figure 4.

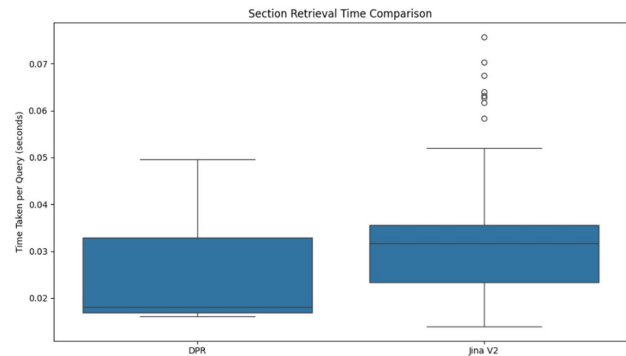


Figure 4: Testing Time per Query for Section Retrieval Models



Figure 5: DAKILA Web Application

3.3 Deployment

3.3.1 *Comparison to Existing Resources.* The comparison between DAKILA and existing legal resources for statute law in the Philippines underscored DAKILA’s advanced capabilities. Other resources often lacked the contextual understanding DAKILA offered, relying on word frequencies and direct string matches that could miss relevant information or introduce irrelevant results due to their inability to interpret the nuances of legal queries. DAKILA’s semantic search, coupled with features like a higher character limit for queries and direct navigation to pertinent NIRC sections, addressed these drawbacks effectively. This not only improved the precision of search results but also enhanced user experience by streamlining access to specific legal information.

3.3.2 *System Usability Scale Survey.* The System Usability Scale (SUS) employed in evaluating DAKILA utilized a 5-scale rating, where participants rated their agreement with a series of statements. Positive statements were assigned to odd-numbered questions, with 5 indicating strong agreement, reflective of a favorable view. Conversely, even-numbered questions were framed negatively, where a score of 1 represented the best outcome, indicating a lack of issues or challenges. This dual-phrased approach provided a balanced

Table 6: Test Accuracy Scores for DPR Fine-tuning

Learning Rate	Training Dataset	Testing Dataset		
		Original	Synthetic	Combined
3e-06	Original	30.39%	20.81%	24.70%
1e-06	Synthetic	28.43%	23.49%	25.50%
6e-07	Combined	34.02%	20.13%	25.50%
	Untrained	22.55%	21.48%	20.72%

Table 7: Test Accuracy Scores for Jina V2 Fine-tuning

Learning Rate	Training Dataset	Testing Dataset		
		Original	Synthetic	Combined
3e-06	Original	59.80%	59.06%	59.36%
7e-06	Synthetic	54.90%	59.06%	57.37%
1e-06	Combined	63.92%	74.68%	70.52%
	Untrained	51.96%	57.72%	55.38%

Table 8: SUS Survey Results

No.	SUS Questions	Mean	SD
1	I think that I would like to use this system frequently.	4.57	0.56
2	I found the system unnecessarily complex.	1.90	1.22
3	I thought the system was easy to use.	4.70	0.64
4	I think that I would need the support of a technical person to be able to use this system.	1.47	0.72
5	I found the various functions in this system were well integrated.	4.67	0.54
6	I thought there was too much inconsistency in this system.	1.60	0.66
7	I would imagine that most people would learn to use this system very quickly.	4.73	0.51
8	I found the system very cumbersome to use.	1.80	1.19
9	I felt very confident using the system.	4.63	0.55
10	I needed to learn a lot of things before I could get going with this system.	1.83	1.04

measure of the system’s usability from the perspectives of ease of use and potential frustrations. The survey results for DAKILA, as shown in Table 8, demonstrated strong user satisfaction across diverse demographics. Participants ranged from under 20 to over 60 years old, with the largest group being those between 21-29 years old. The professional background of respondents spanned law and accountancy, including both students, and practicing professionals. High SUS scores across all age and professional categories emphasized the system’s ease of use and functionality. This broad approval showcases DAKILA’s potential as a specialized tool for legal research, offering a user-centric alternative to traditional resources.

4 CONCLUSION AND RECOMMENDATIONS

The investigation into the section retrieval of the NIRC culminated a deeper understanding of the interplay between dataset quality and model training. The uptick in performance upon the introduction of the LLM-generated synthetic dataset offered a more diverse and representative dataset compared to the original, which sported a skewed section distribution. The study also confirmed a heightened

effectiveness of section retrieval with slower learning rates and less training epochs to avoid diminishing the embedding model’s pre-existing language comprehension due to detrimental overfitting because of excessive fine-tuning. Amidst the evaluation of multiple models Jina V2 outputted the highest test accuracy scores, showcasing the capabilities of advanced language models without the need for pre-processing steps. Overall, the relationship between careful dataset construction and targeted model training strategies for improved legal search tools is essential.

Future developments of NLP-driven Philippine statute law retrieval systems should explore appropriate annotations, methodologies, and metrics in dealing with queries with multiple relevant sections. Although positive datasets are more common, such as what was practiced in this study, incorporation of negative datasets in training may provide more valuable insights into semantic relevance. With the recent exponential rise in caliber and quantity of released LLMs arises the opportunity to expound on the effectiveness of automated dataset generation to alleviate the heavy manual labor of dataset creation and annotation while only minimally compromising quality. This addition to the ensemble should be tested

across a multitude of legal documents in varying sizes and domains. Finally, reranking methods should be explored to further refine the ensemble and improve its accuracy.

ACKNOWLEDGMENTS

We would like to thank the law professionals who dedicated their service without charge to our endeavors.

REFERENCES

- [1] Intisar Almuslim and Diana Inkpen. 2022. Legal judgment prediction for canadian appeal cases. In *2022 7th International Conference on Data Science and Machine Learning Applications (CDMA)*. IEEE, 163–168.
- [2] Dang Hoang Anh, Dinh-Truong Do, Vu Tran, and Nguyen Le Minh. 2023. The impact of large language modeling on natural language processing in legal texts: a comprehensive survey. In *2023 15th International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, 1–7.
- [3] Chieu-Nguyen Chau, Truong-Son Nguyen, and Le-Minh Nguyen. 2020. Vnlawbert: A vietnamese legal answer selection approach using bert language model. In *2020 7th NAFOSTED Conference on Information and Computer Science (NICS)*. IEEE, 298–301.
- [4] Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
- [5] Michael Günther, Jackmin Ong, Isabelle Mohr, Alaeddine Abdessalem, Tanguy Abel, Mohammad Kalim Akram, Susana Guzman, Georgios Mastrapas, Saba Sturua, Bo Wang, et al. 2023. Jina embeddings 2: 8192-token general-purpose text embeddings for long documents. *arXiv preprint arXiv:2310.19923* (2023).
- [6] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300* (2020).
- [7] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).
- [8] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).
- [9] Dahyun Kim, Chanjun Park, Sanghoon Kim, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, et al. 2023. Solar 10.7 b: Scaling large language models with simple yet effective depth up-scaling. *arXiv preprint arXiv:2312.15166* (2023).
- [10] Mi-Young Kim, Juliano Rabelo, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. 2022. COLIEE 2022 summary: methods for legal document retrieval and entailment. In *JSAI International Symposium on Artificial Intelligence*. Springer, 51–67.
- [11] Timo Möller, Julian Risch, and Malte Pietsch. 2021. GermanQuAD and GermanDPR: Improving non-English question answering and passage retrieval. *arXiv preprint arXiv:2104.12741* (2021).
- [12] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. MTEB: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316* (2022).
- [13] Varsha Naik, Rajeswari Kannan, Sanket Agarwal, Aryan Sable, and Himanshu Chaudhari. 2023. An Effective Search Algorithm for Analyzing and Extracting Indian Legal Judgments using NER and Document Summarization. In *2023 7th International Conference On Computing, Communication, Control And Automation (ICCCUBEA)*. IEEE, 1–6.
- [14] Elmer Peramo, Charibeth Cheng, and Macario Cordel. 2021. Juris2vec: Building Word Embeddings from Philippine Jurisprudence. In *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*. IEEE, 121–125.
- [15] Juliano Rabelo, Randy Goebel, Mi-Young Kim, Yoshinobu Kano, Masaharu Yoshioka, and Ken Satoh. 2022. Overview and discussion of the competition on legal information extraction/entailment (COLIEE) 2021. *The Review of Socionetwork Strategies* 16, 1 (2022), 111–133.
- [16] Juliano Rabelo, Mi-Young Kim, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. 2020. A summary of the COLIEE 2019 competition. In *New Frontiers in Artificial Intelligence: JSAI-isAI International Workshops, JURISIN, AI-Biz, LENLS, Kansei-AI, Yokohama, Japan, November 10–12, 2019, Revised Selected Papers 10*. Springer, 34–49.
- [17] Juliano Rabelo, Mi-Young Kim, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. 2021. COLIEE 2020: methods for legal document retrieval and entailment. In *New Frontiers in Artificial Intelligence: JSAI-isAI 2020 Workshops, JURISIN, LENLS 2020 Workshops, Virtual Event, November 15–17, 2020, Revised Selected Papers 12*. Springer, 196–210.
- [18] Lorenz Timothy Barco Ranera, Geoffrey A Solano, and Nathaniel Oco. 2019. Retrieval of semantically similar Philippine supreme court case decisions using Doc2Vec. In *2019 International Symposium on Multimedia and Communication Technology (ISMATC)*. IEEE, 1–6.
- [19] N Reimers. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv preprint arXiv:1908.10084* (2019).
- [20] Samarth Singhal, Siddhant Singh, Sandeep Yadav, and Anil Singh Parihar. 2023. LTSum: Legal Text Summarizer. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, 1–6.
- [21] Oussama Tahtah, Yassine Akhiat, Ahmed Zinedine, and Khalid Fardousse. 2023. Towards a Question/Answering System in Moroccan Legal Domain: data preparation and question classification phase using ML approaches. In *2023 7th IEEE Congress on Information Science and Technology (CiSt)*. IEEE, 140–144.
- [22] Nguyen Ha Thanh, Bui Minh Quan, Chau Nguyen, Tung Le, Nguyen Minh Phuong, Dang Tran Binh, Vuong Thi Hai Yen, Teeradaj Racharak, Nguyen Le Minh, Tran Duc Vu, et al. 2021. A summary of the alqac 2021 competition. In *2021 13th international conference on knowledge and systems engineering (kse)*. IEEE, 1–5.
- [23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).
- [24] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944* (2023).
- [25] Michael Benedict L Virtucio, Jeffrey A Aborot, John Kevin C Abonita, Roxanne S Avinante, Rother Jay B Copino, Michelle P Neverida, Vanesa O Osiana, Elmer C Peramo, Joanna G Sjujuo, and Glenn Brian A Tan. 2018. Predicting decisions of the philippine supreme court using natural language processing and machine learning. In *2018 IEEE 42nd annual computer software and applications conference (COMPSAC)*, Vol. 2. IEEE, 130–135.
- [26] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems* 36 (2023), 46595–46623.

Evaluating ChatGPT for Joint Intent Detection and Slot Filling: Zero-Shot vs. Few-Shot Prompting

Hannah Fong
De La Salle University
Manila, Philippines
hannah_regine_fong@dlsu.edu.ph

Ethel Ong
De La Salle University
Manila, Philippines
ethel.ong@dlsu.edu.ph

ABSTRACT

Natural language understanding transforms an unstructured text into a structured semantic representation through intent detection (ID) and slot filling (SF). Research shows that jointly performing the two tasks results in better performance due to the interdependencies between intent classes and slot labels. The current state-of-the-art joint ID and SF models are based on the pre-trained language model BERT. This raises the question of how large-scale pre-trained language models (LLMs) would perform on the joint task. LLMs are reported to achieve competitive performance in zero-shot prompting, and better in few-shot prompting. In this paper, we report our evaluation of ChatGPT’s performance on the joint ID and SF task under zero-shot and different few-shot (i.e., 1-shot, 5-shot, 10-shot, 20-shot, 30-shot) prompt settings. Results showed that ChatGPT’s performance generally improves when given more examples in the prompt. However, its performance lags behind other LLMs and SOTA joint models. It also has the tendency to exhibit unexpected behaviors such as generating unknown outputs, outputting incorrect number of slot labels, and annotating a different utterance from what was provided. These unexpected behaviors were observed more frequently when sending longer prompts, demonstrating the effect of prompt length to ChatGPT’s performance.

KEYWORDS

joint intent detection and slot filling, large language models, ChatGPT, zero-shot prompting, few-shot prompting

1 INTRODUCTION

Natural language understanding (NLU) is the key component in a conversational agent architecture that transforms an unstructured input in human language into a structured semantic representation that can be processed by a machine [14]. It performs two important tasks: intent detection and slot filling [31]. Intent detection (ID) is a classification task that maps an utterance to a finite set of intent classes that signifies the intention or goal of that particular utterance. Slot filling (SF) is a sequence labeling task that attaches labels to the sequence of tokens in an utterance. Traditionally, these two tasks were treated independently and performed sequentially. However, it was discovered that the intent class and slot labels of an utterance have interdependencies that, when modelled together, can result in better performance [31]. This led research work that investigates the creation of joint ID and SF models to perform the two tasks simultaneously, leading to better performance compared to the traditional independent models.

The earliest approach to joint ID and SF models utilized statistical methods such as conditional random fields and hidden Markov models [2, 11]. These models outperformed the independent ID and

SF models. With the emergence of deep learning, neural network architectures, such as convolutional neural networks (CNN) and recurrent neural networks (RNN), were applied to solve the joint task [15, 32]. In 2017, the transformer neural network architecture was introduced and became the building block of large-scale language models that can understand and generate human language [16, 28]. Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based language model that produced state-of-the-art (SOTA) results in various natural language processing (NLP) tasks upon its release [6]. This prompted the implementation of BERT-based joint ID and SF models which currently exhibit SOTA performance on the joint task [27].

Recently, ChatGPT which is a conversational LLM fine-tuned from the generative pre-trained transformer (GPT) 3.5, has shown strong performance on various NLP tasks [25], including zero-shot performance on the joint ID and SF task [21]. A detailed analysis of the results revealed that ChatGPT performs relatively well in ID but poorly in SF. The authors experimented with adding slot descriptions and examples in the prompts and found these to improve the performance of ChatGPT in the SF task for the benchmark SNIPS NLU dataset [5]. However, the details of the experiment, such as the number of examples provided, were not documented.

In this paper, we report our systematic evaluation of ChatGPT’s performance on the joint ID and SF task across zero-shot and different few-shot prompt settings for the Airline Travel Information System (ATIS) dataset, which is another widely used NLU benchmark dataset [21]. Results showed that ChatGPT’s performance generally improves when given more examples in the prompt. However, its performance lags behind other LLMs and SOTA joint models. It also has a tendency to exhibit unexpected behaviors such as generating unknown outputs, outputting incorrect number of slot labels, and annotating a different utterance from what was provided. Our methodology and findings can serve as a reference for future works that intend to evaluate ChatGPT’s performance on NLU tasks with varying prompt settings.

2 RELATED WORKS

In this section, we provide a review of early works and recent approaches in joint intent detection and slot filling, including the use of large language models.

2.1 Early Approaches in Joint ID and SF

The earliest work on the joint ID and SF task devised triangular-chain conditional random fields (TriCRFs) that jointly represent the ID and SF in a single graphical model, thereby naturally capturing the relationship between intent classes and slot labels [11]. Their proposed approach achieved higher F1-scores for both ID and SF

over independent, cascaded, and reranked approaches. However, this approach requires a large amount of labeled training data, which is expensive to obtain [30]. Wang [30] showed that limited training data degrades the joint model’s ID performance.

To address the problem of scarce labeled training data, Celikyilmaz and Hakkani-Tur [2] proposed leveraging both labeled and unlabeled data to jointly learn the intent classes and slot labels of utterances. Their approach is a semi-supervised generative multi-layer context model where an utterance is represented using hidden Markov models. Prior knowledge extracted from unlabeled web search queries are utilized to enhance the semantic component extraction from utterances. With this, a higher ID accuracy compared to the cascaded and TriCRFs approaches is achieved when the amount of labeled training data is small (i.e., 25% of the training data). However, it underperformed in SF, measured using the F1-score, compared to the two baseline approaches.

2.2 Deep Learning Models for Joint ID and SF

The emergence of deep learning prompted Xu and Sarikaya [32] to apply the neural network architecture in performing the joint task. They proposed a CNN-based TriCRFs model where features are automatically extracted from the utterances by the convolutional layers that are shared between the two tasks. The intent and slot dependencies are jointly modeled using TriCRFs. The proposed model outperformed the independent and TriCRFs approaches by achieving a lower ID error rate and higher SF F1-score.

The strong sequence modeling performance of RNNs prompted Liu and Lane [15] to use it for the joint ID and SF task in an encoder-decoder architecture. A bidirectional RNN encoder is shared by two unidirectional RNN decoders for ID and SF, respectively. Both RNNs use the Long Short-Term Memory (LSTM) cell as the basic RNN unit to model long-term dependencies. The proposed model achieved lower ID error rate and higher SF F1-score compared to the previous joint models (i.e., TriCRFs and CNN TriCRFs).

The deep learning-based models demonstrated strong performance on the joint ID and SF task. However, these require a vast amount of labeled training data which are time-consuming and costly to obtain [27]. The scarcity of labeled data diminishes the generalizability of these models. As an alternative, language models pre-trained on a large corpora of unlabeled text were developed to learn the linguistic patterns of a language [16]. Transformers are used for their efficient parallel processing capability [29]. One example is BERT, which has achieved SOTA performance in many NLP tasks. Shafi and Chachoo [27] investigated how BERT can improve the generalizability of the joint ID and SF model. They proposed a multi-stage framework trained on BERT with a bidirectional gated recurrent unit and self-attention mechanism as the ID decoder and a capsule network as the SF decoder. Their proposed model achieved a higher F1-score for both ID and SF compared to existing SOTA approaches.

2.3 Large Language Models for Joint ID and SF

The transformer architecture, increased computational power, and availability of vast datasets paved the way for large-scale language models (LLM) [16]. LLMs are pre-trained with billions of model parameters and terabytes of data. Through their massive scale, LLMs

acquire emergent abilities that enable them to perform well in a variety of NLP tasks, including arithmetic reasoning, logical reasoning, and question answering without being specifically trained in these domains [3, 16].

ChatGPT is a conversational LLM fine-tuned from GPT 3.5 with reinforcement learning from human feedback to generate human-like responses to prompts [25]. With its demonstration of promising performance in different NLP tasks, research interests in evaluating the model’s performance in ID and SF tasks began to surface. Pan et al. [21] conducted a preliminary study to evaluate ChatGPT’s zero-shot performance on the joint ID and SF task. However, analysis of their experiments revealed that one-shot prompting may have been used instead based on the prompt template wherein one example input-output pair was included to specify the output format. Results showed that ChatGPT reached good ID accuracy but not at par with other LLMs (i.e., GPT-3.5 text-davinci-003 version and Codex) and the fine-tuned SOTA model (i.e., Co-Interactive Transformer). It exhibited the lowest F1-score in SF compared to the other models. The study also experimented with adding slot descriptions and examples to the prompt which increased the SF performance. This experiment was performed for the SNIPS NLU dataset [5].

3 TASK DESCRIPTION

In this section, we define the joint intent detection and slot filling task. We also describe the ATIS dataset which was used in the experiment. Prompt engineering is an important component in instructing ChatGPT to perform the joint ID and SF task; we briefly explain this component in relation to our requirements.

3.1 Joint Task

Intent detection is the task of identifying the main goal of a user utterance [7]. It classifies a given utterance x into one of the predefined intent classes y^i based on the feature/s extracted from the utterance. Slot filling is the task of assigning semantic labels to the words in a given utterance. Given an input word sequence $x = (x_1, \dots, x_N)$, slot filling assigns each word a slot label $y^s = (y_1^s, \dots, y_N^s)$ from a predefined list of slot labels. As a joint task, the relationship between the intent class and slot labels in a given utterance is modeled and used to inform the ID and SF tasks.

3.2 Dataset

The Airline Travel Information Systems (ATIS) dataset [21] is a standardized dataset widely used for the joint ID and SF task. It consists of 5,871 utterances in the air travel domain [10, 22]. The dataset can be divided into 4,978 utterances for the training set and 893 utterances for the test set. It has 21 unique intent classes such as *airfare*, *flight*, and *ground service*. It has 120 slot labels that can be categorized into *airline*, *airport*, *arrival date*, *departure date*, *city name*, and *flight*.

ATIS contains single-turn utterances associated with a single intent in a single domain (i.e., air travel). Table 1 shows one training sample for a given utterance: “*i want to fly from baltimore to dallas round trip*”. It has a single intent, identified as “*atis_flight*”. The slot labels follow the IOB tagging format where the “B” prefix indicates the beginning of a slot, while the “I” prefix indicates

the word inside or at the end of a slot [8]. The “O” label indicates that a word does not correspond to any of the predefined slot labels. In the given example, the word “*baltimore*” is labeled as “B-fromloc.city_name” to indicate the source city, while the word “*dallas*” is labeled as “B-toloc.city_name” to indicate the destination city. The word “*round*” is labeled as “B-round_trip” as it is the first part of the compound word “*round trip*” while “*trip*” is labeled as “I-round_trip” because it is part of the previous slot label. The other words in the given utterance are labeled as “O” because they do not signify any of the predefined slot labels.

3.3 Prompt Engineering

Prompts are used to guide an LLM in generating a response or performing a task [3]. Prompt engineering is the process of designing and optimizing prompts to obtain the desired output or response [3]. To instruct ChatGPT to perform the joint ID and SF task, we utilized zero-shot and few-shot prompting. In zero-shot prompting, an LLM must answer a question it was not trained on without being given any examples. Conversely, in few-shot prompting, examples are included in the prompt to guide the LLM in generating the desired response.

To automatically send queries to and receive responses from ChatGPT, we used the Chat Completions endpoint provided by OpenAI API [17]. The original prompt given to ChatGPT is shown in Table 2. The general structure of the prompt is patterned after the prompt formulated by Pan et al. [21] which consists of three (3) sections: intent class constraints, slot label constraints, and regulations. A persona or role (“*You are a joint intent detection and slot filling model.*”) and a brief task description (“*You need to identify the intent and slot labels of a given sentence.*”) were appended at the start of the prompt as prescribed by OpenAI [18]. The list of intent classes and slot labels, as provided in the ISEAR dataset, is shown in Table 2; the list is redacted for brevity.

There are five (5) regulations specified in the prompt. The first and second regulations inform ChatGPT to generate the correct number of outputs. For ID, only one intent class is provided as each utterance in the ATIS dataset only has a single intent. For SF, every word in a sentence must be given a slot label using IOB tagging. The third and fourth regulations restrict ChatGPT’s responses to the predefined list of intent classes and slot labels provided in the ATIS dataset. The fifth regulation instructs ChatGPT to generate the output in JSON format for easier parsing of responses. An output template is also provided for clarity.

4 METHOD

The performance of ChatGPT on the joint ID and SF task was evaluated under zero-shot and few-shot prompt settings using the ATIS dataset. For the few-shot prompting, five (5) experiments were conducted to insert five (5) different numbers of examples (i.e., 1, 5, 10, 20, 30) to see how these affect ChatGPT’s performance on the joint ID and SF task.

4.1 Dataset

Of the 4,978 utterances in the training set of ATIS, only the first 30 utterances were used as examples in the few-shot prompting experiment. The entire test set (i.e., 893 utterances) was used to

evaluate the performance of ChatGPT in the joint task. ChatGPT was instructed to only choose from the 21 intent classes and 120 slot labels provided in the ATIS dataset for intent classification and slot filling, respectively. The utterances, intent classes, and slot labels from the ATIS dataset were used as is in the experiments.

4.2 Experiments

We performed the experiments using zero-shot prompting and few-shot prompting by inserting five (5) different numbers of examples (i.e., 1, 5, 10, 20, 30). Prompts are sent through the *messages* parameter of the OpenAI API call which takes an array of messages, each represented as a JSON object with two keys: role and content [19]. An example of a message JSON object is `{“role”: “system”, “content”: “You are a helpful assistant.”}`. The *system* role sets the behavior of ChatGPT as a helpful assistant. Another example is `{“role”: “user”, “content”: “Who won the world series in 2020?”}`. In this example, the *user* role signifies that the prompt is sent by the user, which must be responded to by ChatGPT.

The Chat Completions endpoint was used to process the entire test set of the ATIS dataset by ChatGPT. The model employed was `gpt-3.5-turbo-1106` as it was the latest free model that supported the JSON mode [20]. The first message in the API call is a system message where the prompt as described in Table 2 was specified. For few-shot prompting, additional system messages were sent, named “*example_user*” and “*example_assistant*”. These system messages serve as “faked example messages” and are used to demonstrate the desired output to ChatGPT; thus, these messages are not part of a real conversation [26]. The final message in the API call is a user message that specifies the actual utterance to be annotated by ChatGPT. The *response_format* parameter was used to constrain the model’s output to a JSON object. Only one utterance is processed by ChatGPT per API call.

We then validated the output of ChatGPT for the six (6) prompt settings by checking that each utterance has been annotated with the correct number of intent classes and slot labels. To resolve any resulting inconsistencies, the prompts were rerun and modified if necessary. These are further discussed in Section 4.3.

4.3 Performance Evaluation

The performance of ChatGPT was evaluated using ID accuracy and SF F1-score. Specifically, the span-based micro-averaged F1-score was calculated for SF because it is the standard metric reported in literature [31]. A span includes all words with the same class of a slot label, regardless of their prefix. For example, the labels `B-airline_name` and `I-airline_name` comprise a span of class `airline_name`. The results were compared to the co-interactive transformer by Qin et al. [24]), a SOTA model. The zero-shot performance was benchmarked against other experiments with LLMs, i.e., ChatGPT and Codex [21]. However, the SF metric is changed to token-based macro-averaged F1-score as this was the metric used by Pan et al. [21].

5 RESULTS AND FINDINGS

In this section, we present the performance evaluation results of ChatGPT across various prompt settings and when compared with

Table 1: ATIS Dataset Sample with Intent and Slot Annotation (IOB format)

Utterance	i	want	to	fly	from	baltimore	to	dallas	round	trip
Slot Label	O	O	O	O	O	B-fromloc.city_name	O	B-toloc.city_name	B-round_trip	I-round_trip
Intent Class	atis_flight									

Table 2: Prompt for Joint ID and SF

Section	Prompt
Task Description	You are a joint intent detection and slot filling model. You need to identify the intent and slot labels of a given sentence. The sentence you need to annotate will be given in the next prompt after this.
Intent Class Constraints	First, choose the intent of the sentence from the following intent list: [atis_abbreviation, atis_aircraft, ...]
Slot Label Constraints	Next, annotate the sentence with slots from the following slot label list in IOB tagging: [B-aircraft_code, B-airline_code, ...]
Regulations	<p>Strictly follow the regulations below:</p> <ol style="list-style-type: none"> 1. Give only one intent for the sentence. 2. Give the slot label of every word in the sentence based on IOB tagging. 3. Intents must only come from the provided intent list. 4. Slot labels must only come from the provided slot label list. 5. You need to output the intent and slot labels in JSON format: <pre>{ "intent": "<insert chosen intent>", "slots": [{ "word": "<insert word in the sentence>", "slot_label": "<insert chosen slot label>" }] }</pre> 6. The number of slots should be equivalent to the number of words in the sentence.

other LLMs and SOTA models. We also share our findings on misclassified intents and slot labels from analyzing the output generated by ChatGPT.

5.1 Zero-shot v.s. Few-shot Performance

The performance of ChatGPT across the six (6) prompt settings is presented in Table 3. It can be observed that ID accuracy and span-based micro-averaged SF F1-score generally increase as the number of examples provided to ChatGPT increases, with the 30-prompt setting achieving an ID accuracy of 87.46% and SF F1-score of 89.13%. This indicates that providing more context or examples to ChatGPT can improve its ability to produce the desired output.

However, 1-shot prompting is an exception, obtaining the lowest accuracy score for ID. Similar behavior was observed in the work of Zhong et al. [33] where ChatGPT did not perform well under 1-shot prompting for other NLU tasks. This illustrates the risk of providing only one example to ChatGPT as it may serve as noise that could skew its performance.

It is worth noting that ChatGPT had the steepest increase in SF F1-score under 1-shot prompting. This may be attributed to the multiple slot-value pairs that were given to ChatGPT. In multiple slot-value pairs, each word in an utterance has an associated slot label which provided more context that may have aided ChatGPT in performing the SF task, and thus, the increased performance. It can also be observed that score improvements become gradual upon reaching a certain point (i.e., 10-shot prompting). The SF F1-score of the 20-shot prompting is slightly lower than that of the 10-shot prompting. This may be attributed to the increased length of the prompt, as reported in the study of He et al. [9].

5.2 Comparison with Other Models

The zero-shot performance of ChatGPT was compared with the results reported by Pan et al. [21] using ID accuracy and token-based macro-averaged SF F1-score as seen in Table 4. The evaluation metric used for SF was changed to token-based macro-averaged F1-score as this was the metric used by Pan et al. [21]. This means that the slot F1-score was computed as the mean of the individual F1-scores per token-level slot label, where slot label classes with different prefixes are treated as distinct slot labels.

As can be seen in the table, ChatGPT’s performance using the model version released in January 2024 is better compared to its version released in 2023. However, it performed worse compared to Codex, a 12B parameter GPT model fine-tuned on publicly available code from GitHub [4]. Our work was compared to Codex for a more direct comparison with the results reported by Pan et al. who also evaluated Codex [21].

Table 3: ChatGPT’s Performance in Joint ID and SF

Prompt Setting	ID Accuracy (%)	SF F1-Score (%)
0-shot	76.15	80.07
1-shot	69.09	83.17
5-shot	71.44	85.68
10-shot	83.20	87.20
20-shot	86.45	86.99
30-shot	87.46	89.13

Table 4: Zero-shot Performance Comparison of ChatGPT with Other LLMs

Model	ID Accuracy (%)	SF F1-score (%)
Codex	89.92	57.29
ChatGPT 2023	75.22	15.71
This work (ChatGPT 2024)	76.15	47.32

It is interesting to note that the token-based **macro**-averaged F1-score is much lower compared to the span-based micro-averaged F1-score previously reported in Table 3 for SF. A micro-averaged F1-score takes label imbalance into account by computing the result based on the proportion of every class. On the other hand, a **macro**-averaged F1-score does not consider label imbalance as it simply gets the mean of the F1-score of each class. The slot labels of the ATIS dataset are highly imbalanced, which may have introduced bias to the results. Another factor that contributed to the higher micro-averaged F1-score is with the use of spans. Many of ChatGPT’s mislabels belong to the same span or class, which means that a portion of the errors were semantically related. For example, the ground truth slot label "B-depart_time.period_of_day" was frequently mislabeled as "I-depart_time.period_of_day" which belongs to the same class (see Table 7).

The highest scores obtained by ChatGPT under the 30-shot prompting were compared to results of the co-interactive transformer developed by Qin et al. [24]. As seen in the results in Table 5, it is evident that the SOTA model outperforms ChatGPT. However, it is worth noting that the SOTA model was trained on 4,978 utterances, whereas ChatGPT was only shown 30 sample annotations. Based on the results of varying the prompt settings in Table 3, there is a possibility that ChatGPT may achieve the same performance as the SOTA model using less training data. However, this is yet to be proven empirically as there are other factors affecting ChatGPT’s performance such as input length.

5.3 Misclassified Intents

The top 5 misclassified intents with their corresponding top 5 frequently predicted values are listed in Table 6. It can be observed that the ground truth and predicted values are semantically related. The ground truth intent classes are usually mispredicted with combined intents that start with the same intent class (e.g., "atis_flight" is mispredicted as either "atis_flight#atis_flight_time" or "atis_flight#atis_airfare"). These combined intents were created so that utterances in ATIS will only be labeled with a single

Table 5: ChatGPT’s 30-shot Performance Comparison with a SOTA Model

Model	ID Accuracy (%)	SF F1-score (%)
SOTA Model (Co-interactive Transformer)	98.00	96.10
This work (ChatGPT 2024 with 30-shot Prompting)	87.46	89.13

Table 6: Top 5 Misclassified Ground Truth Intents and their Frequently Predicted Values

Ground Truth Intent	Frequently Predicted Intents
1. atis_flight	atis_flight#atis_flight_time, atis_flight#atis_airfare, atis_flight_time, atis_cheapest, atis_airline#atis_flight_no
2. atis_abbreviation	atis_fare_basis_code, atis_airfare, atis_restriction, atis_airport, atis_meal
3. atis_airfare	atis_airfare#atis_flight, atis_cheapest, atis_flight, atis_flight#atis_flight_time
4. atis_airline	atis_airline#atis_flight_no, atis_airline#atis_flight, atis_airline#atis_flight#atis_flight_no, atis_airline#atis_flight_time
5. atis_flight#atis_airfare	atis_airfare#atis_flight_time, atis_airfare, atis_flight, atis_airfare#atis_flight, atis_flight#atis_flight_time

intent despite encompassing overlapping intents. There is inherent ambiguity in the intent annotations, which diminishes the performance of ChatGPT in ID. Relationships can still be gleaned from the other mispredictions that are not straightforward. For example, the "atis_abbreviation" intent class is usually mispredicted with the "atis_fare_basis_code" intent class, which is similar to abbreviations, or to the types of abbreviations in the queries (e.g., *fare code*, *restriction code*, *airport code*, *meal code*). These show that ChatGPT’s misclassifications are logical and not random.

5.4 Mislabeled Slots

The top 5 mislabeled slots with their corresponding top 5 frequently predicted values are listed in Table 7. Upon inspection, it is apparent that a lot of the mislabeled slots are predicted with slot labels that are more specific but still related (e.g., B-city_name is labeled as "B-toloc.city_name" or "B-fromloc.city_name"). The "O" label was the most mislabeled slot, partly because it has the highest frequency compared to the other slot labels. ChatGPT appears to have a tendency to consider a word as a slot despite not corresponding to any of the predefined slot labels. The mislabels of "B-depart_time.period_of_day" are not straightforward, but still logical. It is usually mispredicted with slot labels that are also pertaining to time, such as "I-depart_time.period_of_day," and "B-depart_time.time". These illustrate that ChatGPT’s errors in SF are sensible and not entirely stochastic.

5.5 Unexpected Behaviors of ChatGPT

Throughout the conduct of the experiments, ChatGPT exhibited unexpected behaviors deviant to the specified instructions in the prompt. These unexpected behaviors can be classified into three:

Table 7: Top 5 Mislabeled Ground Truth Slots and their Frequently Predicted Values

Ground Truth Slot	Frequently Predicted Slots
1. O	B-flight, B-transport_type, B-toloc.city_name, I-transport_type, B-aircraft_code
2. B-city_name	B-toloc.city_name, B-fromloc.city_name, I-toloc.city_name, O, B-stoploc.city_name
3. I-airport_name	I-fromloc.airport_name, I-toloc.airport_name, I-fromloc.city_name, B-fromloc.airport_name, I-stoploc.airport_name
4. I-city_name	I-toloc.city_name, I-fromloc.city_name, O, I-fromloc.airport_name, I-stoploc.city_name
5. B-depart_time.period_of_day	I-depart_time.period_of_day, B-depart_time.time, B-depart_time.period_mod, B-flight_time, B-arrive_time.period_of_day

(1) unknown intent classes and slot labels, (2) incorrect number of slots, and (3) annotating incorrect utterances.

5.5.1 Unknown Intent Classes and Slot Labels. ChatGPT outputted several unknown intent classes and slot labels despite being instructed to only select from the predefined lists specified in the prompt. Table 8 lists the top 5 most frequent unknown intent classes and their corresponding ground truth value/s. It can be seen that the unknown intent classes are mostly more specific versions of the general “atis_flight” intent class. ChatGPT also invented a more specific “atis_abbreviation” intent class, that is, “atis_fare_basis_code”, which is a frequently asked type of abbreviation.

Table 9 shows the top 5 most frequent unknown slot labels and their corresponding ground truth value/s. It can be observed that ChatGPT can infer new slot labels from the words found in an utterance as it invents slot labels for those that should have been tagged as “O” or null. ChatGPT also experiences difficulty in identifying the correct prefix of slot labels and has a tendency to label words with the “I-” prefix instead of “B-” despite it being the first word identified for that slot’s span of text.

5.5.2 Incorrect Number of Slot Labels. We found many instances where ChatGPT outputs an incorrect number of slot labels. We clustered them into four (4) different scenarios as shown in Table 10 and provided an example. To address these scenarios, we included a 6th regulation to our original prompt by explicitly stating that *the number of slot labels outputted must be equal to the number of words*

Table 8: Unknown Intent Classes and Their Corresponding Ground Truth Values

Unknown Intent	Ground Truth Intent
1. atis_flight#atis_flight_time	atis_flight, atis_flight#atis_airfare, atis_flight_no, atis_airfare
2. atis_airfare#atis_flight	atis_flight, atis_flight_no#atis_airline, atis_flight_no, atis_city, atis_flight#atis_airline
3. atis_flight#atis_airline	atis_flight, atis_city
4. atis_airline#atis_flight	atis_flight, atis_city, atis_airline
5. atis_fare_basis_code	atis_abbreviation

Table 9: Unknown Slot Labels and their Corresponding Ground Truth Value/s

Unknown Slot	Ground Truth Slot
1. B-flight	O, B-flight, I-round_trip
2. I-aircraft_code	B-aircraft_code, O
3. I-depart_date.day_name	O, B-depart_date.day_name, B-day_name
4. B-capacity	O
5. I-toloc.state_code	B-toloc.state_code, B-state_code

in the given sentence. When this did not fully resolve the problem, the explicit instruction, “Do not leave out any word until the end of the prompt” is sometimes added to the prompt. Some occurrences of concatenations were separated resulting in inconsistent output. For this, the prompt was modified to include a 7th regulation, stating that “contractions must not be separated”.

5.5.3 Annotating Incorrect Utterances. In the 20-shot and 30-shot prompting, ChatGPT exhibited several instances where it annotated a different utterance from what was provided by the user. These utterances were mostly previous utterances that were annotated by it already. This indicates that longer prompts could trigger unexpected behaviors in ChatGPT. This issue was resolved by rerunning the prompt.

6 DISCUSSION

LLMs have shown competitive performance in a variety of tasks across different domains using zero-shot or few-shot prompting against SOTA models [23]. Different prompting strategies can also enhance the responses of LLMs such as chain-of-thought, self-question, clue and reasoning prompting, and persona specification [12, 18]. We found three main factors that may affect ChatGPT’s performance in the joint ID and SF task.

Table 10: Scenarios Causing Incorrect Number of Outputted Slot Labels and their Examples

Scenario	Utterance	Wrong Annotation Example
A new word is annotated	saturday flight on american airlines from milwaukee to phoenix	{'intent': 'atis_flight', 'slots': [{'word': 'saturday', 'slot_label': 'B-depart_date.day_name'}, {'word': 'flight', 'slot_label': 'O'}, {'word': 'on', 'slot_label': 'O'}, {'word': 'american', 'slot_label': 'B-airline_name'}, {'word': 'airlines', 'slot_label': 'I-airline_name'}, {'word': 'from', 'slot_label': 'O'}, {'word': 'milwaukee', 'slot_label': 'B-fromloc.city_name'}, {'word': 'to', 'slot_label': 'O'}, {'word': 'phoenix', 'slot_label': 'B-toloc.city_name'}, {'word': 'to', 'slot_label': 'O'}]}
A word is repeatedly annotated	what types of ground transportation are available in denver	{'intent': 'atis_ground_service', 'slots': [{'word': 'what', 'slot_label': 'O'}, {'word': 'types', 'slot_label': 'O'}, {'word': 'of', 'slot_label': 'O'}, {'word': ' ground ', 'slot_label': 'O'}, {'word': ' ground ', 'slot_label': 'B-transport_type'}, {'word': 'transportation', 'slot_label': 'I-transport_type'}, {'word': 'are', 'slot_label': 'O'}, {'word': 'available', 'slot_label': 'O'}, {'word': 'in', 'slot_label': 'O'}, {'word': 'denver', 'slot_label': 'B-city_name'}]}
A word is not annotated	what does ua mean	{'intent': 'atis_abbreviation', 'slots': [{'word': 'what', 'slot_label': 'O'}, {'word': 'does', 'slot_label': 'O'}, {'word': 'ua', 'slot_label': 'B-abbreviation'}]}
Contractions are separated	list the distance in miles from new york's la guardia airport to downtown new york city	{'intent': 'atis_distance', 'slots': [{'word': 'list', 'slot_label': 'O'}, {'word': 'the', 'slot_label': 'O'}, {'word': 'distance', 'slot_label': 'O'}, {'word': 'in', 'slot_label': 'O'}, {'word': 'miles', 'slot_label': 'O'}, {'word': 'from', 'slot_label': 'O'}, {'word': 'new', 'slot_label': 'B-fromloc.city_name'}, {'word': ' york ', 'slot_label': 'I-fromloc.city_name'}, {'word': ' s ', 'slot_label': 'O'}, {'word': 'la', 'slot_label': 'B-fromloc.airport_name'}, {'word': 'guardia', 'slot_label': 'I-fromloc.airport_name'}, {'word': 'airport', 'slot_label': 'I-fromloc.airport_name'}, {'word': 'to', 'slot_label': 'O'}, {'word': 'downtown', 'slot_label': 'B-toloc.city_name'}, {'word': 'new', 'slot_label': 'I-toloc.city_name'}, {'word': 'york', 'slot_label': 'I-toloc.city_name'}, {'word': 'city', 'slot_label': 'I-toloc.city_name'}]}

Contextual information. The number of examples provided to ChatGPT influences its performance in the joint task. The results showed that providing ChatGPT with more examples generally improves its performance for both ID and SF. However, the improvement becomes gradual upon reaching a certain point. It is also worth mentioning that the 1-shot performance is worse compared to zero-shot for ID, which shows ChatGPT's sensitivity to noise.

Metrics for evaluation. Metric selection is another crucial factor to obtain an accurate evaluation of ChatGPT's performance, as evidenced by the big gap between the span-based micro-averaged slot F1 score and the token-based macro-averaged slot F1 score in this study. The former metric accounts for data imbalance and semantically related slot labels (i.e., belonging to the same class), and is the standard used in literature [31].

Unexpected behaviors. Lastly, ChatGPT tends to exhibit unexpected behaviors that deviate from the specified instructions. These unexpected behaviors include generating unknown outputs, outputting incorrect number of slot labels, and annotating previous prompts instead of the current one. These unexpected behaviors were observed more frequently with longer prompts, which shows

that prompt length has an impact on ChatGPT's performance. ChatGPT also demonstrated unexpected behaviors in other tasks such as emotion classification [13] and code generation [1].

7 CONCLUSION

In this study, we compared ChatGPT's performance on the joint ID and SF task under zero-shot and different few-shot (i.e., 1-shot, 5-shot, 10-shot, 20-shot, and 30-shot) prompt settings. Results showed that giving more examples to ChatGPT generally improves its performance, but becomes gradual upon reaching a certain threshold. Future work could investigate the number of examples at which ChatGPT's performance begins to decline on the joint ID and SF task. This would also reveal whether ChatGPT can be at par, or even surpass the performance of SOTA models by providing more examples to it. This raises the need for a mechanism to handle long prompts, such as compression, due to the token limits of LLMs. This may also mitigate ChatGPT's performance issues associated with long prompts.

REFERENCES

- [1] Alessio Buscemi. 2023. A Comparative Study of Code Generation using ChatGPT 3.5 across 10 Programming Languages. *ArXiv abs/2308.04477* (2023). <https://api.semanticscholar.org/CorpusID:260735527>
- [2] Asli Celikyilmaz and Dilek Hakkani-Tur. 2012. A joint model for discovery of aspects in utterances. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Haizhou Li, Chin-Yew Lin, Miles Osborne, Gary Geunbae Lee, and ong C. Park (Eds.). Association for Computational Linguistics, Jeju Island, Korea, 330–338. <https://aclanthology.org/P12-1035>
- [3] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.* (jan 2024). <https://doi.org/10.1145/3641289> Just Accepted.
- [4] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *arXiv:2107.03374* [cs.LG]
- [5] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips Voice Platform: An embedded spoken language understanding system for private-by-design voice interfaces. *arXiv:1805.10190* [cs.CL]
- [6] Jacob Devlin, Ming-Wei Chang, and Kristina Toutanova Kenton Lee. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. <https://doi.org/10.18653/v1/N19-1423>
- [7] Maujama Firdaus, Shobhit Bhatnagar, Asif Ekbal, and Pushpak Bhattacharyya. 2018. *Intent detection for spoken language understanding using a deep ensemble model: 15th Pacific Rim International Conference on Artificial Intelligence, Nanjing, China, August 28–31, 2018, Proceedings, Part I*. 629–642. https://doi.org/10.1007/978-3-319-97304-3_48
- [8] Daniel Guo, Gokhan Tur, Wen tau Yih, and Geoffrey Zweig. 2014. Joint semantic utterance classification and slot filling with recursive neural networks. In *2014 IEEE Spoken Language Technology Workshop (SLT)*. 554–559. <https://doi.org/10.1109/SLT.2014.7078634>
- [9] Mutian He and Philip N. Garner. 2023. Can ChatGPT detect intent? Evaluating large language models for spoken language understanding. In *Proc. INTERSPEECH 2023*. 1109–1113. <https://doi.org/10.21437/Interspeech.2023-1799>
- [10] Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*. <https://aclanthology.org/H90-1021>
- [11] Minwoo Jeong and Gary Geunbae Lee. 2008. Triangular-chain conditional random fields. *IEEE Transactions on Audio, Speech, and Language Processing* 16 (2008), 1287–1302. <https://api.semanticscholar.org/CorpusID:14617149>
- [12] Katikapalli Subramanyam Kalyan. 2024. A survey of GPT-3 family large language models including ChatGPT and GPT-4. *Natural Language Processing Journal* 6 (2024), 100048. <https://doi.org/10.1016/j.nlp.2023.100048>
- [13] Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, Anna Kocoń, Bartłomiej Koptyra, Wiktoria Mieszczyszko-Kowszewicz, Piotr Milkowski, Marcin Oleksy, Maciej Piasecki, Lukasz Radliński, Konrad Wojtasik, Stanisław Woźniak, and Przemysław Kazienko. 2023. ChatGPT: Jack of all trades, master of none. *Information Fusion* 99 (2023), 101861. <https://doi.org/10.1016/j.inffus.2023.101861>
- [14] Sheetal Kusal, Shruti Patil, Jyoti Choudrie, Ketan Kotecha, Sashikala Mishra, and Ajith Abraham. 2022. AI-Based conversational agents: A scoping review from technologies to future directions: Conversational agents. *IEEE Access* 10 (Aug 2022), 92337–92356. <https://doi.org/10.1109/ACCESS.2022.3201144>
- [15] Bing Liu and Ian R. Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, Nelson Morgan (Ed.). ISCA, 685–689. <https://doi.org/10.21437/INTERSPEECH.2016-1352>
- [16] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A comprehensive overview of large language models.
- [17] OpenAI. [n. d.]. <https://platform.openai.com/docs/quickstart?context=python>
- [18] OpenAI. [n. d.]. <https://platform.openai.com/docs/guides/prompt-engineering>
- [19] OpenAI. [n. d.]. <https://platform.openai.com/docs/guides/text-generation>
- [20] OpenAI. 2023. <https://openai.com/blog/new-models-and-developer-products-announced-at-devday#OpenAI>
- [21] Wenbo Pan, Qiguang Chen, Xiao Xu, Wanxiang Che, and Libo Qin. 2023. A preliminary evaluation of chatGPT for zero-shot dialogue understanding. *arXiv:2304.04256* [cs.CL]
- [22] Jangwon Park. 2021. Jointbert: Pytorch implementation of Jointbert: “Bert for joint intent classification and slot filling”. <https://github.com/monologg/JointBERT/tree/master>
- [23] Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is ChatGPT a general-purpose natural language processing task solver?. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Singapore, 1339–1384. <https://doi.org/10.18653/v1/2023.emnlp-main.85>
- [24] Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. 2021. A co-interactive transformer for joint slot filling and intent detection. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 8193–8197. <https://doi.org/10.1109/ICASSP39728.2021.9414110>
- [25] Partha Pratim Ray. 2023. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems* 3 (2023), 121–154. <https://doi.org/10.1016/j.iotcps.2023.04.003>
- [26] Ted Sanders. 2023. How to format inputs to CHATGPT models. https://cookbook.openai.com/examples/how_to_format_inputs_to_chatgpt_models
- [27] Nahida Shafi and Manzoor Ahmed Chachoo. 2023. Fine-Tuned BERT with attention-based Bi-GRU-CapsNet framework for joint intent recognition and slot filling. In *2023 International Conference on Advancement in Computation Computer Technologies (InCACCT)*. 369–374. <https://doi.org/10.1109/InCACCT57535.2023.10141744>
- [28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., 1–11. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fb053c1c4a845aa-Paper.pdf
- [29] Haifeng Wang, Jiwei Li, Hua Wu, Eduard Hovy, and Yu Sun. 2023. Pre-Trained language models and their applications. *Engineering* 25 (2023), 51–65. <https://doi.org/10.1016/j.eng.2022.04.024>
- [30] Ye-Yi Wang. 2010. Strategies for statistical spoken language understanding with small amount of data - an empirical study. In *Proceedings of Interspeech* (proceedings of interspeech ed.). International Speech Communication Association. <https://www.microsoft.com/en-us/research/publication/strategies-for-statistical-spoken-language-understanding-with-small-amount-of-data-an-empirical-study/>
- [31] Henry Weld, Xiaoqi Huang, Siqun Long, Josiah Poon, and Soyeon Caren Han. 2022. A survey of joint intent detection and slot-filling models in natural language understanding. *Comput. Surveys* 55, 8 (2022), 1–38. <https://doi.org/10.1145/3547138>
- [32] Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular CRF for joint intent detection and slot filling. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding* (2013), 78–83. <https://api.semanticscholar.org/CorpusID:10532715>
- [33] Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. 2023. Can ChatGPT understand too? A comparative study on ChatGPT and fine-tuned BERT. *arXiv:2302.10198* [cs.CL]

PERFORMANCE EVALUATION: TRIPLET LOSS IMPLEMENTATION ON ALEXNET WITH MTCNN IN DETECTING DEEPPFAKE

Rich Tristan Lim
Department of Computer,
Information Sciences, and
Mathematics
University of San Carlos
Cebu City, Philippines
richlim16@gmail.com

Angie Ceniza-Canillo,
PHD
Department of Computer,
Information Sciences, and
Mathematics
University of San Carlos
Cebu City, Philippines
amceniza@usc.edu.ph

Raymond Anthony Aya-ay
Department of Computer,
Information Sciences, and
Mathematics
University of San Carlos
Cebu City, Philippines
raymond.p.ayaay@gmail.com

ABSTRACT

Deep fakes are media, usually in the form of a video, that have been altered to change certain features from the original video, such as swapping the face of the subjects to make them appear as someone else, this form of deep fake is also known as FaceSwap. This is done by using an existing video and replacing the actor with someone else's appearance or likeness. They are created using artificial intelligence, where real images are fed to a system and are trained by utilizing two parts - one which creates fake images and the other which spots the fakes until it cannot distinguish between the real and fake. This paper analyzes an already existing deepfake detection system known as AlexNet, aided by a Multi-Task Cascaded Convolutional Neural Network, and seeks to find the effects of implementing a Triplet Loss Network. The system is given a video, which will be preprocessed and analyzed to be classified as either deep fake or authentic. Training and testing of the machine is to be done over pre-existing datasets, namely: the Celeb-DF dataset, the DeepFake Detection Challenge dataset, and the FaceForensics++ dataset. After several iterations of testing, the authors of this paper recorded the following results from applying different loss functions to the proposed model; Binary Cross Entropy yielded an AUC score of 75.58%, Semi Hard Triplets yielded a score of 72.04%, Contrastive Loss yielded a score of 62.37%, and lastly, Sparse CategoricalCross Entropy yielded a score of 55.58%.

The authors of this paper will then compare the results to measure the effects of implementing the Triplet Loss Network to the base system of AlexNet with MTCNN. This will be done by comparing the true positive rate over the false positive rate of the models.

CCS CONCEPTS

- Software and its engineering
- Software creation and management
- Software verification and validation
- Empirical software validation

KEYWORDS

Deep fake detection, deep fakes, Deep Neural Network, MTCNN, Triplet Loss

ACM Reference format:

Rich Tristan Lim, Angie Ceniza-Canillo and Raymond Anthony Aya-ay. 2024. A PERFORMANCE EVALUATION: TRIPLET LOSS IMPLEMENTATION ON ALEXNET WITH MTCNN IN DETECTING DEEPPFAKES. In *Proceedings of Philippine Computing Science Congress (PCSC2024)*. Laguna, Philippines, 6 pages.

1 Introduction

Deepfakes generally refer to AI-generated videos where the identities of the subjects have been swapped for those of another person. These videos were originally created with conventional computer graphics methods, but due to the recent advances in deep learning networks and the considerable increase in the computing power of personal computers, newer deep fake videos are easier to make and are much harder to detect due to their continued evolution and improvement. In particular, Generative Adversarial Networks (GANs) have been used to generate newer generations of deep fake videos. While deepfakes can be used for beneficial purposes, they can also be weaponized. Such deep fakes can be used to spread misinformation, propaganda, and fake news in general. One of the earliest examples of widespread deep fakes is the collection of pornographic videos of celebrities posted on Reddit (Harris, 2018), where the videos had the faces of celebrities superimposed into the pornographic videos to make it look like the celebrities were the main subjects of the videos.

Such fraudulent videos have the potential for grave consequences, such as discrediting institutions or even political candidates, thereby tipping electoral outcomes (Chesney & Citron, 2019) and damaging international relations.

Having been inspired by previous studies on this topic, the authors of this paper sought to combine two existing studies, namely, a study in 2020 by Xie, Chatterjee, Liu, Roy, & Kossi that used a modified version of a convolutional neural network (CNN) called AlexNet, and another study in 2020 by Bhavsar, Kumar, & Verma that used the Xception architecture with MTCNN and applied Triplet Loss to the embeddings. This research aims to find the effects of applying the Triplet Loss Network to AlexNet with MTCNN, as well as understanding why the results are such.

2 Related Literature

2.1 Generating Deep Fakes

Major improvements and advances in computer graphics, computer vision, and machine learning have led to the development of deep fake image, video, and audio as well as their continuous improvement. Due to the fact that AI-synthesized content is relatively new (Agarwal et al. 2019), there have been continuous improvements in the development. For example, a study by Yuezun, Ming-Ching, and Siwei (2018) made the observation that there was an irregularity in the blinking done by individuals in face-swap deep fakes due to the fact that the training data used to synthesize faces usually did not depict the person with their eyes closed. It did not take long after this study was made public for synthesis techniques to make the necessary changes to render this method of detecting deep fakes less effective.

Generative Adversarial Networks (GANs) are a form of deep neural networks for both supervised and semi-supervised learning (Creswell et al., 2018) that has been used in generating deep fakes. These models require a large set of training data to create deep fake media, and the larger the training set is the more realistic and indistinguishable the result will be. GANs have two neural network components: a generator and a discriminator. These two can be thought of as an art forger and an art expert. The model uses the generator to train on the training set provided to generate the deep fake. This is then given to the discriminator which has been trained to differentiate the real data from the fake data (Almars, 2021). However, the drawback of requiring large amounts of training data is offset by the large amount of publicly accessible data online, especially for public figures and celebrities. Although deepfakes usually require a large number of images to create a realistic forgery, techniques have already been

developed where one can generate a fake video by feeding it only one photo such as a selfie (Westurlund, 2019).

2.2 Detecting Deep Fake Videos

While there are potentially beneficial applications for deep fake technology, because of the fact that videos of an individual have a significant impact on their image and reputation, they can also be weaponized in ways that far outweigh their potential benefits (Citron & Chesney, 2019). The potential threats can range from revenge porn, to a politician saying outrageous or controversial things, causing political or religious tensions between countries, or even of a company official making statements of the company to affect the outcome of the stock market (Ngyuen et al., 2019). Due to the potential damage, as well as how rapid such videos can spread in the current digital environment.

There are multiple methods of how a DeepFake can be detected. General Network-based Methods regard detection as a frame-level classification task which is finished by CNNs. Temporal Consistency-based Methods identify DeepFakes by detecting inconsistencies between adjacent frames due to the defects of the algorithm. Visual Artifacts-based Methods use the intrinsic image discrepancies found in blending boundaries, called artifacts. Camera Fingerprints-based Methods use the different traces that are left by devices in captured images. Which helps acknowledge that faces and background images are from different devices. In DeepFake generation, it is difficult to synthesize humans with believable behavior, thus these biological signals are extracted to detect DeepFake videos, this is an example of a Biological Signals-based Method (Yu et al., 2021). These methods can be categorized to Low-level Approaches and High-Level Approaches.

2.3 Alexnet

AlexNet is a Convolutional Neural Network that contains many layers. For this study, a modified lighter version of AlexNet based on a study by Daniel, Chatterjee, Liu, & Roy (2020) will be used. Instead of the more in depth model, three convolutional layers, three max-pooling layers, one flatten later, one dense layer, one activation layer, and an optional dropout layer will be used.

Layer (type)	Output Shape	Param #
conv2d_19 (Conv2D)	(None, 32, 124, 124)	832
max_pooling2d_19 (MaxPooling)	(None, 32, 41, 41)	0
conv2d_20 (Conv2D)	(None, 64, 37, 37)	51264
max_pooling2d_20 (MaxPooling)	(None, 64, 12, 12)	0
conv2d_21 (Conv2D)	(None, 128, 8, 8)	204928
max_pooling2d_21 (MaxPooling)	(None, 128, 2, 2)	0
flatten_7 (Flatten)	(None, 512)	0
dense_13 (Dense)	(None, 128)	65664
activation_13 (Activation)	(None, 128)	0
dropout_7 (Dropout)	(None, 128)	0
dense_14 (Dense)	(None, 5)	645
activation_14 (Activation)	(None, 5)	0

Total params: 323,333
 Trainable params: 323,333
 Non-trainable params: 0

Figure 1. AlexNet Structure

2.4 Multi-task Cascaded Networks

Multi-task Cascaded CNNs (MTCNN) extracts faces from frames using face detection and alignment to boost performance. It mainly consists of three parts: Proposal networks, which detects faces across multiple resolutions by generating a list of candidate windows and using it for classifying which are the faces and estimating bounding box regression vectors and non-maximum suppression. Refine net rejects the false candidates. Finally, the output network will output five facial landmarks (Xiang & Zhu, 2017).

2.5 Triplet Network

Triplet network is a type of metric learning that requires three sample input images, which are the anchor sample (A), the positive sample (P) which is of the same class as the anchor, and the negative sample (N) which is of a different class. Triplet loss is used to calculate the loss of estimation results of the three samples. The network minimizes the distance between P and A, and maximizes the distance between the N and A, this formula is shown in Figure 2.

$$\|f(x)\|_2 = 1$$

$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

where $[z]_+ = \max(z, 0)$

Figure 2. Triplet Loss Formula

The first item is the distance between the anchor and positive and the second item is the distance between the anchor and negative. The value of the first item is learned to be smaller while the second item is bigger. If the difference is smaller than minus alpha, the loss would become zero and the network parameters would not be updated at all (Hoffer & Ailon, 2014).

In a study by Bhavsar, Kumar, & Verma (2020) the triplet loss network was implemented in which they used semi-hard triplets, where the negative is farther from the anchor than the positive, but still produces a positive loss. In this study, the negative triplet is a deepfake while the positive and anchor triplets are genuine. The researchers used FaceNet to generate face embeddings of a 512 dimension vector and applied triplet loss to those embeddings. From there the network learns the discriminative features to the embeddings of the original and the manipulated faces separately.

3 Technical Background

3.1 Deepfake

It is a generated media in which a person in an existing image or video is replaced with someone else's likeness. While the act of faking content is not new, deep fakes leverage powerful techniques from machine learning and artificial intelligence to manipulate or generate visual and audio content to appear more authentic to the viewer. (Kietzmann, 2020). The most significant problem with deep fakes is the scope, scale, and sophistication of the technology, and how easy it is to generate, since almost anyone with a computer can create convincing fake videos with free applications and online services (Fletcher, 2018).

3.2 Deep Learning

It is a category of machine learning algorithms that uses numerous layers to gradually extract higher-level features from the raw input. These neural networks attempt to simulate the behavior of the human brain. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy (IBM Cloud Education, 2020). For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.

4 Methodology

4.1 Research Environment and Respondents

The data used in this study is based on a previous research by Li et al. (2020) entitled Celeb-DF where the authors found that several datasets being used by studies on deep fake detection had a considerable amount of data that was too easy to detect and, therefore, were not representative of current deep fake videos that one is more likely to find. As a result of this, they created a dataset which was more sophisticated and proved to result in lower average detections when compared to several other popular datasets. Similar results were also shown in a later study by Tolosona et al. (2020), in which they compared several deep fake detectors against different datasets and found that Celeb-DF consistently had lower detection scores along with the DFDC database (Dolhansky et al., 2019), which is a public database released by Meta in collaboration with other companies such as Microsoft, Amazon, and MIT to be used in their Deep Fake Detection Challenge which was created to boost development in this field. FaceForensics++ is another modern dataset that was commonly used among recent studies in deep fake detection and will serve as another benchmark for comparison of results.

4.2 Research Instrument or Sources of Data

This study uses two recent databases of the 2nd generation of deep fakes, namely Celeb-DF, DFDC, and one database of the 1st generation, FaceForensics++. These three were chosen because they are the latest databases for deep fake videos and have also proven to be more sophisticated and challenging for deep fake detectors (Li et al., 2020; Deepfake Detection Challenge Dataset, 2020). DFDC will consist of two versions, one of which contains around 5,000 videos that feature two facial modification algorithms and the other version contains around 124,000 videos that feature eight facial modification algorithms. Celeb-DF will contain around 5,600 high-quality videos of celebrities and

FaceForensics++ will contain 1,000 real videos taken from Youtube and will have 4 types of deep fake videos of varying qualities.

4.3 Research Procedure

4.3.1 Gathering of Data. The general objective of this study is to measure the effects of implementing triplet loss to the CNN, particularly in classifying deep fake videos from authentic videos that have not been tampered in a way that changes the identity of the subjects in the video. The datasets used were developed for the purpose of providing a more challenging and sophisticated training set for deep fake detectors to improve the effectiveness of following studies in this field of study (Li et al., 2020). Prior studies have proven these datasets to be the most challenging to test against when compared to other popular datasets.

4.3.2 Treatment of Data. All the datasets utilized in this study are composed of videos which will be extracted into frames so the models will focus more on the facial imagery. The OpenCV package will be used to read each video file and extract every 5th frame of the video, as well as reducing the resolution of each frame. An MTCNN to extract faces out of the frames and place them in the center of alignment .

4.4 Conceptual Framework

The deep learning model used in this research is based on a modified, lighter version of AlexNet by Xie, Chatterjee, Liu, Roy, & Kossi (2020) to detect deep fake videos from real videos. Initially, the faces are extracted from the frames of the videos via MTCNN to place five landmarks for each face. Afterwards the authors of this paper will take the processed frames and generate a 512 dimension vector using the modified AlexNet. One of the models will apply semi-hard triplets to the embeddings generated by AlexNet and the other model will not. The results will be measured and compared to determine the effects of applying the Triplet Loss Network. The main purpose of this study is to find the benefits and drawbacks of applying the Triplet Loss Network to the deep fake detection system.

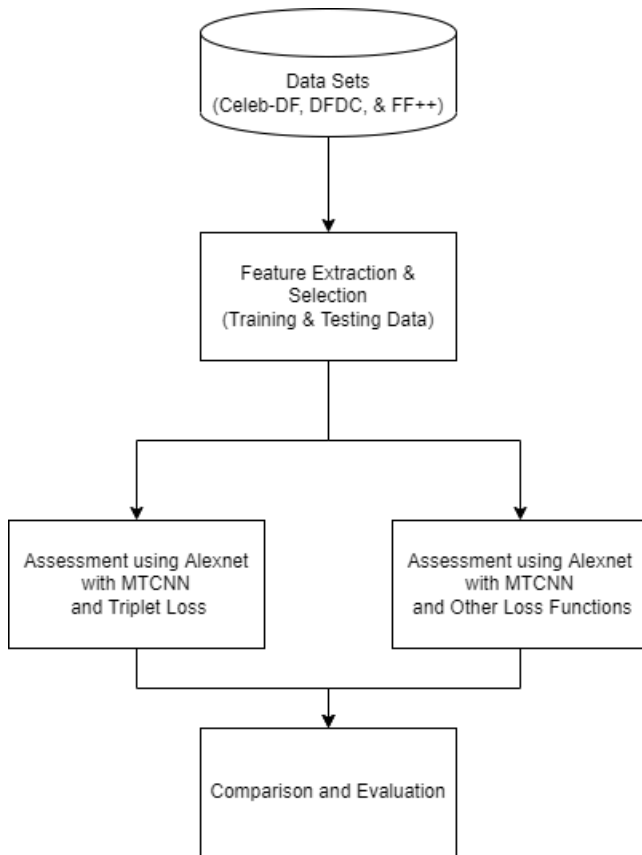


Figure 3. Conceptual Framework

5. Results and Analysis

The authors of this paper started testing each dataset individually to see how they affected the results of the model. The results in Figure 4 show that Celeb-DF V2 resulted in higher performance scores in accuracy, whereas FF++ and DFDC yielded closer scores in relation to each other. However, all the average scores were within 20% of each other, therefore, the researchers concluded that this difference would not significantly skew the results of the model.

With a combination of all three datasets at 5 frames per second for a total of 450,000 frames, however no significant difference in performance was found when compared to a smaller subset of 75,000 frames, thus, succeeding tests were performed with the smaller number of frames (75,000 frames) with 25,000 frames coming from each dataset.

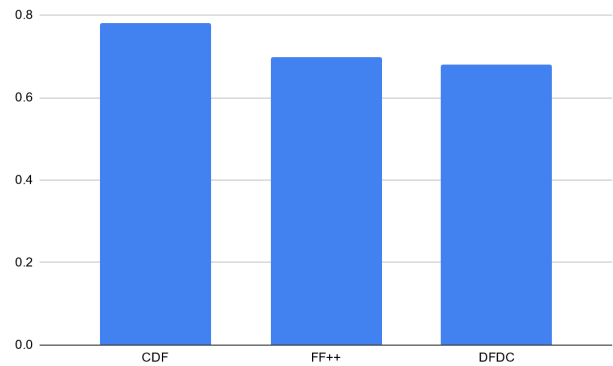


Figure 4. Performance Scores in Accuracy of Each Dataset

In the training of the model all frames were extracted with MTCNN applied and the model used was AlexNet in all tests. The different Loss functions were tested and the results for their average accuracy are recorded in the bar chart illustrated below in Figure 5. The results show that the one outlier in the results is contrastive loss, with a significantly lower accuracy score compared to the other loss functions, with an accuracy of 69.54%, whereas the other three are within a 5% difference. The highest accuracy was achieved by using Binary Cross Entropy with an accuracy of 89.08%, followed by Sparse Categorical Entropy with an accuracy of 87.44%, and Semi Hard Triplets achieved an accuracy of 85.71%.

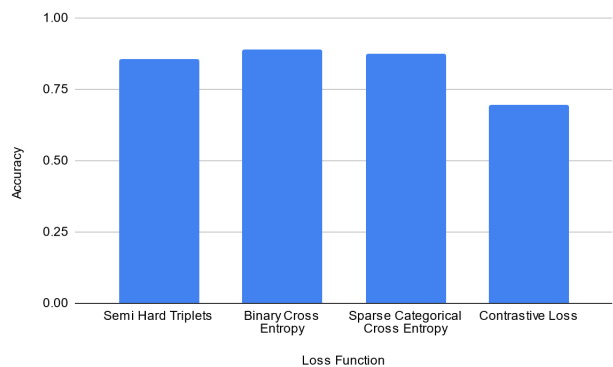


Figure 5. Average Accuracy Scores of Each Loss Function

When the results were finalized with AUC scores as shown in Figure 6 below, it is seen that Binary Cross Entropy still maintains the highest performance. However Sparse Categorical Cross Entropy is now the lowest performing, with Semi Hard Triplets and Contrastive Loss having better AUC scores, meaning that although using Sparse Categorical Cross Entropy for loss yielded in higher accuracy, Semi Hard Triplets and Contrastive Loss actually, this may hint at poorer positive class classification in comparison to negative classification for Sparse Categorical Cross Entropy, as ROC scores are biased to positive classes.

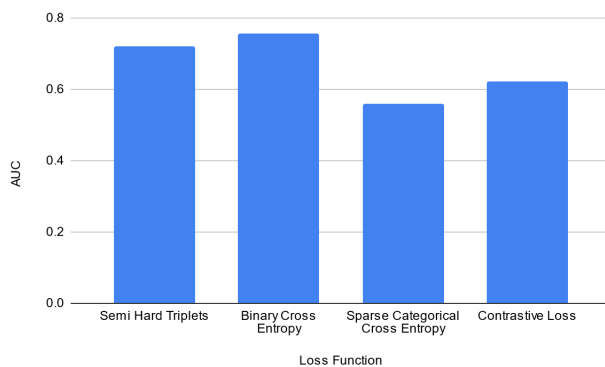


Figure 6. Average AUC Scores of Each Loss Function

6. Conclusion and Recommendation

In this work, the authors of this paper presented a deep study for classification of deep fake videos using a benchmark testing procedure with MTCNN and Alexnet Deep Learning Model, to collect and analyze standardized results that give a better understanding of possible improvements in the field of deep fake detection with differing loss functions. The effects of different datasets were studied and how they affect the performance results as well as testing the results of having a combined dataset. For future work, our aim is to use a more modern dataset and test it on the newer generation of deep fake videos to evaluate their effectiveness on newer methods.

In a previous study by Hoffer & Ailon (2014), they stated that triplet loss, despite not specializing in classification, does perform well when compared to other models specific for classification tasks. Our findings prove that triplet loss does work well, but is outperformed by loss functions specific to binary classification tasks such as deepfake detection. Therefore, the authors of this paper conclude that triplet loss works well for classification tasks, but its method of learning differences does not benefit in the task of binary

classification. Instead it is better to use loss functions specific to binary classification tasks.

ACKNOWLEDGMENTS

Firstly, the authors of this paper would like to thank God, their families, friends, and everyone who has provided support and guidance to them in one way or another.

REFERENCES

- [1] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, H. Li: Protecting world leaders against deep fakes. In: IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Media Forensics. pp. 38–45 (2019)
- [2] A.M. Almars (2021). Deepfakes Detection Techniques Using Deep Learning: A Survey. *Journal of Computer and Communications*, 9(5), 20-35. scirp.org. doi: 10.4236/jcc.2021.95003
- [3] R. Chesney, & D. Citron. (2019). Deepfakes and the new disinformation war: The coming age of post-truth geopolitics. *Foreign Aff.*, 98, 147.
- [4] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, & A.A. Bharath (2018). Generative Adversarial Networks: An Overview. *IEEE Signal Processing Magazine*, 35(1), 53-65. doi: 10.1109/MSP.2017.2765202
- [5] B. Dolhansky, R. Howes, B. Pfau, N. Baram, & C.C. Ferrer (2019). The Deepfake Detection Challenge (DFDC) Preview Dataset. arXiv Preprint. arXiv:1910.08854
- [6] J. Fletcher (2018). Deepfakes, Artificial Intelligence, and Some Kind of Dystopia: The New Faces of Online Post-Fact Performance. *Theatre Journal*, 70(4), 455-471. Project MUSE. doi:10.1353/tj.2018.0097
- [7] D. Harris (2018). Deepfakes: False pornography is here and the law cannot protect you. *Duke L. & Tech. Rev.*, 17, 99.
- [8] E. Hoffer. & N. Ailon (2014). Deep metric learning using Triplet network. arXiv. <https://arxiv.org/abs/1412.6622>
- [9] A. Kumar, A. Bhavsar, & R. Verma (2020). Detecting deepfakes with metric learning. In 2020 8th international workshop on biometrics and forensics (IWBF)(pp.1-6).IEEE.
- [10] D. Xie, P. Chatterjee, Z. Liu, K. Roy and E. Kossi, "DeepFake Detection on Publicly Available Datasets using Modified AlexNet," 2020 IEEE Symposium Series on Computational Intelligence (SSCI), Canberra, ACT, Australia, 2020, pp. 1866-1871, doi: 10.1109/SSCI47803.2020.9308428.
- [11] R. Tolosona, S. Romero-Tapiador, J. Fierrez, & R. Vera-Rodriguez (2020). DeepFakes Evolution: Analysis of Facial Regions and Fake Detection Performance. *Biometrics and Data Pattern Analytics - BiDA Lab*, Universidad Autonoma de Madrid.
- [11] M. Westurlund (2019). The Emergence of Deepfake Technology: A Review. *Technology Innovation Management Review*, 9(11), 39-52. <http://doi.org/10.22215/timreview/1282>
- [12] J. Xiang, & G. Zhu (2017). Joint Face detection and Facial Expression Recognition with MTCNN. 2017 4th International Conference on Information Science and Control Engineering. doi:10.1109/ICISCE.2017.95
- [13] Y. Li, M. -C. Chang and S. Lyu, "In Ictu Oculi: Exposing AI Created Fake Videos by Detecting Eye Blinking," 2018 IEEE International Workshop on Information Forensics and Security (WIFS), Hong Kong, China, 2018, pp. 1-7, doi: 10.1109/WIFS.2018.8630787.
- [14] P. Yu, Z. Xia, J. Fei, & Y. Lu (2021). A Survey on Deepfake Video Detection. *IET Biometrics*, 10(6), 607-624. <https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/bme2.12031>

Summarization-Driven Collaborative Filtering for Explainable Recommendation

Reinald Adrian Pugoy
University of the Philippines Open University
Los Baños, Laguna, Philippines
rdpugoy@up.edu.ph

Hung-Yu Kao
National Cheng Kung University
Tainan City, Taiwan
hykao@mail.ncku.edu.tw

ABSTRACT

Neural review-based recommender systems often lack explainability due to the black-box nature of neural networks. This paper introduces SUMMER, a novel, accurate, and explainable collaborative filtering (CF) framework. SUMMER generates summary-level explanations for each item and user, mirroring the style of real-life explanation texts. This integration of summarization into the CF architecture not only improves explainability but also enhances the encoding of users and items, boosting recommendation performance. SUMMER is the first summarization-driven CF model capable of generating both extractive and abstractive explanations, offering flexibility in explanation generation. We further argue for reformulating explainability as unsupervised summarization, recognizing the impracticality of obtaining ground-truth explanations for every item and user. Our experiments demonstrate SUMMER's strong rating prediction accuracy, comparable to other state-of-the-art approaches. Moreover, our explainability study reveals a user preference for extractive summary-level explanations.

KEYWORDS

Collaborative Filtering, Explainable Recommender Systems, Unsupervised Summarization, Rating Prediction

ACM Reference Format:

Reinald Adrian Pugoy and Hung-Yu Kao. 2024. Summarization-Driven Collaborative Filtering for Explainable Recommendation. In *Proceedings of Philippine Computing Science Congress (PCSC2024)*. Laguna, Philippines, 8 pages.

1 INTRODUCTION

Recommender systems have become indispensable tools in navigating the vast landscape of online information. Widely integrated into web applications, they revolutionize how users discover and assess products and services across various domains, from shopping to entertainment and news consumption [3, 35]. Collaborative filtering (CF) lies at the heart of these systems, aiming to accurately capture user preferences and item characteristics. While early CF models relied solely on numeric ratings, this approach oversimplifies the nuanced nature of user preferences and suffers from sparse rating matrices, impairing accuracy [16, 21, 38]. To address these challenges, researchers have turned to review texts as a valuable

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

PCSC2024, May 2024, Laguna, Philippines

© 2024 Copyright held by the owner/author(s).

Table 1: Illustration of the different explanation types.

A review-level explanation is simply the highest weighted review, and a word-level explanation is composed of underlined words with the highest attention scores. Our proposed summary-level explanations closely resemble real-life explanations by deriving information from multiple reviews.

Reviews Received by an Item (e.g., Printer)

- (1) This printer has it all. Print, scan, copy, fax and wifi. Wifi makes this printer. No more cables all over the place and no more cluttered desks. Before, if I wanted to print something from my laptop I had to go to the printer and connect the cable. Now I can print over wifi. It prints very beautiful and also scans very high resolutions. Set up was a breeze. Getting other computers to print was also a breeze.
- (2) First of all, it does it all, and does it well. Print, scan, fax, and photos. Its six-ink system give archival photo prints with long life. This is my first wireless printer, and I have to say, it is a great system: easy to set up, and eliminates that spaghetti-ball of wires. Definitely a big plus. It's fast; very fast. Really cool-looking, and easy to use.

Generated Explanations

- **Word-Level:** First of all, it does it all, and does it well. Print, scan, fax, and photos. Its six-ink system give archival photo prints with long life. This is my first wireless printer, and I have to say, it is a great system: easy to set up, and eliminates that spaghetti-ball of wires. Definitely a big plus. It's fast; very fast. Really cool-looking, and easy to use.
- **Review-Level:** First of all, it does it all, and does it well. Print, scan, fax, and photos. Its six-ink system give archival photo prints with long life. This is my first wireless printer, and I have to say, it is a great system: easy to set up, and eliminates that spaghetti-ball of wires. Definitely a big plus. It's fast; very fast. Really cool-looking, and easy to use.
- **Extractive Summary-Level:** No more cables all over the place and no more cluttered desks. Before, if I wanted to print something from my laptop I had to go to the printer and connect the cable. Its six-ink system give archival photo prints with long life. This is my first wireless printer, and I have to say, it is a great system: easy to set up, and eliminates that spaghetti-ball of wires.
- **Abstractive Summary-Level:** I love this product. It is a great-looking printer and has an answering machine in one place. Setup was easy and I was happy to find this product, but it's a bit less expensive than a good purchase. It is a good value for the money.

source of information. By leveraging user-given reviews that discuss the rationale behind the ratings, recommender systems can uncover latent properties and dimensions of user opinions that are not captured by ratings alone [31]. Reviews offer a wealth of rich, multidimensional insights that cannot be otherwise acquired solely from ratings [3].

However, most neural review-based recommender systems lack explainability, which is crucial for user trust and decision-making [23, 25, 37]. The inherent opacity of neural networks, often referred to as *black-boxes*, creates a dilemma: a trade-off between

accuracy and explainability. [23, 25, 30]. The most accurate models often suffer from complexity and a lack of explainability [36]. Conversely, simple, explainable methods may compromise accuracy. Striking a balance between explainability and accuracy poses a significant challenge. Constructing models that are both explainable and accurate is a critical research agenda for the machine learning community to ensure that we derive benefits from machine learning fairly and responsibly [23].

To improve user trust and understanding, recent research has explored various explainability methods for recommender systems. Common approaches include review-level and word-level explanations. Review-level explanations, utilizing attention mechanisms to select high-scoring reviews, are considered state-of-the-art [3, 8]. Word-level explanations, which select top words based on attention weights, offer another strategy [26]. Nevertheless, both types may not fully resemble real-life explanations; as an illustration, in Table 1, the review-level explanation is identical to the second item review, assuming that it has the higher attention weight. It also inadvertently disregards other possibly useful sentences from reviews with lower attention scores. In essence, this degenerates into a review selection task. Moreover, while a word-level explanation highlights relevant terms, its fragmented nature may hinder real-world recommendation scenarios, as it may lack overall coherence and intelligibility for users.

Therefore, we propose and pioneer a novel CF framework called **SUMMER** (derived from **Summarization-Driven Collaborative Filtering for Explainable Recommendation**). Our model offers competitive recommendation performance and contributes to explainability research by exploring a less-investigated approach: treating it as an unsupervised summarization task within recommender systems. Unlike a review-level explanation, a summary-level explanation is expected to retain the most relevant texts across multiple reviews. Other advantages that make it preferred are its coherence, non-redundancy, and readability [4, 22]. In our implementation, SUMMER integrates a summarization layer into a CF architecture. This layer generates either extractive (selecting salient text segments) or abstractive (rephrasing with natural language generation) summaries for each item and user. The summarization layer acts as an encoding mechanism, with item/user embeddings pre-trained on the summarization task and fine-tuned for rating prediction. This novel approach unifies representation and explanation – the summary both *represents* and *explains* an item (or user). Importantly, our model performs unsupervised summarization, as expecting ground-truth summaries for large datasets is unrealistic and obtaining them manually is cumbersome. This lack of reliance on labeled data makes the approach especially appealing.

1.1 Contributions

These are the major contributions of our study:

- (1) We pioneer the integration of summarization and collaborative filtering for explainability. To the best of our knowledge, SUMMER is the first CF model capable of generating either extractive or abstractive explanations, offering a certain degree of flexibility.
- (2) To our knowledge, we are the first to emphasize that reformulating explainability as unsupervised summarization

is necessary to address the impracticality of ground-truth explanations.

- (3) Our experiments demonstrate SUMMER’s competitive rating prediction accuracy, aligning with or surpassing other state-of-the-art methods. In the context of explainability, our study suggests a potential preference for summary-level explanations, with extractive summaries being particularly well-received.
- (4) This study additionally explores the impact of explanation type (extractive or abstractive) on both recommendation performance and real-life acceptability.

2 REVIEW OF RELATED LITERATURE

Designing a collaborative filtering (CF) model involves two key steps: learning user/item representations and modeling user-item interactions based on those representations [11]. A fundamental work in this domain is neural collaborative filtering (NCF), which utilizes multilayer perceptron (MLP) layers to learn flexible interactions between users and items [12]. NCF overcomes the limitations of inner product-based interaction functions, enabling it to capture rich patterns in real-world data. DeepCoNN is the first model to jointly represent users and items using reviews, employing convolutional neural networks (CNN) in parallel networks connected by a shared layer [38]. NARRE, similar to DeepCoNN, incorporates review-level attention mechanisms to enhance embedding quality and provide review-level explanations [3].

Other notable studies include AHN, HUITA, MPCN, and NCEM, which employ various attention mechanisms to improve accuracy and explainability [7, 8, 28, 32]. These models integrate attention mechanisms differently to discern informative parts of data samples, leading to enhanced performance. HUITA incorporates a hierarchical, three-tier attention network. MPCN is similar to NARRE, but the former does not rely on convolutional layers. Instead, it introduces a review-by-review pointer-based mechanism that is co-attentive to model user-item relationships. AHN proposes a multi-hierarchical paradigm that recognizes user and item reviews through co-attention. NCEM replaces the CNN with a pre-trained BERT model in its parallel user/item networks. Incorporating BERT is found to be more advantageous since it can fully retain global context and word frequency information, crucial factors that can have consequences on rating prediction accuracy or recommendation performance [29]. In summary, there appears to be a trend; tackling explainability improves prediction and recommendation performance consequentially. While most recommender models address this via attention mechanisms, our proposed model solves this by unifying representation and explanation in the form of summaries.

On the principles of text summarization, two main approaches exist: extractive, which selects important sentences as they are, and abstractive, which rewrites sentences. Extractive methods are more commonly researched [22], while abstractive methods require advanced natural language generation techniques [34]. Summarization can also be categorized by the number of source documents: single-document summarization (SDS) and multi-document summarization (MDS), with MDS being more challenging due to integrating information from multiple sources [4]. Most summarization

models rely on supervised learning, necessitating labeled training data, which is often scarce and leads to poor generalization across domains [4, 5]. Miller proposed an unsupervised extractive method using BERT embeddings and K -Means clustering for sentence selection [20]. Chu and Liu introduced MeanSum, an unsupervised abstractive summarization model based on an autoencoder architecture [5].

The challenges inherent in summarization also apply to explainable recommender systems, where obtaining labeled datasets with ground-truth explanations is impractical. This makes unsupervised multi-document summarization particularly valuable. Within the proposed SUMMER framework, each document represents a user review, providing flexibility in adopting either extractive or abstractive methods.

3 METHODOLOGY

3.1 Problem Formulation and Overview

The training dataset τ consists of N tuples, with the latter indicating the size of the dataset. Each tuple follows this form: (u, i, r_{ui}, v_{ui}) where r_{ui} and v_{ui} respectively denotes the ground-truth rating and review given by user u to item i . Let $RV_u = \{v_{u1}, v_{u2}, \dots, v_{uj}\}$ be the set of all j reviews written by user u . Similarly, let $RV_i = \{v_{1i}, v_{2i}, \dots, v_{ki}\}$ be the set of all k reviews received by item i . Both RV_u and RV_i are acquired from scanning τ itself row-by-row. SUMMER's input is a user-item pair (u, i) from each tuple in τ . We specifically feed RV_u and RV_i to the model as the initial inputs. The primary output is the predicted rating $\hat{r}_{ui} \in \mathbb{R}$ that user u may give to item i . The rating prediction task can be expressed as:

$$\text{predict}(u, i) = (RV_u, RV_i) \rightarrow \hat{r}_{ui} \quad (1)$$

Its corresponding objective function, the mean squared error (MSE), is given below:

$$\text{MSE} = \frac{1}{|\tau|} \sum_{u, i \in \tau} (r_{ui} - \hat{r}_{ui})^2 \quad (2)$$

SUMMER's architecture is illustrated in Figure 1. It has two parallel modeling networks that respectively learn summarization-derived user and item representations. For the following subsection of this paper (i.e., 3.2 *Summarization Layer*), we will only discuss the item modeling procedure since it is nearly identical to user modeling, with their inputs as the only difference.

3.2 Summarization Layer

Through the summarization layer, our model's design is flexible enough to accommodate two possible options for explainability: extractive and abstractive, both of which can effectively represent, explain, and encode users and items. This layer produces the pre-trained summary-level explanation (for every user and every item), which we also call *representative summary*, *representation-explanation*, or *explanation-summary* in different parts of this paper. This section discusses our unsupervised implementations for either summarization approach.

3.2.1 EXTRACTIVE SUMMARIZATION LAYER. The reviews in RV_i are first concatenated together to form a single document. We employ spaCy's Sentencizer, a sentence segmentation tool for splitting the document into individual sentences [9]. The set of all

sentences in RV_i is now given by $E_i = \{e_{i1}, e_{i2}, \dots, e_{ig}\}$ where g refers to the total number of sentences. Afterward, E_i is fed to a pre-trained BERT model to obtain corresponding sentence embeddings. This process produces the set of sentence embeddings $E'_i = \{e'_{i1}, e'_{i2}, \dots, e'_{ig}\}$, where $E'_i \in \mathbb{R}^{g \times a}$ and a denotes BERT's embedding dimension. The BERT model choices can either be standard BERT-Large, wherein the contextualized embeddings can be derived from the penultimate encoder layer [20] or Sentence-BERT, which is based on RoBERTa-Large previously trained on the semantic textual similarity task [24]. Embedding clustering, based on K -Means, is then performed to partition the sentence embeddings in E'_i into K clusters. In our approach, K can be calculated using a hyperparameter called summary ratio (ϕ), which is the percentage of sentences that shall comprise the actual summary.

$$K = \phi \times g \quad (3)$$

The objective of embedding clustering is to minimize the sum of squared errors (SSE), i.e., the intra-cluster sum of the distances from each sentence to its nearest centroid, given by the following equation [33]:

$$\text{SSE}_i = \sum_{x=1}^K \sum_{e'_{iy} \in C_x} \|e'_{iy} - c_x\|^2 \quad (4)$$

where c_x is the centroid of cluster C_x that is closest to the sentence embedding e'_{iy} . The objective function is optimized for item i by running the assignment and update steps until the cluster centroids stabilize. The assignment step assigns each sentence to a cluster using the shortest distance between the sentence embedding and cluster centroid, provided by the formula below:

$$nc(e'_{iy}) = \text{argmin}_{x=1, \dots, K} \{\|e'_{iy} - c_x\|^2\} \quad (5)$$

where nc is a function that obtains the cluster closest to e'_{iy} . The update step recomputes the cluster centroids based on new assignments from the previous step. This is defined as:

$$c_x = \frac{1}{|C_x|} \sum_{y=1}^g \{e'_{iy} | nc(e'_{iy}) = x\} \quad (6)$$

where $|C_x|$ refers to the number of sentences that cluster C_x contains. By introducing clustering, redundant and related sentences are grouped in the same cluster. Sentences closest to each cluster centroid are selected and combined to form the extractive summary. This is expressed as:

$$\begin{aligned} ns(C_x) &= \text{argmin}_{y=1, \dots, g} \{\|e'_{iy} - c_x\|^2\} \\ XS_i &= [e'_{i, ns(C_1)}, e'_{i, ns(C_2)}, \dots, e'_{i, ns(C_K)}] \\ \overline{XS}_i &= \frac{1}{K} \sum_{x=1}^K e'_{i, ns(C_x)} \end{aligned} \quad (7)$$

where ns is a function that returns the nearest sentence to the centroid c_x of cluster C_x , $XS_i \in \mathbb{R}^{K \times a}$ is an embedding matrix of the extractive summary sentences, and $\overline{XS}_i \in \mathbb{R}^{1 \times a}$ is the extractive summary embedding of item i .

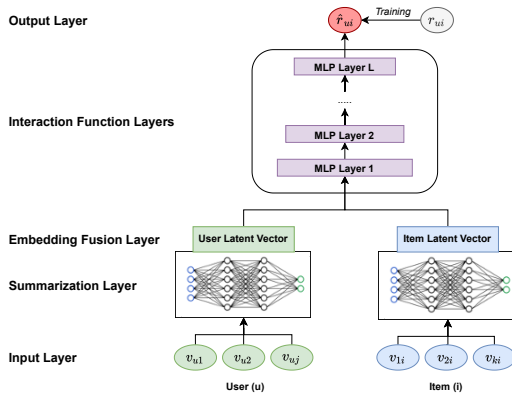


Figure 1: The proposed SUMMER framework.

3.2.2 ABSTRACTIVE SUMMARIZATION LAYER. Let \mathbb{D} be the set of all reviews in τ and $|\mathbb{D}|$ be the number of all tuples (i.e., user-item pairs) in τ . We initially have an invertible tokenizer T that maps the reviews in \mathbb{D} to token sequences $T(\mathbb{D})$ from a fixed vocabulary. Also, let $\mathbb{V} \subset T(\mathbb{D})$ denote the tokenized reviews that have a maximum length of H . For item i , given a set of reviews $RV_i \subset \mathbb{V}$, the goal is to produce an explanation-summary $XS_i \in T(\mathbb{D})$ using the same vocabulary.

The abstractive summarization layer contains two key components: the autoencoder and summarization modules. The autoencoder learns representations for each review in the training dataset τ and consequently constrains the generated summaries in its language domain. The encoder ϕ_E maps reviews to real-vector codes denoted by $z_y = \phi_E(v_{yi})$. After processing v one token per every time step, its encoding is expressed by concatenating the LSTM's final hidden and cell states, i.e., $\phi_E(v) = [h, c]$ [13]. Afterward, the decoder LSTM defines a distribution over \mathbb{V} contingent on the latent code $p(v|z_y) = \phi_D(z_y)$. This is accomplished by initializing the decoder's initial state with z_y and training it by teacher-forcing using a standard cross-entropy loss to reconstruct the original reviews. The autoencoder's objective is to minimize the reconstruction loss (*REC*), which is the collective cross-entropy losses (*CE*) between the original reviews and their corresponding reconstructed versions:

$$REC(RV_i, \phi_E, \phi_D) = \sum_{y=1}^k CE(v_{yi}, \phi_D(\phi_E(v_{yi}))) \quad (8)$$

On the other hand, the summarization module learns to produce explanation-summaries that are semantically similar to the input reviews. The latent codes of the reviews received by item i (i.e., $\{z_1, z_2, z_3, \dots, z_k\}$) are integrated by averaging their hidden and cell states in $\bar{z} = [\bar{h}, \bar{c}]$. The joint latent code \bar{z} is decoded by ϕ_D into summary s , which is then later encoded by $\phi_E(s) = [h_s, c_s]$. The encoded summary's hidden state also serves as the item's abstractive summary embedding: $\bar{XS}_i = h_s \in \mathbb{R}^{1 \times a}$, where a is the hidden unit size of the encoder.

The process of re-encoding and calculating the similarity loss between the generated summary and its source reviews further constrains the former to be semantically similar to the latter. Regarding this, the following is the objective function that minimizes

the similarity loss (*SIM*) based on the average cosine distance (*COS*) between the hidden states h_y of each encoded review and \bar{XS}_i of the encoded summary:

$$SIM(RV_i, \phi_E, \phi_D) = \frac{1}{k} \sum_{y=1}^k COS(h_y, \bar{XS}_i) \quad (9)$$

Similar to Chu and Liu's approach [5], the actual summary text is also generated using the Straight Through Gumbel-Softmax strategy. This performs approximated sampling from a categorical distribution, i.e., a softmax over the vocabulary, allowing gradients to be backpropagated through discrete generation.

3.3 Embedding Fusion Layer

We also draw certain principles from the traditional latent factor model by incorporating rating-based vectors that depict users and items to a certain extent [3]. These are represented by IV_u and IV_i , both in $\mathbb{R}^{1 \times m}$ where m is the dimension of the latent vectors. The hidden vectors are fused with their corresponding summary embeddings. This is facilitated by these fusion levels, illustrated by the following formulas:

$$\begin{aligned} f_u &= (\bar{XS}_u W_u + b_u) + IV_u \\ f_i &= (\bar{XS}_i W_i + b_i) + IV_i \\ f_{ui} &= [f_u, f_i] \end{aligned} \quad (10)$$

where f_u and f_i pertain to the preliminary fusion layers and both are in $\mathbb{R}^{1 \times m}$; W_u and W_i are weight matrices in $\mathbb{R}^{a \times m}$; b_u and b_i refer to bias vectors; and $f_{ui} \in \mathbb{R}^{1 \times 2m}$ denotes the initial user-item interactions from the third fusion layer.

3.4 Interaction Function and Rating Prediction

The MLP is essential to model the CF effect to learn meaningful interactions between users and items. An MLP with multiple layers implies a higher degree of non-linearity and flexibility. Similar to the strategy of He et al. [12], SUMMER adopts an MLP with a tower pattern wherein the bottom layer is the widest while every succeeding top layer has fewer neurons. A tower structure enables the MLP to learn more abstractive data features. Notably, we halve the size of hidden units for each successive higher layer. SUMMER's MLP component is defined as follows:

$$\begin{aligned} h_1 &= ReLU(f_{ui} W_1 + b_1) \\ h_L &= ReLU(h_{L-1} W_L + b_L) \end{aligned} \quad (11)$$

where h_L represents the L -th MLP layer, and W_L and b_L pertain to the L -th layer's weight matrix and bias vector, respectively. We choose the rectified linear unit (ReLU) as the activation function since it generally yields better performance than other activation functions [12]. Finally, the MLP's output is projected to one more linear layer to produce the predicted rating:

$$\hat{r}_{ui} = h_L W_{L+1} + b_{L+1} \quad (12)$$

Table 2: The datasets utilized for our experiments.

Dataset	#Reviews	#Users	#Items
Digital Music	64,706	5,541	3,568
Office Products	53,258	4,905	2,420
Patio, Lawn, & Garden	13,272	1,686	962

Table 3: Variants of SUMMER used in our ablation study.

Variant	Type	Item Encoder	User Encoder
SUMMER-1SE	Null	First Sentence	First Sentence
SUMMER-1A-U1	Hybrid	Abstractive Summ.	First Sentence
SUMMER-1X-U1	Hybrid	Extractive Summ.	First Sentence
SUMMER-11-UA	Hybrid	First Sentence	Abstractive Summ.
SUMMER-11-UX	Hybrid	First Sentence	Extractive Summ.
SUMMER-5RE	Null	Five Reviews	Five Reviews
SUMMER-1A-U5	Hybrid	Abstractive Summ.	Five Reviews
SUMMER-1X-U5	Hybrid	Extractive Summ.	Five Reviews
SUMMER-15-UA	Hybrid	Five Reviews	Abstractive Summ.
SUMMER-15-UX	Hybrid	Five Reviews	Extractive Summ.
SUMMER-Ext	Original	Extractive Summ.	Extractive Summ.
SUMMER-Abs	Original	Abstractive Summ.	Abstractive Summ.

4 EMPIRICAL EVALUATION

4.1 Research Questions

In this section, we provide the details of our experimental configuration as we aim to answer the following research questions (RQs):

- **RQ1:** How does SUMMER’s rating prediction accuracy compare to other state-of-the-art baselines?
- **RQ2:** To what extent does summarization (as an encoding mechanism) improve the effectiveness of user and item representations?
- **RQ3:** How are summary-based explanations perceived by humans in real life?
- **RQ4:** How do extractive and abstractive representation-explanations compare in the following:
 - **RQ4a:** Their impact on recommendation performance?
 - **RQ4b:** Their real-life acceptability to users?

4.2 Datasets, Baselines, and Evaluation Metric

Table 2 summarizes the three Amazon datasets¹ we utilized in our experiments. These datasets are 5-core, implying that every user and every item have a minimum of five reviews [10, 19]. The ratings across all the datasets are in the range of one to five. We further divided a given dataset into training, validation, and test sets using the 80%-10%-10% split. Then, to compare recommendation performances and validate our model’s effectiveness, the following state-of-the-art baselines were used:

- **DeepCoNN** [38]: The first deep collaborative neural network model that is based on two parallel CNNs to jointly learn user and item features.

- **MPCN** [28]: Akin to NARRE, MPCN implements a new type of dual attention for identifying relevant reviews.
- **NARRE** [3]: Similar to DeepCoNN, it is a neural attentional regression model that integrates two parallel CNNs and the review-level attention mechanism.
- **NCF** [12]: An interaction-based model that is fundamental in neural recommender systems; the first to introduce the MLP as the interaction function.

For the evaluation metric, we calculated each baseline’s root mean square error (RMSE) on the test dataset (\bar{r}). RMSE is a widely accepted metric for assessing a model’s accuracy and recommendation performance [27].

$$RMSE = \sqrt{\frac{1}{|\bar{r}|} \sum_{u,i \in \bar{r}} (r_{ui} - \hat{r}_{ui})^2} \quad (13)$$

4.3 Experimental Settings

For the CF component of SUMMER, we operated an exhausting grid search on the number of epochs: [1, 30] and latent vector dimension (m): {128, 200, 220} while fixing the values of the learning rate at 0.006 and number of MLP layers at 4. Moreover, we implemented NCF and also ran a grid search over the number of epochs: [1, 30] and latent vector dimension: {128, 200}. For DeepCoNN, MPCN, and NARRE, we availed the extensible NRRec framework² and retained the hyperparameters’ values reported in the framework [18]. We performed an exhaustive grid search over the number of epochs: [1, 30] and learning rates: {0.003, 0.004, 0.006}.

All the above-mentioned baselines used the same optimizer, Adam, which leverages the power of adaptive learning rates during training [15]. This makes the selection of learning rates less cumbersome, leading to faster convergence [3]. Without special mention, the models shared the same random seed, batch size (128), and dropout rate (0.5). We selected the model configuration with the lowest RMSE on the validation set. We then separately trained SUMMER’s summarization and rating prediction tasks due to limitations on hardware resources.

For the extractive summarization layer, we primarily based its implementation on BERT Extractive Summarizer³ by Miller [20] using the pre-trained BERT_{LARGE} model. The summary ratio (ϕ) was set to 0.4. For the abstractive summarization layer, we patterned its design after the original MeanSum⁴ model of Chu and Liu [5]. The language model, encoders, and decoders were multiplicative LSTMs [17] with hidden unit size of 512, dropout rate of 0.1, word embedding size of 256, and layer normalization [1]. We also took advantage of the Adam optimizer to train the language and summarization models with learning rates of 0.001 and 0.0005, respectively.

²<https://github.com/ShomyLiu/Neu-Review-Rec>

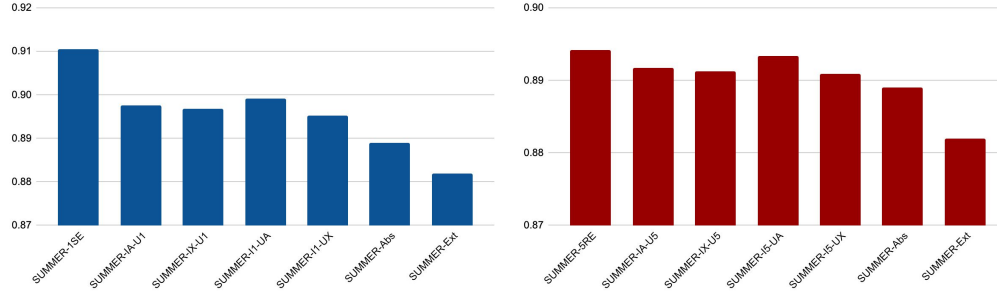
³<https://github.com/dmmiller612/bert-extractive-summarizer>

⁴<https://github.com/sosuperic/MeanSum>

¹<http://jmcauley.ucsd.edu/data/amazon/>

Table 4: Recommendation performance comparison. The best RMSE values are boldfaced.

Model	Digital Music	Office Products	Patio, Lawn, & Garden	Average RMSE
DeepCoNN	0.8904	0.8410	0.9316	0.8876
MPCN	0.9298	0.8487	0.9362	0.9049
NARRE	0.8915	0.8426	0.9539	0.8960
NCF	1.0822	1.0008	1.2359	1.1063
SUMMER-Ext	0.8831	0.8332	0.9298	0.8820
SUMMER-Abs	0.8917	0.8356	0.9398	0.8890

**Figure 2: Performance comparison of SUMMER variants. The left (a) and right (b) figures illustrate the RMSE scores of SUMMER variants based on 1SE and 5RE ablations, respectively.**

5 PREDICTION RESULTS AND DISCUSSION

5.1 Performance Comparison

After conducting our experiments, we found out that SUMMER’s extractive version is the top-performing model, acquiring the lowest RMSE scores across all datasets and baselines. This is closely followed by SUMMER’s abstractive variant, whose performance is comparable to other baselines. These observations answer RQ1; our findings prove that integrating summarization in a CF architecture is effective, subsequently resulting in better representations and accurate recommendations. Specifically, summarization is helpful in further refining and producing semantically meaningful features with finer granularity and fewer redundancies to comprise user and item embeddings. Notwithstanding, extractive SUMMER appears to have a better generalization capability than abstractive SUMMER; this addresses RQ4a.

An interesting trend is that models that take advantage of review information (i.e., DeepCoNN, MPCN, NARRE, and SUMMER) consistently outperform NCF, the only model based on rating data alone. This validates the importance of review texts, which are excellent rich information sources for learning user and item properties. Generally, review-based recommender systems have become more reliable nowadays in yielding satisfactory and quality prediction performance.

5.2 Ablation Study

5.2.1 Configuration. In order to examine further the efficacy of our proposed summarization layer for encoding users and items, we separately replaced the user’s and item’s summarization layer with non-summarization encoding approaches (listed below). The rationale behind these approaches is that to examine better the

perceived overall effect of summarization in CF, we must ensure that the ablated component is replaced by an encoding that does not resemble SUMMER-generated summaries.

- **First Sentence Encoding (1SE):** We chose the first sentence of the item (user) review set to represent it. We then projected it to a pre-trained Sentence-BERT model to derive the item (user) embedding.
- **Five Reviews Encoding (5RE):** We randomly selected five reviews from the item (user) review set. We later fed the concatenated reviews to Sentence-BERT to obtain the embedding of the item (user).

Accordingly, as described in Table 3, we prepared ten other variants of SUMMER that utilize different combinations of encoding mechanisms for the user and item components. These are different from the *original variants* (i.e., SUMMER-Ext and SUMMER-Abs); they are non-original and can be categorized into the *null variant* or *hybrid variant*. A null variant, such as SUMMER-1SE and SUMMER-5RE, completely replaces the summarization layer with its corresponding encoding mechanism (i.e., 1SE or 5RE). On the other hand, for a hybrid variant, either the user or item component maintains the summarization layer while the other ablated component employs 1SE or 5RE.

5.2.2 Analysis. The results of our ablation experiments are depicted in Figure 2. Completely removing summarization expectedly results in the least accurate performances, as evidenced by SUMMER-1SE and SUMMER-5RE receiving the lowest RMSE scores. The performance immediately improves even if only either component (item or user) takes advantage of summarization. Hence, it is noticeable that the RMSE values of hybrid variants are significantly

Table 5: Acceptability comparison between extractive and abstractive summary-level explanations. The best mean values are boldfaced. The symbol ** indicates that the difference is statistically significant (at $p < 0.05$).

Quality	Extractive	Abstractive
Coherence	3.33	3.27
Focus	3.31	3.27
Grammaticality	3.25	3.20
Non-Redundancy	3.15	3.18
Referential Clarity	3.59	3.16
Usefulness	3.50**	3.20

better than the null variants. Lastly, the full benefits of summarization are realized when both item and user components utilize our proposed summarization layer. Across all variants, SUMMER-Ext and SUMMER-Abs have the two best RMSE scores; this conclusively answers RQ2.

6 EXPLAINABILITY STUDY

6.1 Listwise Evaluation of Helpfulness

It is intuitive to perform a listwise evaluation (i.e., a ranking-based analysis) to examine whether explanations generated by SUMMER are on par with review-level explanations in real life, considering that the latter have been popular in recommender systems literature. In this regard, we generated a tuple of explanation texts (SUMMER’s extractive and abstractive summaries and NARRE’s review-level texts) for 30 items, totaling 90 explanations to be assessed. For each item explanation tuple, we instructed four English-speaking human judges to rank them according to helpfulness. A statement is considered helpful if it can aid the viewing user both to know a product better and make a future purchasing (or non-purchasing) decision. Additionally, we determined the strength of inter-judge ranking agreement by utilizing Fleiss’ Kappa (κ) wherein 0 refers to a random agreement, and 1 denotes a perfect agreement [2, 14].

Figure 3 shows a clear tendency toward extractive summaries, ranked first for nearly 60% of items. Review-level explanations follow (31%), while abstractive summaries are favored least (ranked first 10% and ranked last 63%). With a fair inter-judge agreement (Kappa value of 0.25), this highlights the superiority of extractive summaries in real-life acceptability (RQ3), exceeding review-level explanations. The lower ranking of abstractive summaries suggests potential inaccuracies or irrelevancies introduced by the generative model (partly answering RQ4b).

6.2 Acceptability Comparison

To fully address RQ4b, it is reasonable to conduct a detailed comparison between extractive and abstractive explanations based on their inherent qualities. In the absence of a ground-truth summary, we propose assessing the real-life acceptability of explanations based on the following summarization criteria, measured on a scale of 1 (poor) to 5 (excellent). We adopted the first five qualities from Dang [6], while we added the sixth dimension of usefulness.

- (1) **Coherence**: Should be well-structured and well-organized. It should build from sentence to sentence to a coherent body of information about an item/product/topic.
- (2) **Focus**: Should only contain information that is related to the rest of the summary.
- (3) **Grammaticality**: No obvious ungrammatical sentences (e.g., fragments, missing components) that make the text difficult to read.
- (4) **Non-Redundancy**: No unnecessary repetition in the summary.
- (5) **Referential Clarity**: Easy to identify who or what the noun phrases and pronouns are referring to in the summary.
- (6) **Usefulness**: Provides useful information to help users decide in making a purchasing decision.

Considering this, we produced a total of 60 item explanations, 30 each from SUMMER’s extractive and abstractive types. We likewise asked four English-speaking judges to independently evaluate the explanations according to the earlier-mentioned summarization criteria, and we ascertained their agreement level by utilizing Fleiss’ Kappa. Furthermore, we employed the t -test to determine whether the difference between the two explanation types is statistically significant for each summarization quality.

Table 5 reveals that extractive explanations outscore abstractive explanations in five out of six quality metrics. Specifically, they are most grammatical, coherent, and useful; they also provide the best focus and clarity. Their strongest aspects are referential clarity and usefulness. The former has a mean score of 3.59 and a κ value of nearly 0.1, indicating a slight agreement between a pair of judges. The latter has a mean score of 3.50 and a p -value of about 0.03, implying that the difference is statistically significant from abstractive explanations’ usefulness. Interestingly, abstractive summaries are less redundant than their extractive counterparts (with a mean score of 3.18).

Overall, these findings demonstrate a strong preference for extractive explanations over abstractive ones in real-life settings, addressing RQ4b. Extractive explanations provide clear, well-structured guidance to help users understand items and make informed purchasing decisions.

7 CONCLUSION AND FUTURE WORK

We have successfully implemented SUMMER, a novel summarization-driven CF model that is both accurate and explainable. By integrating summarization as an encoding mechanism within a CF architecture, we generate semantically rich user and item representations, resulting in competitive recommendation performance. Our experiments confirm that SUMMER’s accuracy aligns with or surpasses other state-of-the-art approaches. Moreover, our explainability study demonstrates the real-life favorability of extractive summary-level explanations.

Nevertheless, our study highlights several avenues for further research. Expanding human evaluation with a larger pool of judges and thoroughly analyzing factors contributing to varying inter-rater agreement offers one valuable direction. Additionally, a deeper investigation of the accuracy-explainability trade-off is crucial. Analyzing specific cases where less helpful explanations might also impact prediction accuracy could reveal areas for improving both

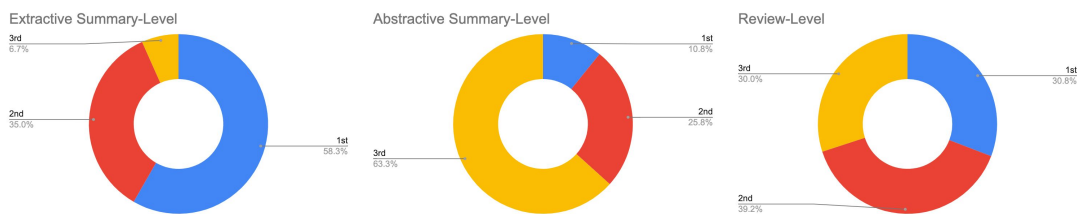


Figure 3: Distribution of the judges' given helpfulness rankings for listwise evaluation.

the informativeness of explanations and the overall model performance.

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).

[2] Ria Mae Borromeo and Motomichi Toyama. 2015. Automatic vs. crowdsourced sentiment analysis. In *Proceedings of the 19th International Database Engineering & Applications Symposium*. 90–95.

[3] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference*. 1583–1592.

[4] Radia Rayan Chowdhury, Mir Tafseer Nayeem, Tahsin Tasnim Mim, Md Sairur Rahman Chowdhury, and Taufiqul Jannat. 2021. Unsupervised Abstractive Summarization of Bengali Text Documents. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 2612–2619.

[5] Eric Chu and Peter Liu. 2019. Meansum: a neural model for unsupervised multi-document abstractive summarization. In *International Conference on Machine Learning*. PMLR, 1223–1232.

[6] Hoa Trang Dang. 2006. DUC 2005: Evaluation of question-focused summarization systems. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*. 48–55.

[7] Xin Dong, Jingchao Ni, Wei Cheng, Zhengzhang Chen, Bo Zong, Dongjin Song, Yanchi Liu, Haifeng Chen, and Gerard de Melo. 2020. Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 7667–7674.

[8] Xingjie Feng and Yunze Zeng. 2019. Neural Collaborative Embedding From Reviews for Recommendation. *IEEE Access* 7 (2019), 103263–103274.

[9] Sarang Gupta and Kumari Nishu. 2020. Mapping Local News Coverage: Precise location extraction in textual news content using fine-tuned BERT based language model. In *Proceedings of the 4th Workshop on Natural Language Processing and Computational Social Science*. 155–162.

[10] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*. 507–517.

[11] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer product-based neural collaborative filtering. *arXiv preprint arXiv:1808.03912* (2018).

[12] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Selim Kiliç. 2015. Kappa test. *Psychiatry and Behavioral Sciences* 5, 3 (2015), 142.

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[17] Ben Krause, Liang Lu, Iain Murray, and Steve Renals. 2016. Multiplicative LSTM for sequence modelling. *arXiv preprint arXiv:1609.07959* (2016).

[18] Hongtao Liu, Fangzhao Wu, Wenjun Wang, Xianchen Wang, Pengfei Jiao, Chuhan Wu, and Xing Xie. 2019. NRPA: Neural Recommendation with Personalized Attention. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1233–1236.

[19] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 43–52.

[20] Derek Miller. 2019. Leveraging BERT for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165* (2019).

[21] Cataldo Musto, Marco de Gemmis, Giovanni Semeraro, and Pasquale Lops. 2017. A Multi-criteria Recommender System Exploiting Aspect-based Sentiment Analysis of Users' Reviews. In *Proceedings of the 11th ACM Conference on Recommender Systems*. 321–325.

[22] Mir Tafseer Nayeem, Tanvir Ahmed Fuad, and Yllias Chali. 2018. Abstractive unsupervised multi-document summarization using paraphrastic sentence fusion. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1191–1204.

[23] Georgina Peake and Jun Wang. 2018. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2060–2069.

[24] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[25] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1135–1144.

[26] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the 11th ACM Conference on Recommender Systems*. 297–305.

[27] Harald Steck. 2013. Evaluation of recommendations: rating-prediction and ranking. In *Proceedings of the 7th ACM Conference on Recommender systems*. 213–220.

[28] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018. Multi-pointer co-attention networks for recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2309–2318.

[29] Qianqian Wang, Si Li, and Guang Chen. 2018. Word-driven and context-aware review modeling for recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1859–1862.

[30] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. 2018. TEM: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference*. 1543–1552.

[31] Jobin Wilson, Santanu Chaudhury, and Brijesh Lall. 2014. Improving collaborative filtering based recommenders using topic modelling. In *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 01*. 340–346.

[32] Chuhan Wu, Fangzhao Wu, Junxin Liu, and Yongfeng Huang. 2019. Hierarchical user and item representation with three-tier attention for recommendation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 1818–1826.

[33] Shuyin Xia, Daowan Peng, Deyu Meng, Changqing Zhang, Guoyin Wang, Elisabeth Giem, Wei Wei, and Zizhong Chen. 2020. A fast adaptive K-means with no bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[34] Haoyu Zhang, Jianjun Xu, and Ji Wang. 2019. Pretraining-based natural language generation for text summarization. *arXiv preprint arXiv:1902.09243* (2019).

[35] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 5.

[36] Yongfeng Zhang and Xu Chen. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192* (2018).

[37] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 83–92.

[38] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*. 425–434.

ErrgoEngine: A Contextualized Programming Error Analysis Translation Engine

Ned Isaiah Palacios
University of the Immaculate
Conception
Davao City, Philippines
npalacios_20000000446@uic.edu.ph

James Bryan Parcasio
University of the Immaculate
Conception
Davao City, Philippines
jparcasio_20000000245@uic.edu.ph

Joseph Anthony Pornillos
University of the Immaculate
Conception
Davao City, Philippines
jpornillos_20000001046@uic.edu.ph

Rogelio Badiang Jr.
University of the Immaculate
Conception
Davao City, Philippines
rbadiang@uic.edu.ph

Aurora Cristina Manseras
University of the Immaculate
Conception
Davao City, Philippines
amanseras@uic.edu.ph

ABSTRACT

A novel approach called ErrgoEngine is presented to enhance programming error messages for novice programmers. Current solutions have limitations in providing personalized, interactive, and domain-specific assistance to address the challenges novice programmers face in understanding error messages. ErrgoEngine aims to provide a rich explanation to novice programmers, alleviating the burden of getting the right answer to their programming error with minimal intervention. Designed to be language-agnostic and leverages innovations in program analysis and language tooling, as well as relevant programming learning theories, ErrgoEngine uses regular expressions to match the input error message to a corresponding error template. It extracts relevant data and generates an error explanation and bug fix suggestions based on the template and context. Evaluation showed ErrgoEngine successfully identified and translated 42 programming errors with high test coverage. Although weaknesses were identified in semantic analysis and output generation, ErrgoEngine has the potential to simplify learning for novice programmers of all skill levels. Ongoing work aims to address these weaknesses and improve the tool's effectiveness.

CCS CONCEPTS

• **Applied computing** → **E-learning**; • **Software and its engineering** → *Software testing and debugging*;

KEYWORDS

contextualized programming error analysis translation, debugging, programming error, program repair

ACM Reference Format:

Ned Isaiah Palacios, James Bryan Parcasio, Joseph Anthony Pornillos, Rogelio Badiang Jr., and Aurora Cristina Manseras. 2024. ErrgoEngine: A Contextualized Programming Error Analysis Translation Engine. In *Proceedings*

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

PCSC2024, May 2024, Laguna, Philippines

© 2024 Copyright held by the owner/author(s).

of Philippine Computing Science Congress (PCSC2024). Laguna, Philippines, 8 pages.

1 INTRODUCTION

Software automation is becoming a vital part of almost every major industry nowadays and programming is an essential skill for students pursuing computing-related courses and is becoming a useful skill for non-tech fields[26]. For those who take programming courses, known as novice programmers[17], the skill of programming enables them to solve problems and enhances their analytical and reasoning capabilities. However, programming is a "high-level" cognitive task[51] and learning proved to be a challenge for many of the novices. One area of difficulty for novices when it comes to programming is in addressing programming errors[24, 56, 65] which requires the development of debugging skills.

Debugging is the process of finding and resolving errors in software[35]. In this process, the programmer replicates the failure for two main purposes: fault localization and bug fixing[38]. Fault localization is about identifying the cause of the failure, which can be a tedious and time-consuming task in manual processes[62, 64]. However, it is essential for maintaining software quality because the faster the bug location is identified, the faster it can be removed[2]. Techniques for fault localization include program logging, assertions, profiling[64], and stack-trace analysis[63].

Bug fixing, on the other hand, is the process of preventing failures by modifying, adding, or deleting code[38]. There are three approaches: search-based, semantic-based, and template-based[10]. Search-based bug fixing uses search algorithms to automatically replace defective code[58] with a solution that meets specific criteria, such as passing unit tests or satisfying formal specifications[16]. Semantic-based bug fixing analyzes the program's behavior to generate patches and constraints that avoid the bug[31], using a constraint solver[67]. Template-based bug fixing uses fixing templates based on corresponding bug identification, which are extracted from historical data such as bug reports and answers from question-and-answer websites[33].

Debugging a program, including fault localization and bug fixing, is similar to the processes outlined in Brown and VanLehn's repair theory. It is based on the concept that individuals acquire

procedural skills by attempting to solve problems[23]. Novice programmers apply the knowledge they have gained from their classes to resolve issues. If they are successful, they acquire a new approach to address the error. If not, they encounter an impasse. To overcome the impasse, individuals may either try again later or choose to repair their knowledge through a problem-solving process[34]. As they work to identify the source of the impasse, in this case, programming errors or bugs, they seek to develop a new solution that may lead to success or present an entirely new impasse[8].

Like programming, debugging is a difficult skill to master[41] as it requires programmers to constantly use new debugging strategies as they learn[14]. Proficiency in interpreting result codes, error, and warning messages of the compiler is essential[48]. A study found that the majority of students have only fair to satisfactory performance in program tracing, a method of fault localization[25]. This may be due to fragile knowledge[53], which is described as knowledge that students may only partially know, have difficulty harnessing, or be unable to recall[41]. Text-based programming error messages can be helpful, but they often use technical vocabulary[37] and can be notoriously cryptic, leading to confusion and discouragement[5]. Error messages may also be inaccurate, imprecise, and vary between different versions of the compiler used[40].

The students of a computer studies department at a local institution in Davao City also encountered issues with debugging. Researchers conducted an in-person evaluation survey to assess the students' current level of programming experience by analyzing and solving programming errors. Three programming errors were selected and presented to the students based on their current programming proficiency. The respondents consisted of freshman students from the CCS department with 59 of 70 (84.3%) of them identifying themselves as beginners or novices. Of the 59 respondents, 16 of them (27.12%) were able to correctly identify and produce a correct solution for three programming errors, 12 respondents (20.34%) correctly identified and solved two errors, 12 respondents (20.34%) also correctly identified and solved one error, 1 respondent (1.69%) was unable to both identify and produce a solution for the errors while the remaining 18 respondents (30.5%) have varying numbers of identified and solved programming errors. Notably, 53 of the 59 respondents (89.83%) who correctly identified and solved the first programming error were less confident in identifying and solving the remaining errors. Some students sought help from their peers, while others relied on their integrated development environments (IDEs) to do fault localization through program tracing. The results of this evaluation survey highlight the need for enhanced or improved programming error messages that are more informative and easier to understand, especially for beginner and novice programmers.

Several studies have been conducted to improve the understanding of programming error messages. One study found that enhanced forms of Java error messages reduced the overall number of errors, repeated errors, and created a positive learning experience for students[4]. However, other studies have shown that enhanced programming error messages did not significantly impact student learning[13, 54, 68]. This is due to various factors, such as difficulties in typing code[68], improper reporting[54], and not paying attention[13, 54]. Another concern is the lack of easy to use and

understand debugging tools that could help novice programmers debug their code more quickly and easily. Traditional debuggers, such as those found in IDEs, have a steep learning curve[20] because they assume that the user already has advanced knowledge and experience with debugging tools[21, 36]. This is also evident in the use of print statements over IDE debuggers to study the behavior of the program[6].

To address the issues around programming errors and debugging tools, several studies[15, 19, 21, 32, 49, 57] have employed various methods and techniques to improve the learning curve of novice programmers when it comes to fixing errors. An automated tool called Gauntlet was developed to catch and explain the top fifty programming errors, both syntax and semantic-based, in a pre-compiler for student cadets at the United States Military Academy[15]. Backstop is another tool for enhancing runtime error messages and debugging logical errors in Java[49]. This tool was tested on a set of students with little to no programming experience, and the results showed it was proven effective despite some students initially finding the output hard to understand. Another tool called SeeC is developed as a novice-focused debugging tool prototype for C programmers that provides runtime error checking[21]. Their tool uses a localized format string used to generate the explanations of the error, and a C compiler named Clang for creating mappings into the program's respective source code files for rich explanation. A study by Charles and Gwilliam has developed and implemented an error explainer tool for use by non-CS students to assist them in understanding and interpreting Python error messages[19]. The tool, built as a plugin into an interactive notebook environment, collects and provides plain English explanations for various error types, helping students to identify common causes and gain a better understanding of the error messages. While the paper acknowledged the lack of quantitative improvement, the students found the tool useful.

Recently, researchers have investigated the use of large language models (LLMs) to enhance the programming error messages. One study used OpenAI's Codex programming model to explain and to provide fix suggestions for given programming errors[32]. The model was able to provide comprehensible messages 88% of the time, with an average rate of correctness at 47%, with improved rates if "prompt engineering" is employed. Another study conducted by Sobania investigated the performance of ChatGPT in automatic bug fixing[57] using a benchmark which consists of small erroneous Python programs that fit in the dialog system of ChatGPT. The results showed that ChatGPT successfully solved 19 out of 40 bugs, compared to 7 bugs fixed by standard program repair methods. However, ChatGPT required additional information for most of the bugs in order to solve them, which increased its success rate by 77.5%. Despite the positive results, the researchers noted that the mental cost of verifying the bug fixes provided by ChatGPT may outweigh its advantages.

The solutions explored in these studies indicate the possibilities of analyzing and translating programming error messages, relying on "hardcoded" predefined templates or rules that use traditional string matching techniques[19, 21]. However, these methods hinder extensibility when new types of errors arise or when users want to add custom support for an error. Additionally, the tools only support a limited range of programming error types, notably syntax and

semantic errors. While large language models show promise, they do not provide accurate explanations due to the lack of contextual data[32, 57] and face limitations in understanding source codes due to missing information such as the source code of related files and the libraries used. Prompt engineering can be employed to overcome these limitations[59], but it may require multiple attempts and tricks to overcome token limitations with no guaranteed consistent output[61]. Therefore, these models fall short in offering the personalized, interactive, and domain-specific assistance necessary for novice programmers without further intervention.

Newer programming language tooling technologies and analysis strategies, such as Tree-Sitter[9] and a program-feedback graph from a self-supervised, program repair neural network[66], have emerged recently and have the potential to provide a common approach to programming analysis and translation, making it easier to deploy or integrate into common software. This would greatly benefit novice programmers as these tools are more accessible and user-friendly compared to the tools and programs built from previous studies[39].

Furthermore, current programming error feedback mechanisms lack a strong foundation in programming learning theories for novice programmers[5]. To address these challenges, incorporating experiential learning theory by Kolb can be beneficial. This theory emphasizes "learning by doing" or a learning process in which knowledge is gained through observation, abstract conceptualization, and active experimentation. By providing novice programmers with concrete learning through real-world programming error experiences, they can engage in reflective observation and gain insights into effective debugging strategies. Through abstract conceptualization, they can derive general principles and mental models for effective bug localization and fixing. Finally, active experimentation enables programmers to apply their newly formed concepts and theories in practice, refining their debugging skills[30]. This approach can enhance the understanding of programming error messages and facilitate the learning process for novice programmers.

Given the identified gaps and opportunities, the researchers aimed to develop a contextualized programming error analysis translation engine, named ErrgoEngine, to assist new programmers in effectively overcoming common through experience[12]. It involves a four-stage process: concrete learning, ref debugging challenges. The engine will use a language-agnostic approach to enhance error programming messages by leveraging innovations in program analysis and language tooling as well as applying relevant programming learning theories. This will provide a rich and helpful explanation to novice programmers, alleviating the burden of getting the right answer to their programming error with minimal to no intervention. Ultimately, the researchers believe that ErrgoEngine will simplify the learning process for novice programmers and programmers of all skill levels, helping them transform into skilled troubleshooters.

2 APPROACH

2.1 Conceptual Framework

Figure 1 illustrates the process of error analysis and translation. Upon receiving the input programming error message text from

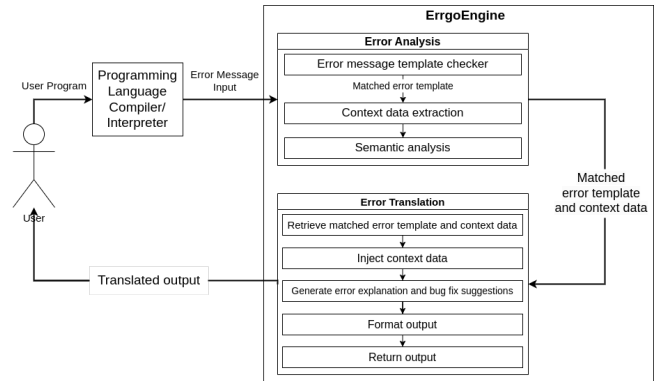


Figure 1: Conceptual Framework of ErrgoEngine

the compiler / interpreter, it goes through the error analysis and error translation stages to produce its desired output.

2.1.1 Error Analysis. In this stage, the input error message is examined and relevant data is extracted into context data. ErrgoEngine searches for a corresponding error template in its collection using a regular expression pattern included in each template. Upon discovering a match, the engine proceeds to context data extraction. If no match is found, a fallback error template is used instead. During context data extraction, variables from the input error message are obtained using the same regular expression pattern utilized in the error template matching process. If stack trace data is available, it is separately extracted using a specific regular expression pattern defined in the programming language configuration linked to the matching error template. This process extracts the locations of the offending files, whose contents are then semantically analyzed to generate a symbol table containing the program's symbols (variables, functions, classes, etc.) using the same programming language configuration used earlier. The context data, along with the matched error template, is subsequently forwarded to the error translation stage.

2.1.2 Error Translation. In the final stages, the error explanation and the bug fix for the corresponding error are generated using the error template and the context data. The error explanation is produced by executing an error explanation function, which accepts context data and returns a text output. This function can use the context data to inject values into the error explanation or selectively generate text based on the symbol information or the AST node structure of the offending code. Once the error explanation is generated, the bug fix can be produced. A function then accepts the context data and returns a list of bug fix suggestions based on the identified error through the error template upon execution. The newly-generated error explanation and bug fix suggestions are then compiled and formatted into Markdown, a plain text format for writing structured documents[1]. The formatted output is sent back to the user for consumption.

2.2 Context Data

The context data contains crucial information primarily used for the error translation stage and portions of the error analysis stage.

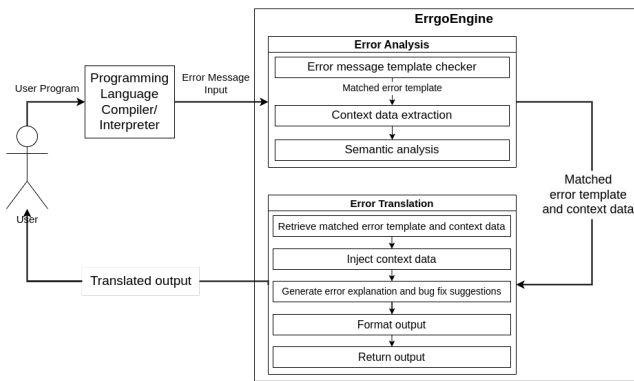


Figure 2: Flow diagram of the BugBuddy application

It provides guidance and personalization for the generated output. The data includes information extracted from the input error message, such as variables and stack trace locations. Additionally, it contains data extracted from semantic analysis, including files, abstract syntax trees (ASTs), and the symbol table containing the extracted symbol information.

2.3 Language-agnostic Approach

ErrgoEngine is able to identify errors and source codes written in any programming language through a supported programming language parser and an additional configuration file for that specific language. The supported programming parsers are the ones that were generated from parser generators such as ANTLR[52] and Yacc[27] as the generated parsers use common data structures that are easy to traverse through the provided mechanisms [3]. In this case, ErrgoEngine used Tree-Sitter parser as the library for parsing and programming languages with Tree-Sitter support are automatically supported.

In addition, a separate language configuration file is also created which are linked into the error templates of their respective programming errors. It contains a set of operations and variables needed in order to produce additional information helpful in identifying errors such as functions for converting stack trace positions, tracking dependency imports, and capturing symbol information as part of the semantic analysis.

2.4 Library / Application Development

ErrgoEngine is a software component library that can be used on a variety of similarly-scoped applications. To test its functionality in the real-world scenario, the researchers developed a purpose-built application called BugBuddy. Shown in Figure 2 and 3, BugBuddy is a console-based server application and editor extension that enhances programming error messages within the user’s preferred text editor. It consists of multiple components:

- (1) **Program Executor** - monitors incoming programming error messages via the executed program’s standard error output pipe (STDERR). Error messages are sent to the daemon server for processing.

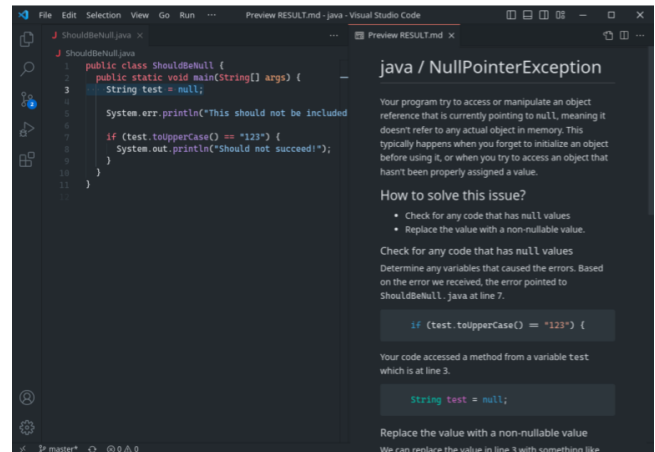


Figure 3: BugBuddy editor example user interface

- (2) **Daemon Server** - an independent, background process that acts as a main hub for receiving, processing, and dispatching output generated by ErrgoEngine. Communication between clients (e.g. program executors and language servers) is done via the JSON Remote Procedure Call (JSON-RPC) protocol[28]
- (3) **Language Server** - receives and dispatches the output from the daemon server to the supported text editors and IDEs. It communicates with editors via Microsoft’s Language Server Protocol (LSP)[44].
- (4) **Editor Extension** - acts as a client for receiving and displaying the output from the language server to the text editor diagnostics to be displayed to the user.

The researchers chose the Go programming language[18] for developing ErrgoEngine library and BugBuddy server due to its performance, security, and package availability [50, 60]. Visual Studio Code was used for development and testing the editor extension, known for its flexibility, ease of development via its extension API[43], and compatibility with LSP[45]. TypeScript, a superset of JavaScript with type annotations[46], was used to create the extension for VSCode.

3 EVALUATION

In order to provide initial insights on the usability of ErrgoEngine, the following objectives were established:

RQ1: Test Programs and Error Templates Development. What set of test programs and error message templates or rules should be developed to thoroughly evaluate the performance of the engine?

RQ2: Contextualization Mechanism Design. How can a mechanism be designed and implemented to enable the contextualization of error messages by the translation engine?

RQ3: Real-world Use. How can the ErrgoEngine library and application be integrated with existing development tools and environments to enhance its accessibility and ease of use for programmers?

3.1 RQ1: Test Programs and Error Templates Development

3.1.1 Data Gathering. The researchers collected data from previous studies and articles to determine the programming errors when creating error templates. First, the researchers identified the programming languages that would be used in this study. Based on the current programming curriculum used by the computer studies department, the researchers chose the Java programming language. In addition to Java, the researchers also included the Python programming language as a sample implementation to test the language-agnostic capabilities of ErrgoEngine.

Afterwards, studies and articles collecting the most common programming errors of Java[4, 7, 11, 42, 47, 49] and Python[22, 29, 55, 68] programming languages from students and developers were selected, ensuring a comprehensive coverage of programming errors made from individuals with varying levels of experience. Errors that do not output an error message such as logical programming errors were excluded. Upon collection, the researchers identified 49 programming errors which consisted of 43 programming errors for Java and 6 programming errors for Python. The reason for the low number of supported programming errors for Python compared to Java is due to only it being a test implementation.

3.1.2 Test Programs. For each identified programming error, the researchers utilized ChatGPT to generate the program code that would reproduce the error. The code was then verified running it on a language interpreter. In some cases, modifications were also made to the code to ensure that the exact programming error was thrown. Actual error messages from the interpreter were also extracted for the error template design. Upon running the test programs, the researchers discovered that some errors, particularly the Java *expected* and *cannot find symbol* programming errors, produced similar messages. These errors were merged with other similar errors and kept as additional test cases for testing. As a result, the number has been reduced to 42 programming errors.

3.1.3 Error Template Design. The error template format encapsulates essential components which consists of an error name, a regular expression pattern for error message matching, and crucial functions for detecting error positions, generating explanations, and crafting bug fixes. In addition, custom regular expression patterns for error message and stack trace formats were added to handle such situations found in dynamic interpreted languages that provide different error message and stack trace formats for compile-time and runtime errors.

3.1.4 Translated Error Output Design. Shown in Figure 4, the Markdown-based output consists of an error explanation and bug fix suggestions. Error explanation offers a brief description of the error, accompanied with the code snippet pinpointing the exact location of the error. Following this, bug fix suggestions provide one or more recommendations, each equipped with step-by-step instructions and a visualized code snippet of the fix to guide developers in resolving the identified issue.

3.1.5 Testing. The effectiveness in terms of detection and output generation was evaluated through unit testing. The test cases

```
# PrivateAccessError
This error occurs when you try to access a private variable from another
class, which is not allowed.
...

// Attempting to access a private variable from another class
int value = anotherClass.privateVariable;
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
System.out.println(value);
...
}

### Steps to fix
### 1. Use a public accessor method
1. To access a private variable from another class, create a public
accessor method in 'AnotherClass'.
'''diff
class AnotherClass {
    private int privateVariable = 10;
+
+   public int getPrivateVariable() {
+       return this.privateVariable;
+   }
+
+ }
...
}

2. Then, use this method to get the value.
'''diff
AnotherClass anotherClass = new AnotherClass();
// Attempting to access a private variable from another class
int value = anotherClass.privateVariable;
+ int value = anotherClass.getPrivateVariable();
System.out.println(value);
...
}

This way, you respect encapsulation by using a method to access the
private variable.

### 2. Make the variable public (not recommended)
If you must access the variable directly, you can make it public, but
this is generally not recommended for maintaining encapsulation.
'''diff
class AnotherClass {
-   private int privateVariable = 10;
+   public int privateVariable = 10;
}
...
}
```

Figure 4: Sample engine-translated output for Java’s private access error

	Java	Python	Total
Implemented test cases	44	6	50
No. of passed tests	36 (81.81%)	5 (83.33%)	41 (73.21%)
No. of failed tests	1 (2.27%)	1 (16.67%)	2 (3.57%)
No. of skipped tests	7 (15.9%)	0	7 (15.9%)
Code coverage	92.7%	97.4%	95.05% (avg)

Table 1: Error template unit testing statistics summary

consisted of input test programs, extracted programming error messages, and the expected engine-generated output. Each identified error was given one test case. To ensure comprehensive coverage, some programming errors were also given additional test cases. These test cases were taken from the similar errors identified earlier, as well as new test cases that addressed scenarios with unexpected bugs. Based on the results shown in Table 1, 50 test cases were created in total with 44 (78.57%) test cases created for the Java programming language while 6 (10.71%) test cases were created for Python. During the test, 44 (79.55%) test cases were able to pass while there were 2 test cases (3.57%) that failed and 7 test cases

Component	Test Passed / Failed / Skipped	Code Coverage
Initial context data extraction	6/0/0	39%
Error stack trace detection	4/0/0	25.96%
Semantic Analysis - File Opening	5/0/0	37.93%
Semantic Analysis - Symbol Collection	1/0/0 - Java 1/0/0 - Python	33.3% - Java 34.2% - Python

Table 2: Contextualization mechanism implementation unit testing results

(15.9%) that were skipped due to missing standard library symbol information and lack of implementation.

3.1.6 Programming Error Coverage. The researchers were not able to fully implement all of the identified programming errors. 29 of the 36 Java programming errors (80.55%) and 5 of 6 Python programming errors (83.33%) were successfully implemented. In total, there are 34 of 42 programming errors (80.95%) that are currently implemented and supported by ErrgoEngine. Apart from the reasons mentioned earlier, the incomplete implementation support was also due to its inability to process error messages of the same programming error in different versions of the same programming language which was the case for Java. Another obstacle was in bug fixing in which the lack of code formatting information would make ErrgoEngine provide inaccurate fixes that involve adding characters or braces for example.

3.2 RQ2: Contextualization Mechanism Design

3.2.1 Implementation Testing. The researchers employed unit testing into the implementations of its contextualization mechanism which includes initial context data extraction, error stack trace extraction, and semantic analysis which is further divided into file opening and symbol collection tests. The initial context data extraction process demonstrated strengths in accurately extracting relevant information from error messages using regular expressions and passing all tests with a test coverage of 39%. However, the test coverage could be improved to ensure more comprehensive testing. The error stack trace extraction process also showed strengths in accurately extracting data from raw stack trace text and handling partially invalid inputs, but the test coverage of 25.96% was lower than that of the initial context data extraction process. The semantic analysis process revealed weaknesses in the system, including issues with feature parity of symbols collected for each supported language, limited support of the importing dependencies implementation, and inconsistencies in the order of outputs generated.

3.2.2 Error Position Adjustment using Context Data. Added at the end of the error analysis process, it is achieved by introducing an additional property into the error template. This addition is a crucial part as stated in the repair theory as the added process dictates where the error in the code occurred, utilizing information gathered during the preceding steps of error analysis. It provided

Test Case	Test Passed / Failed / Skipped	Test Coverage
Explanation generator	13/0/0	95.5%
Bug fix Generator	16/0/0	
Output Generator	8/0/0	84.3%

Table 3: ErrgoEngine error translation testing results

Component	Test Passed / Failed / Skipped	Test Coverage
Program Executor	4/2/0	93.8%
Daemon Server	15/0/0	66.1%
Information Logger	17/0/0	85.3%
Language Server	13/0/0	53.6%

Table 4: BugBuddy server unit testing results

enhanced fault localization especially in errors that lack column position data and improved code organization and clear separation of concerns in terms of error template implementation.

3.3 RQ3: Real-world Use

3.3.1 Unit testing. The researchers conducted unit testing to evaluate the performance and reliability of the ErrgoEngine library and BugBuddy application. For ErrgoEngine, the researchers concentrated on testing the components involved in the error translation stage, as error analysis had already been addressed during the error template and contextual mechanism tests. The error translation stage consists of three main components: the explanation generator, bug fix generator, and output generator.

The explanation generator and bug fix generator has shown strengths in handling their respective tasks, with high test coverage rates of 95.5%. However, the output generator component showed a potential weakness, with a lower test coverage of 84.3%. While all 8 test cases were successfully passed, the lower test coverage suggests that there may be some scenarios that were not tested. Further testing may be needed to ensure the robustness of the output generator component. Overall, the high test coverage rates for the explanation generator and bug fix generator components indicate their reliability in generating informative explanations and bug fixes, but more work may be needed to improve the output generator component.

As for BugBuddy, the testing process is divided into server application tests and extension tests. For the server application tests, the researchers evaluated the core components of BugBuddy, which include the program executor, daemon server, information logger, and language server.

Results in table 4 reveal both strengths and weaknesses in these components. The program executor had a high code coverage of 93.8% but only a 66.67% pass rate due to issues with processing multiple programming errors. The Daemon Server had a perfect pass rate of 100% but a relatively low code coverage of 66.1%. The Information Logger had a high pass rate of 100% and a code coverage of 85.3%, but there were still areas of the code that were not tested. The Language Server had a pass rate of 100% but a

very low code coverage of 53.6%, indicating that more testing is needed to ensure its robustness and accuracy. Overall, while the testing results demonstrate the effectiveness and reliability of the BugBuddy framework's components, there are still areas that need improvement, particularly in terms of code coverage and processing multiple programming errors. The researchers plan to address these weaknesses and continue refining the framework to improve its overall performance.

The BugBuddy extension tests, on the other hand, were all successful, including the connection from the server to the editors and receiving of translated error outputs. However, it is worth noting that the tests only validated the core functionalities of the extension, and further testing may be necessary to ensure its suitability for a wider range of projects and programming languages. Overall, the successful tests highlight the strengths of the extension as a user-friendly and robust interface for BugBuddy.

3.3.2 Initial usability test. Another test was conducted to evaluate the efficacy and user experience of the BugBuddy application. The assessment consists of three (3) student participant testers which were all given a set of sample programs that to use to test in terms of setup, functionality, and overall usability. Results indicated a generally positive reception among participants. The setup process was straightforward, with participants successfully installing and configuring both the BugBuddy application and its accompanying extension within their preferred text editors. It also demonstrated accurately in capturing and translating programming error messages during code execution, thereby assisting users in pinpointing errors and providing relevant explanations and suggestions for resolution. Participants found the explanations and suggestions to be concise, comprehensible, and beneficial for enhancing their understanding of programming concepts.

However, there are limitations of the assessment. The controlled environment and the modest sample size may not encapsulate the full spectrum of user interactions with BugBuddy in real-world programming environments. While the initial usability assessment offers valuable insights into BugBuddy's user experience, a more comprehensive performance evaluation is yet to be done right after the usability tests.

4 CONCLUSION

The evaluation of ErrgoEngine provided valuable insights into its usability and effectiveness. The researchers successfully developed a set of test programs and error templates for 42 programming errors, which were reduced from 49 after merging similar errors. The contextualization mechanism was designed and implemented, enabling the translation engine to provide more accurate and relevant error messages. The ErrgoEngine library and BugBuddy application were integrated with existing development tools and environments, enhancing their accessibility and ease of use for programmers. The unit testing results showed that the error template and contextual mechanism tests were successful, with high test coverage rates. However, there were some weaknesses identified in the semantic analysis process, which will need to be addressed in future work. The error translation stage also demonstrated strengths in handling its tasks, with high test coverage rates for the explanation generator

and bug fix generator components. However, the output generator component showed potential weaknesses, and further testing may be needed to ensure its robustness. The initial evaluation results also shown it was able to capture errors and is easy to setup. Overall, the evaluation results suggest that ErrgoEngine has the potential to be a useful tool for improving the debugging process for programmers.

REFERENCES

- [1] [n. d.]. CommonMark. <https://commonmark.org/>
- [2] Rui Abreu, Peter Zoetewij, and Arjan Gemund. 2007. *On the Accuracy of Spectrum-based Fault Localization*. <https://doi.org/10.1109/TAIC.PART.2007.13> Journal Abbreviation: Proceedings - Testing: Academic and Industrial Conference Practice and Research Techniques, TAIC PART-Mutation 2007 Pages: 98 Publication Title: Proceedings - Testing: Academic and Industrial Conference Practice and Research Techniques, TAIC PART-Mutation 2007.
- [3] Saman Amarasinghe, Adam Chlipala, Srinivas Devadas, Michael Ernst, Max Goldman, John Guttag, Daniel Jackson, Rob Miller, Martin Rinard, and Armando Solar-Lezama. [n. d.]. Reading 18: Parser Generators. <https://web.mit.edu/6.005/www/fa15/classes/18-parser-generators/>
- [4] Brett A. Becker. 2016. An Effective Approach to Enhancing Compiler Error Messages. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. ACM, Memphis Tennessee USA, 126–131. <https://doi.org/10.1145/2839509.2844584>
- [5] Brett A. Becker, Paul Denny, Raymond Pettit, Durell Bouchard, Dennis J. Bouvier, Brian Harrington, Amir Kamil, Amey Karkare, Chris McDonald, Peter-Michael Osera, Janice L. Pearce, and James Prather. 2019. Compiler Error Messages Considered Unhelpful: The Landscape of Text-Based Programming Error Message Research. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*. ACM, Aberdeen Scotland UK, 177–210. <https://doi.org/10.1145/3344429.3372508>
- [6] Moritz Beller, Niels Spruit, Diomidis Spinellis, and Andy Zaidman. 2018. On the dichotomy of debugging behavior among programmers. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, Gothenburg Sweden, 572–583. <https://doi.org/10.1145/3180155.3180175>
- [7] Mordechai Moti Ben-Ari. 2007. Compile and runtime errors in java.]- : http://introcs.cs.princeton.edu/java/11_cheatsheet/errors.pdf (2007).
- [8] John Seely Brown and Kurt VanLehn. 1980. Repair Theory: A Generative Theory of Bugs in Procedural Skills. *Cognitive Science* 4, 4 (Oct. 1980), 379–426. https://doi.org/10.1207/s15516709cog0404_3
- [9] Max Brunsfield. 2018. Tree-sitterIntroduction. <https://tree-sitter.github.io/tree-sitter/>
- [10] Heling Cao, YangXia Meng, Jianshu Shi, Lei Li, Tiaoli Liao, and Chenyang Zhao. 2020. A Survey on Automatic Bug Fixing. In *2020 6th International Symposium on System and Software Reliability (ISSSR)*. IEEE, Chengdu, China, 122–131. <https://doi.org/10.1109/ISSSR51244.2020.00029>
- [11] Ioana Chan Mow. 2012. Chan Mow I (2012) Analyses of Student Programming Errors In Java Programming Courses Journal of Emerging Trends in Computing and Information Sciences 3(5) pp 740 - 749. *Analyses of Student Programming Errors In Java Programming Courses* 3 (May 2012), 740–749.
- [12] Adrian Cordiner. 2021. What Is the Kolb Experiential Learning Theory? <https://practera.com/what-is-the-experiential-learning-theory-of-david-kolb/>
- [13] Paul Denny, Andrew Luxton-Reilly, and Dave Carpenter. 2014. Enhancing syntax error messages appears ineffectual. In *Proceedings of the 2014 conference on Innovation & technology in computer science education - ITiCSE '14*. ACM Press, Uppsala, Sweden, 273–278. <https://doi.org/10.1145/2591708.2591748>
- [14] Sue Fitzgerald, Gary Lewandowski, Renée Mccauley, Laurie Murphy, Beth Simon, Lynda Thomas, and Carol Zander. 2008. Debugging: Finding, fixing and flailing, a multi-institutional study of novice debuggers. *Computer Science Education* 18 (June 2008). <https://doi.org/10.1080/08993400802114508>
- [15] T. Flowers, Curtis Carver, and J. Jackson. 2004. Empowering students and building confidence in novice programmers through Gauntlet. *T3H/10-T3H/13* Vol. 1. <https://doi.org/10.1109/FIE.2004.1408551>
- [16] Xiang Gao, Yannic Noller, and Abhik Roychoudhury. 2022. Program Repair. <http://arxiv.org/abs/2211.12787> arXiv:2211.12787 [cs].
- [17] Abdul Rahman Mohamad Gobil, Zarina Shukor, and Itaza Afiani Mohtar. 2009. Novice difficulties in selection structure. In *2009 International Conference on Electrical Engineering and Informatics*. IEEE, Bangi, Malaysia, 351–356. <https://doi.org/10.1109/ICEEL.2009.5254715>
- [18] Google. 2011. The Go Programming Language. <https://go.dev/>
- [19] Carl Gwilliam. 2022. error_explainer/error_explainer.py at main · carlgwilliam/error_explainer · GitHub. https://github.com/carlgwilliam/error_explainer/blob/main/error_explainer.py

- [20] Mercedes Gómez-Albarrán. 2005. The Teaching and Learning of Programming: A Survey of Supporting Software Tools. *Comput. J.* 48, 2 (Jan. 2005), 130–144. <https://doi.org/10.1093/comjnl/bxh080>
- [21] Matthew Heinsen Egan and Chris McDonald. 2013. Runtime Error Checking for Novice C Programmers. https://doi.org/10.5176/2251-2195_CSEIT13.03
- [22] Petri Ihantola, Arto Hellas, Matthew Butler, Jürgen Börstler, Stephen Edwards, Essi Isohanni, Ari Korhonen, Andrew Petersen, Kelly Rivers, Miguel Rubio, Judy Sheard, Bronius Skupas, Jaime Spacco, Claudia Szabo, and Daniel Toll. 2015. Educational Data Mining and Learning Analytics in Programming: Literature Review and Case Studies. <https://doi.org/10.1145/2858796.2858798>
- [23] InstructionalDesign.org. 2022. Repair Theory (K. VanLehn). <https://www.instructionaldesign.org/theories/repair-theory/>
- [24] Noman Islam, Ghazala Sheikh, Ridah Fatima, and Farrukh Alvi. 2019. A Study of Difficulties of Students in Learning Programming. *Journal of Education & Social Sciences* 7 (Oct. 2019), 38–46. <https://doi.org/10.20547/jess0721907203>
- [25] Billy Javier. 2021. Understanding their voices from within: difficulties and code comprehension of life-long novice programmers. *International Journal of Arts, Sciences and Education* 1, 1 (2021), 53–73.
- [26] JobStreet.com.my. 2021. What Is Coding And Why Is It Important. <https://www.jobstreet.com.my/career-resources/plan-your-career/5-reasons-why-coding-is-an-important-skill-for-your-career-growth/>
- [27] Stephen C Johnson. 1975. *Yacc: Yet another compiler-compiler*. Vol. 32. Bell Laboratories Murray Hill, NJ.
- [28] JSON-RPC Working Group. [n. d.]. JSON-RPC 2.0 Specification. <https://www.jsonrpc.org/specification>
- [29] Anne K. Kelley. 2018. *A system for classifying and clarifying Python syntax errors for educational purposes*. Thesis. Massachusetts Institute of Technology. <https://dspace.mit.edu/handle/1721.1/119750> Accepted: 2018-12-18T19:48:27Z.
- [30] D. A. Kolbe. 1984. Experiential learning. *New Jersey, Eaglewood Cliffs* (1984).
- [31] Xuan-Bach D. Le, Ferdian Thung, David Lo, and Claire Le Goues. 2018. Overfitting in semantics-based automated program repair. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, Gothenburg Sweden, 163–163. <https://doi.org/10.1145/3180155.3182536>
- [32] Juho Leinonen, Arto Hellas, Sami Sarsa, Brent Reeves, Paul Denny, James Prather, and Brett A. Becker. 2023. Using Large Language Models to Enhance Programming Error Messages. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 563–569. <https://doi.org/10.1145/3545945.3569770>
- [33] Xuliang Liu and Hao Zhong. 2018. *Mining stackoverflow for program repair*. <https://doi.org/10.1109/SANER.2018.8330202> Pages: 129.
- [34] Jason M. Lodge, Gregor Kennedy, Lori Lockyer, Amael Arguel, and Mariya Pachman. 2018. Understanding Difficulties and Resulting Confusion in Learning: An Integrative Review. *Frontiers in Education* 3 (2018). <https://www.frontiersin.org/articles/10.3389/feeduc.2018.00049>
- [35] Phil Lombardi. 2023. What Is Debugging? (Plus 8 Important Strategies To Try) | Indeed.com. <https://www.indeed.com/career-advice/career-development/debugging>
- [36] Andrew Luxton-Reilly, Emma McMillan, Elizabeth Stevenson, Ewan Tempero, and Paul Denny. 2018. Ladebug: an online tool to help novice programmers improve their debugging skills. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*. ACM, Larnaca Cyprus, 159–164. <https://doi.org/10.1145/3197091.3197098>
- [37] Guillaume Marceau, Kathi Fisler, and Shiram Krishnamurthi. 2011. Mind your language: on novices' interactions with error messages. In *Proceedings of the 10th SIGPLAN symposium on New ideas, new paradigms, and reflections on programming and software*. ACM, Portland Oregon USA, 3–18. <https://doi.org/10.1145/2048237.2048241>
- [38] Wes Masri. 2015. Automated Fault Localization. In *Advances in Computers*. Vol. 99. Elsevier, 103–156. <https://doi.org/10.1016/bs.adcom.2015.05.001>
- [39] Davin McCall. 2016. *Novice Programmer Errors - Analysis and Diagnostics*. phd. University of Kent. <https://kar.kent.ac.uk/61340/>
- [40] Davin McCall and Michael Kölling. 2019. A New Look at Novice Programmer Errors. *ACM Transactions on Computing Education* 19, 4 (Dec. 2019), 1–30. <https://doi.org/10.1145/3335814>
- [41] Renée Mccauley, Sue Fitzgerald, Gary Lewandowski, Laurie Murphy, Beth Simon, Lynda Thomas, and Carol Zander. 2008. Debugging: A review of the literature from an educational perspective. *Computer Science Education* 18 (June 2008). <https://doi.org/10.1080/08993400802114581>
- [42] Cameron McKenzie. 2022. Fix these 10 common examples of the Runtime-Exception in Java | TheServerSide. <https://www.theserverside.com/tip/Fix-these-10-common-examples-of-the-RuntimeException-in-Java>
- [43] Microsoft. 2015. Why Visual Studio Code? <https://code.visualstudio.com/docs/editor/whyvscode>
- [44] Microsoft. 2016. Official page for Language Server Protocol. <https://microsoft.github.io/language-server-protocol/>
- [45] Microsoft. 2019. Language Server Extension Guide. <https://code.visualstudio.com/api/language-extensions/language-server-extension-guide>
- [46] Microsoft. 2020. Documentation - TypeScript for the New Programmer. <https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html>
- [47] Kabiru Mijinyawa, Murtala Mohammed, Abdulwahab Lawan, and Bashir Galadanci. 2015. Detection and Categorization of Errors by Novice Programmers in a First Year Java Programming Class.
- [48] IT. Chan Mow. 2008. Issues and Difficulties in Teaching Novice Computer Programming. In *Innovative Techniques in Instruction Technology, E-learning, E-assessment, and Education*, Maged Iskander (Ed.). Springer Netherlands, Dordrecht, 199–204. https://doi.org/10.1007/978-1-4020-8739-4_36
- [49] Christian Murphy, Eunhee Kim, Gail Kaiser, and Adam Cannon. [n. d.]. Backstop: A Tool for Debugging Runtime Errors. ([n. d.]).
- [50] Jacek Olszak and Karolina Rusinowicz. 2022. Why Golang may be a good choice for your project. <https://codilime.com/blog/why-golang/>
- [51] Tom Ormerod. 1990. Human Cognition and Programming. Elsevier, 63–82. <https://doi.org/10.1016/B978-0-12-350772-3.50009-4> Book Title: Psychology of Programming.
- [52] T. J. Parr and R. W. Quong. 1995. ANTLR: A predicated-LL(k) parser generator. *Software: Practice and Experience* 25, 7 (1995), 789–810. <https://doi.org/10.1002/spe.4380250705> _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/spe.4380250705>.
- [53] David Perkins and Fay Martin. 1985. *Fragile Knowledge and Neglected Strategies in Novice Programmers*. IR85-22. Technical Report. <https://eric.ed.gov/?id=ED295618> ERIC Number: ED295618.
- [54] Raymond S. Pettit, John Homer, and Roger Gee. 2017. Do Enhanced Compiler Error Messages Help Students?: Results Inconclusive.. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*. ACM, Seattle Washington USA, 465–470. <https://doi.org/10.1145/3017680.3017768>
- [55] David Pritchard. 2015. Frequency distribution of error messages. In *Proceedings of the 6th Workshop on Evaluation and Usability of Programming Languages and Tools*. ACM, Pittsburgh PA USA, 1–8. <https://doi.org/10.1145/2846680.2846681>
- [56] Siti Rosminah, siti rosminah md derus, and Ahmad Zamzuri Mohamad Ali. 2012. Difficulties in learning programming: Views of students. <https://doi.org/10.13140/2.1.1055.7441>
- [57] Dominik Sobania, Martin Briesch, Carol Hanna, and Justyna Petke. 2023. An Analysis of the Automatic Bug Fixing Performance of ChatGPT. <http://arxiv.org/abs/2301.08653> arXiv:2301.08653 [cs].
- [58] Shin Hwei Tan, Hiroaki Yoshida, Mukul R. Prasad, and Abhik Roychoudhury. 2016. Anti-patterns in search-based program repair. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. ACM, Seattle WA USA, 727–738. <https://doi.org/10.1145/2950290.2950295>
- [59] Team GyaniPandit. 2023. Advantages and Disadvantages of Prompt Engineering. <https://gyanipandit.com/programming/advantages-and-disadvantages-of-prompt-engineering/> Section: Prompt Engineering.
- [60] The Go Programming Language. 2011. Go Packages. <https://pkg.go.dev/>
- [61] Todd Kelsey PhD. 2023. 20 Prompt Engineering Paint Points | LinkedIn. <https://www.linkedin.com/pulse/20-prompt-engineering-paint-points-todd-kelsey-phd/>
- [62] Iris Vessey. 1985. Expertise in debugging computer programs: A process analysis. *International Journal of Man-Machine Studies* 23, 5 (Nov. 1985), 459–494. [https://doi.org/10.1016/S0020-7373\(85\)80054-7](https://doi.org/10.1016/S0020-7373(85)80054-7)
- [63] Chu-Pan Wong, Yingfei Xiong, Hongyu Zhang, Dan Hao, Lu Zhang, and Hong Mei. 2014. Boosting Bug-Report-Oriented Fault Localization with Segmentation and Stack-Trace Analysis. In *Proceedings of the 2014 IEEE International Conference on Software Maintenance and Evolution (ICSME '14)*. IEEE Computer Society, USA, 181–190. <https://doi.org/10.1109/ICSME.2014.40>
- [64] W. Eric Wong, Ruizhi Gao, Yihao Li, Rui Abreu, and Franz Wotawa. 2016. A Survey on Software Fault Localization. *IEEE Transactions on Software Engineering* 42, 8 (Aug. 2016), 707–740. <https://doi.org/10.1109/TSE.2016.2521368>
- [65] Stelios Xinogalos. 2016. Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. *Education and Information Technologies* 21, 3 (May 2016), 559–588. <https://doi.org/10.1007/s10639-014-9341-9>
- [66] Michihiro Yasunaga and Percy Liang. 2020. Graph-based, Self-Supervised Program Repair from Diagnostic Feedback. <http://arxiv.org/abs/2005.10636> arXiv:2005.10636 [cs, stat].
- [67] Zhongxing Yu, Matias Martinez, Benjamin Danglot, Thomas Durieux, and Martin Monperrus. 2019. Alleviating patch overfitting with automatic test generation: a study of feasibility and effectiveness for the Nopol repair system. *Empirical Software Engineering* 24, 1 (Feb. 2019), 33–67. <https://doi.org/10.1007/s10664-018-9619-4>
- [68] Zihé Zhou, Shijuan Wang, and Yizhou Qian. 2021. Learning From Errors: Exploring the Effectiveness of Enhanced Error Messages in Learning to Program. *Frontiers in Psychology* 12 (2021). <https://www.frontiersin.org/articles/10.3389/fpsyg.2021.768962>

Immune Survivors: A Serious Game for Raising Immune System Awareness

Junma Ogawa*
De La Salle University
Manila, Metro Manila
junma_ogawa@dlsu.edu.ph

Setiel Olivarez*
De La Salle University
Manila, Metro Manila
setiel_b_olivarez@dlsu.edu.ph

James Gabriel Omalin*
De La Salle University
Manila, Metro Manila
james_omalin@dlsu.edu.ph

Gil Perfecto Noel Alva*
De La Salle University
Manila, Metro Manila
gil_alva@dlsu.edu.ph

Ian Joshua Nemeño*
De La Salle University
Manila, Metro Manila
ian_nemeno@dlsu.edu.ph

Thomas James Tiam-Lee
De La Salle University
Manila, Metro Manila
thomas.tiam-lee@dlsu.edu.ph

ABSTRACT

The immune system is a crucial and relevant component of the human body, yet many people lack awareness on it. To address this, we developed a serious game called *Immune Survivors* that aims to spread awareness about the immune system while providing an engaging gameplay loop. In this paper, we discuss the design, implementation, and evaluation of *Immune Survivors*, and how we aligned it to a set of target learning objectives about the immune system. *Immune Survivors* is a “bullet heaven” game where you play the role of immune cells aiming to protect the human body from invading pathogens. Quasi-experimental results on late high school to early college students reveal significant learning gains, as measured through a pre-test and post-test, after an hour of playing the game. Furthermore, the results suggest that more substantial learning gains can be observed on concepts that were integrated into the gameplay in a more critical and ubiquitous way. These findings show the potential of serious games as a tool to supplement gaps in education while highlighting the need for deeper studies within this domain.

CCS CONCEPTS

• **Applied computing** → **Interactive learning environments; Computer games; E-learning;**

KEYWORDS

serious games, e-learning, immune system

ACM Reference Format:

Junma Ogawa, Setiel Olivarez, James Gabriel Omalin, Gil Perfecto Noel Alva, Ian Joshua Nemeño, and Thomas James Tiam-Lee. 2024. Immune Survivors: A Serious Game for Raising Immune System Awareness. In *Proceedings of Philippine Computing Science Congress (PCSC2024)*. Laguna, Philippines, 8 pages.

*These authors contributed equally to the paper

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
PCSC2024, May 2024, Laguna, Philippines
© 2024 Copyright held by the owner/author(s).

1 INTRODUCTION

Serious games have demonstrated great potential in education and in raising awareness. A serious game is a game that is designed with a primary goal – usually education, apart from just mere entertainment [6]. Serious games, both as an industry and as a research field, has seen continuous growth in recent years, underscoring its relevance and promise [15, 25]. They are commonly used in education and training, and has seen applications in a wide variety of domains. Some recent applications of serious games include skills development for kids with autism spectrum disorder [8, 13], climate change awareness [11, 19], bullying and cyberbullying prevention [7, 12], and disaster training and response [10, 20].

The potential of serious games in learning and raising awareness is attributed to the idea that students can learn more effectively when they are engaged. Research has shown that positive affective states such as engagement positively impacts learning, in contrast to negative affective states like boredom [5]. Games have been shown to enable emotion formation effectively, and thus are appropriate mediums to induce positive engagement on players to facilitate learning [18]. Apart from this, games have the potential to introduce to its players concepts and ideas that they would otherwise be uninterested to explore on their own. As such, serious games can be used to entice people to explore and learn about topics that are normally not given much attention in standard learning curricula but are nonetheless important to life and society.

The immune system is an example of an important and relevant, yet often overlooked topic. The immune system is an integral part of the human body tasked with protecting us from diseases, viruses, pathogens, and other entities that may bring harm. Despite the critical role that it plays in our lives, many people have a surface level understanding of the immune system or may even have misconceptions on how it functions [4, 17]. In the Philippine curriculum, various systems in the human body are introduced from 6th grade to 12th grade, but it fails to provide coverage on the immune system and immunology [22–24]. Although parts of the immune system may be covered within other topics, a search in the Department of Education learning portal does not reveal any material or module that provides a substantial coverage of the immune system [1]. This is quite surprising considering the crucial role the immune system plays in our bodies. The COVID-19 pandemic has further underscored the importance of awareness about our health, the immune system included. According to Google Trends, searches

for the immune system peaked in the Philippines during March 15 to 21 in the year 2020 [2]. This highlights the increased public interest in learning about the immune system.

Serious games can be a tool to help address the gaps in education, by introducing these concepts to students in an engaging environment. In this paper, we explore the design and development of a serious game for introducing the fundamentals of the immune system called *Immune Tactics*. The game combines the emerging “bullet heaven” genre, which has gained immense popularity in recent years [9], with fundamental immune system concepts to educate players as they play. This paper discusses the design, development, and evaluation of *Immune Survivors*, as well as the key findings and learnings from its implementation.

The paper is structured as follows. Section 2 discusses related serious games that cover the immune system. Section 3 discusses the game design and implementation of *Immune Survivors*. Section 4 discusses the evaluation process to test the effectiveness of the game. Section 5 discusses the results and findings. Finally, Section 6 provides conclusions and directions for future work.

2 RELATED WORKS

In this section, we introduce related serious games that also aim to teach concepts about the immune system.

2.1 ImmuneQuest

ImmuneQuest is a serious game that aims to teach players immunology [3]. In the game, players take the role of a scientist who must build and control a virtual immune system to defend their human host from increasingly cunning microbes. It is a turn-based strategy game, akin to a board game, where you control units of the immune system to protect the body from enemies. Each unit has abilities and properties based on their real-world counterparts. Learning takes place through a combination of participating in the gameplay, and through tooltips and messages that occasionally pop-up to provide more information about the concepts.



Figure 1: ImmuneQuest gameplay screenshot

The game plays out like a campaign. The story introduces different aspects of the immune system, from macrophages, neutrophils, blood vessels, and how they interact with one another. However, so far only the first two parts have been published, covering the topic of “innate immunity”. Due to the nature of the gameplay, replayability can be limited, as once the campaign has been completed there is little reason to play it again.

Empirical evaluation has shown that ImmuneQuest led to positive learning effects in an undergraduate immunology course, as measured through pre-tests and post-tests [21]. The evaluation also revealed that the average enjoyability rating of the game was 6.6 / 10. Furthermore, many students found the game to be “frustrating”. These can be attributed to the serious presentation and nature of the gameplay. Nonetheless, *ImmuneQuest* remains to be a solid evidence that serious games can indeed help educate people in serious topics like the immune system, although it is mainly targeted towards undergraduate biology majors, as it requires some preliminary knowledge.

2.2 Immune Attack / Immune Defense

Immune Attack and *Immune Defense* are strategy games about molecular immunity [16]. In both of these games, the player assumes the role of the immune system. At its core, it is a shooter strategy game, with the main control being the mouse. Both games follow a handcrafted script that follows a story, designed to introduce immunology concepts to the students. Empirical tests revealed that those who played *Immune Attack* demonstrated better performance on a test of cellular immunology, and had better self-confidence than their classmates who played a control video game [26]. However, the increase in confidence was only observed on the non-gamer group, and a positive correlation was observed between understanding of the game mechanisms and the improvement in performance, which suggest that learning is more effective when the students understand and enjoy the game.

2.3 Conflict: Immunity Game

Conflict: Immunity Game is a web game by BioMan, a web site containing various virtual labs and games about biology. The game is a simple game that introduces the functions of the different white blood cells. Pop-up message show the functions of each white blood cells, then the player is tasked to control these cells through simple actions, such as moving a macrophage towards a pathogen, or using the space bar to shoot antibodies on a pathogen.

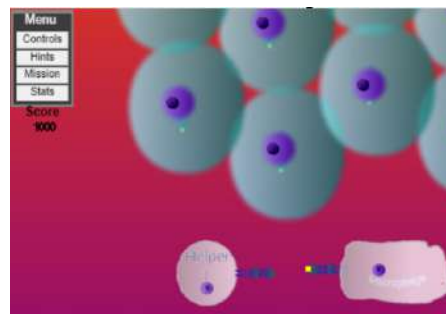


Figure 2: Conflict: Immunity gameplay screenshot

While the game is straightforward and easy to understand, there is not much complexity in its gameplay. Furthermore, to the authors’ knowledge, there has not been any empirical evaluation on the effectiveness of *Conflict: Immunity Game* in terms of its learning effects or learning.

Table 1: Summary of Published Serious Games that Teach Immune System Concepts

Game	Genre	Demographic	Evaluation	Key Findings
ImmuneQuest	Turn-Based Strategy	undergraduate university students taking immunology	quasi-experiment through pre-test and post-test	positive learning effect, but average enjoyability
Immune Attack / Immune Defense	Shooter Strategy	students (varied)	experiment with a control group playing a game unrelated to immune system	positive learning effect, increased confidence, but only for gamers group
Conflict: Immunity Game	Action	students (varied)	no evaluation	
Cells of War	Card Game	anyone	perceived quality evaluation only using MEEGA+	majority reported perceived positive learning and engagement

2.4 Cells of War

Finally, a serious game called *Cells of War* was proposed in 2018. It is a card game that aims to teach players the complex functionalities of the immune system, as well various daily habits affect a person’s health [14]. It is quite challenging to find detailed information about this game, as it appears to have not been published publicly. However, its gameplay was described as a turn-based card game where the player can a variety of cards related to the immune system. The goal of the game is to prevent the player’s health from dropping below zero while attempting to reach level 50.



Figure 3: Some cards in the game Cells of War [14]

A pilot evaluation of *Cells of War* was conducted using the MEEGA+ model, which examines the perceived quality of a serious game through two aspects: perceived learning and player experience. Majority of the participant reported that they learning something new by playing the game, although they did not consider that the topics were important to them. Majority of the players also reported that the game was fun, but only a minority reported that they were immersed in the game. These results show potential for the game, but may need further verification to assess its real learning potential, as perceived learning does not always equate to actual learning.

2.5 Summary

There are not a lot of serious games developed about the immune system, and even less have been empirically evaluated for their effectiveness. Nonetheless, the games discussed in this section show

that there is potential for serious games to be used in this domain. Furthermore, it appears that player experience plays an important role in the impact of a game in terms of learning. Table 1 summarizes the games discussed in this section. In this paper, we present a serious game for learning immune system awareness called *Immune Survivors* targeted towards high school and early undergraduate students, taking into consideration the insights from the above related games. We also present a detailed empirical evaluation on its effectiveness.

3 THE DESIGN OF IMMUNE SURVIVORS

In this section, we discuss the design and implementation of *Immune Survivors* – the serious game we developed for raising immune system awareness.

3.1 Learning Objectives

The game is aimed at three learning objectives:

- **LO1:** To leave players with a better appreciation for the role of the immune system and its significance in protecting the human body.
- **LO2:** To learn about the common elements involved in the process of the immune system in fighting off bacteria and viruses.
- **LO3:** To raise awareness of the implicit and explicit factors that could affect the immune system and its ability to protect the body from foreign entities.

3.2 Game Mechanics

The game is set in a microscopic view of the human body, artistically interpreted as a miniature city. At the start of the game, the player takes the role of a newly formed immune cell who is tasked with defending the body from a horde of invading pathogens. The gameplay follows a “bullet heaven” type of gameplay, a variant of the roguelite genre popularized by titles such as *Vampire Survivors* and *Brotato*. Figure 4 shows a screenshot of the game.

In the game, the player controls a set of immune cells that move around the map while dodging pathogens. The immune cell automatically attacks when there are nearby enemies. Each type of



Figure 4: Screenshot of *Immune Survivors*

immune cell has types of attacks that are aligned with its real-world characteristics of how it fights pathogens. Though some creative liberties were used in depicting the abilities of the immune cells, the general characteristics and properties of each cell type were preserved. Furthermore, the game explains the connection of the in-game representations of the cells with their real-world counterparts through text prompts that show various sorts of real-world information about the immune system.

Over time, other immune cells appear in the level, and can be recruited to increase the number of cells controlled by the player. The player still only controls the main cell, while the other recruited cells hover around the main cell. Once the player collects enough cells of the same type, the cell can be upgraded, and the player is presented a set of choices for the upgrade. These choices are randomly selected from a pool of possible upgrades for each cell. Most upgrades are based on characteristics of the immune cells in real life or factors that affect the immune system in some way.

When pathogens are defeated, they sometimes drop antigens which the dendritic cell can collect. The number of antigens collected is displayed in the game HUD as well. Once enough antigens are collected, helper T-Cells and B-Cells can be activated on the map. These cells are powerful units that provide huge utility in eliminating the pathogens. The helper T-Cell creates a cytokine buff zone around its radius, allowing the player to deal more damage to the pathogens that enter the zone. The B-Cell fires antibodies that shreds the defenses of pathogens that it hits. However, helper T-Cells and B-Cells only work for pathogens with the same type of antibody it was made from. The behavior of antigens, helper T-Cells and B-Cells are based on how these cells function in real life.

The goal of the game is to completely eradicate the ongoing infection by defeating all the pathogens within the level time. On the other hand, the player loses when the player’s health, which is affected by the amount of infection in the body, reaches zero. The player’s health is represented as a bar at the top of the HUD.

Each level is designed around a particular symptom. Symptoms are environmental hazards that first occur after 30 seconds, after which they will then occur every minute mark on the timer. Symptoms are different each level they are in. In the current version of the game used for testing, there are three levels. The first level has no symptoms. The second level focuses on fever, while the

third level focuses on cough. In the “fever” symptom, the body temperature momentarily rises, causing the player as well as all enemies to gain a damage over time debuff. The duration of the fever varies from one occurrence to another. In the “cough” symptom, a warning shows up in one of the cardinal directions. After the warning, all enemies will get pushed in the opposite direction of the warning. These symptoms are also accompanied with the relevant information regarding their real-life manifestations and how they affect the immune system response.

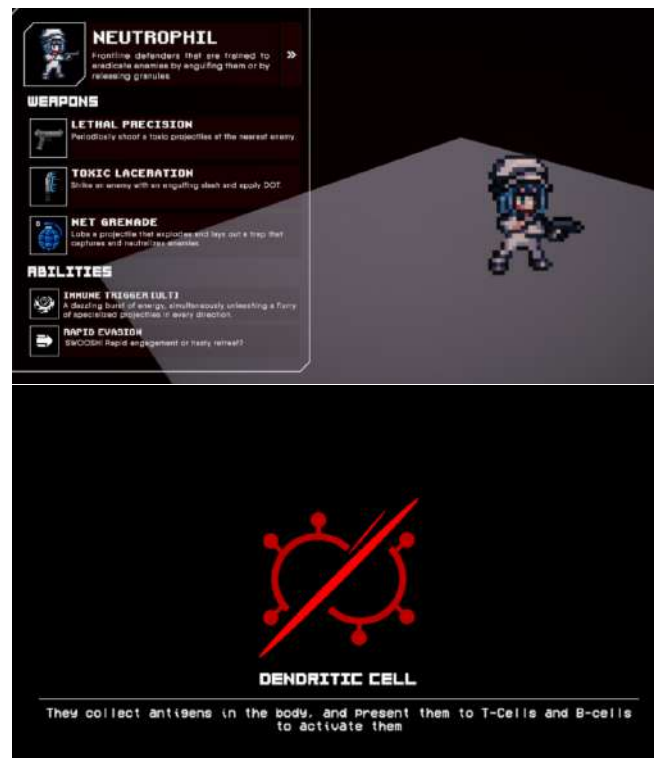


Figure 5: Text and Visual Prompts Used to Reinforce Immune System Concepts to the Player

3.3 Alignment to Learning Objectives

Immune Survivors is designed to facilitate learning through its core gameplay elements, as well as through supplementary prompts that introduce the player to the real-world concepts.

The game mechanics of *Immune Survivors* were designed in such a way that they aligned learning objectives described above. The game was developed using an iterative process of prototyping and playtesting, and the design process was done in consultation with a health expert who provided invaluable feedback and guidance. The following subsections describe how the game addresses each learning objective.

3.3.1 *Appreciation of the Role of the Immune System (LO1)*. This is reinforced in the game by having the player control the elements of the immune system themselves, giving them a feel of the role of the immune system, the threats that it faces, and how it protects the human body.

3.3.2 *Knowing the Elements of the Immune System Response (LO2).* This is taught to the player through the different units in the game, which were intentionally designed to have characteristics close to their real-world counterparts. For example, macrophages are specialized white blood cells that play an important role in maintaining homeostasis and protecting the body from diseases, which can engulf and destroy a wide range of pathogens like bacteria. In the game, this cell was designed as a tanky unit that specialized in engulfing swarms of bacteria.

3.3.3 *Awareness on Factors that Affect the Human Body (LO3).* Different levels in the game as designed around different “symptoms”, which involve specific factors that affect the human body. The current version of the game, which was used for testing, contained three levels. The first level had no additional factors. The second level is a level where the human host is experiencing a fever, and the third level is a level where the human host is experiencing a cough.

In addition to these, learning is also facilitated through informative text and visual prompts that explain the relationship of the in-game components with their real-life counterparts. These are intended to help reinforce to the player the connection between the game mechanics to how the immune system actually functions. Figure 5 some examples of these text and visual prompts.

4 EVALUATION

This section discusses the evaluation process we used to measure the effectiveness of the serious game. The evaluation process can be divided into three phases: the pre-test, play session, and post-test.

To do this, we conducted a playtesting session on students. We asked the participants to answer a pre-test and post-test questionnaire to measure the effects of playing the game. This section discusses the evaluation process, the results, and a discussion on its implications.

4.1 Participants

Following the target demographic of *Immune Survivors*, we recruited senior high school students as well as first year college students from De La Salle University Integrated School and De La Salle University Science and Technology Complex for the evaluation process. A total of 21 students participated in the evaluation process. Among them, 13 (61.9%) are first year college students and 8 (38.1%) are senior high school students.

The subjects voluntarily expressed their interest to participate in the evaluation process, through an open recruitment call. The relevant informed consent forms were administered to the participants, and the evaluation procedures, including how their data will be stored and processed, were properly explained to the participants prior to their commitment.

4.2 Evaluation Methodology

The evaluation methodology followed a quasi-experimental setup, and was comprised of three phases: the pre-test, play session, and post-test.

4.2.1 *Pre-test.* The pre-test consists of a set of questions administered to each subject prior to playing the game. It consisted of two

Table 2: Selected Questions from the Pre-Test and Post-Test Under Each Learning Objective

Learning Objective	Sample Questions
LO1. To leave players with a better appreciation for the role of the immune system and its significance in protecting the human body. (2 questions)	What do you believe is the main function of the human immune system? Which of the following is not a key function of the immune system?
LO2. To learn about the common elements involved in the process of the immune system in fighting off bacteria and viruses. (9 questions)	Which of the following are part/s of the immune system? What is the primary role of the neutrophil in the immune system? Which of the immune cells is the first to respond during bacterial infection?
LO3. To raise awareness of the implicit and explicit factors that could affect the immune system and its ability to protect the body from foreign entities. (7 questions)	What is the primary purpose of a fever in the body? When should you consider seeking medical attention for a fever? Which of the following is the best way to help our immune cells fight infection?

parts. The first part contained background questions to get a better background on the student demographics and their exposure to immune system concepts. The second part contained 18 questions about immune system concepts, which all aligned with one of the three learning outcomes in this study. The 18 questions are worth 25 points, as some of the questions involve multiple answers. All these questions are multiple choice questions, so as to facilitate more objective scoring.

The questions were designed to align with the learning objectives defined in section 3.1 to ensure that they were measuring the intended objectives the game was designed for. Table 2 shows selected questions under each learning objective.

Each subject was given 20 minutes to answer the questions. The subjects were not allowed to look at any other materials while answering the pre-test. The scores of each subject, as well as the average and median scores were used to characterize the general level of understanding of the subjects about the immune system prior to playing the game. After the pre-test, the subjects were not informed about the results, nor which items they answered correctly or incorrectly. This information was only revealed to the subjects at the end of the whole evaluation process, to minimize bias in the post-test.

4.2.2 *Play Session.* After the pre-test, each subject was asked to play the game. The play session was mostly unrestricted; the subjects can explore the game to their hearts’ content, as long as they stay in the game and play through (not necessarily beat) at least three levels by the end of the play session. The play session lasted for 60 minutes. During the play session, the subjects were

not given any additional guidance or prompts from the researchers, allowing the game itself to deliver the intended experience to the players.

4.2.3 Post-test. After the play session, the subjects were asked to answer the post-test. The post-test also consisted of two parts. The first contained a list of questions asking for feedback on the play session. This was to gain a better understanding of the students' self-reported experiences while playing the game. The second part contained the same 18 questions testing the subjects' knowledge and understanding of the immune system. Note that the students were not given the correct answers, nor which items they got right or wrong during the pre-test, so they still had no prior knowledge of the correct answers for the test. The 18 questions are still worth 25 points, as some of the questions contained multiple answers.

Each subject was also given 20 minutes to answer the questions. Similar to the pre-test, the subjects were not allowed to look at any other materials while the post-test was ongoing. The difference in pre-test and post-test scores were used as an empirical measure of learning gains.

5 RESULTS AND FINDINGS

This section discusses the results and findings of the evaluation process, along with an analysis and discussion of its implications.

5.1 Empirical Learning Gains

Figure 6 shows the scores of the test subjects for the pre-test and the post-test questions about the immune system concepts. The average score for the pre-test was 15.2857, while the average score of the post-test was 19.7143. The difference between the post-test and the pre-test average was +4826, with a paired t -test statistic of 7.801 and a p -value of 1.6959×10^{-7} , showing a statistically significant positive effect on the students' understanding of immune system concepts under a significance level of $\alpha = 0.01$.

The standard deviation of the pre-test scores was 2.8835 while the standard deviation of the post-test scores was 1.696, revealing a drop in variance. This highlights the fact that during the pre-test, the subjects' knowledge of the immune system was more variant, as some students already knew a lot about the immune system, but the others knew very little. By the end of the play session, the variance was reduced as most of the subjects who had low scores in the pre-test improved. In fact, subjects who got scores in the pre-test that are lower than the median score of 15 demonstrated the highest learning gains. Subjects below the 50th-percentile of the pre-test had an average improvement of 7.125 in the post-test. These results show promise that *Immune Survivors* was able to impact learning on its players.

5.2 Learning Objective Analysis

To check if the game was able to address its learning objectives, we looked at each group of questions under each learning objective and measured the learning gains for each. Table 3 shows the summary of this analysis. Across the three learning objectives, the evaluation showed a positive learning gain.

The learning objective that saw the most increase was LO2 (+3.0476), which pertained to the knowledge of the components of the immune system and how they function. This was followed by

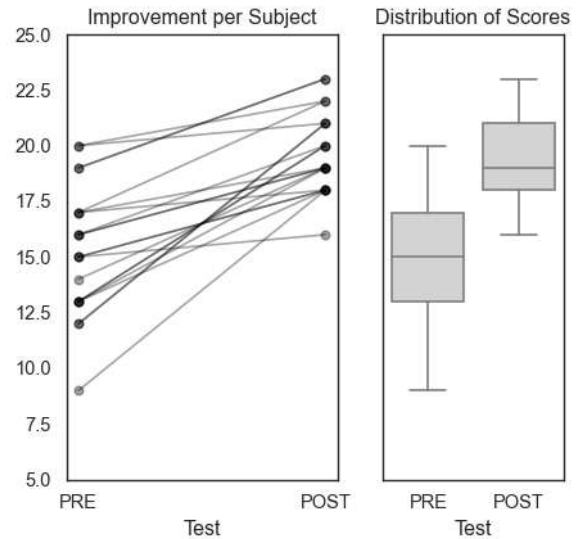


Figure 6: Improvement of scores from pre-test to post-test for each subject (left) and the distribution of scores in the pre-test and post-test (right)

LO3 (+1.2857), which pertained to the awareness of various factors that affect the immune system and their implications to it. Both these learning objectives saw a statistically significant positive difference between the post-test and pre-test scores using a paired t -test under a significance value of $\alpha = 0.01$, indicating empirical evidence of subjects' learning throughout the play session. LO1 also saw a positive learning effect (+0.0952), although it was not statistically significant ($p = 0.5471$). Two factors could have played a role to this. First, there were only two questions under this category, worth only two points. Second, more than half of the participants already scored perfectly in these two questions during the pre-test. These two factors leave not much room for improvement.

Overall, these results indicate *Immune Survivors* is able to show positive learning effects across the defined learning objectives after the hour-long playthrough session.

5.3 Item-Based Analysis

To further expand the insights from this study, we also performed item analysis on the questions with respect to the pre-test and post-test results.

Table 4 shows a list of questions with the lowest and highest number of correct answers during the pre-test. It can be seen that during the pre-test, while most subjects are generally knowledgeable about factors surrounding cough, most subjects were not aware of the components of the immune system and their roles. In addition to this, there were also a lot of incorrect answers on when one should consider seeking medical attention for a fever. These give us a general idea of which aspect students struggle with.

Comparing this with the corresponding post-test results, improvements were observed in terms of the number of wrong answers for every question. However, these improvements are not

Table 3: Observed Learning Gains for Each Learning Objective

Learning Objective	Points Possible	Pre-test Mean	Post-test Mean	Difference	<i>p</i> -value
LO1. To leave players with a better appreciation for the role of the immune system and its significance in protecting the human body. (2 questions)	2	1.4762	1.5714	+0.0952	0.5471
LO2. To learn about the common elements involved in the process of the immune system in fighting off bacteria and viruses. (9 questions)	12	6.000	9.0476	+3.0476	1.4945×10^{-7}
LO3. To raise awareness of the implicit and explicit factors that could affect the immune system and its ability to protect the body from foreign entities. (7 questions)	11	7.8095	9.0952	1.2857	0.0002

Table 4: Questions with the Highest and Lowest Number of Wrong Answers in the Pre-Test, and the Corresponding Number of Wrong Answers in the Post-Test

ID	Question	Pre-test Wrong Answers	Post-Test Wrong Answers
Highest Number of Wrong Answers in Pre-Test			
Q2	Which of the following are part/s of the immune system?	16	4
Q9	Which of the following is the primary resource that is collected when a bacteria is killed?	15	6
Q14	When should you consider seeking medical attention for a fever?	13	7
Q4	What is the primary role of the macrophage in the immune system?	12	5
Q7	What is the primary role of the B cell in the immune system?	12	9
Q3	What is the primary role of the neutrophil in the immune system?	11	6
Q5	What is the primary role of the dendritic cell in the immune system?	11	9
Lowest Number of Wrong Answers in Pre-Test			
Q15	What is the primary purpose of coughing in the body?	1	0
Q17	Which of the following is the best way to help our immune cells fight infection?	2	0
Q1	What do you believe is the main function of the human immune system?	4	0
Q16	Which of the following is a sign that a cough is becoming more serious?	6	4

equal. Q2 (Which of the following are part/s of the immune system?) showed the highest improvement, with a difference of 12. Likewise, Q14 (When should you consider seeking medical attention for a fever?), Q9 (Which of the following is the primary resource that is collected when a bacteria is killed?), Q4 (What is the primary role of the macrophage in the immune system?), and Q3 (What is the primary role of the neutrophil in the immune system?) showed decent improvement as well, with differences in the range of 6 to 9. In contrast, Q5 (What is the primary role of the dendritic cell in the immune system?), Q7 (What is the primary role of the B cell in the immune system?), and Q16 (Which of the following is a sign that a cough is becoming more serious?) did not show substantial improvements, with differences only in the range of 2 to 3. This

suggests that the game was better at teaching some concepts than others.

An interesting pattern to note is that despite all information needed to answer the questions being clearly presented in the game through information prompts, questions where larger improvements were observed tend to correspond to elements that appear more prominently in the gameplay. For example, the parts of the immune system manifest in the game as the characters that the players directly control over each level, which means that for every level, the player is constantly being reminded of the components of the immune system. The antigen is presented as the integral resource being collected by the player as the pathogens are defeated, and is also prominently displayed in the HUD. The neutrophil and the macrophage are the two first characters that appear on the character select screen, and might have contributed to the players spending more time familiarizing with those characters' abilities. On the other hand, the B-Cell is not a playable character; it instead quietly shows up when the player has collected enough antigen, without a strong or memorable prompt. The fact that it appears more subtly and sporadically in the gameplay could explain the lesser impact on learning its role in the immune system. Overall, these findings suggest that careful planning and integration of game mechanics can affect the impact of learning across different target outcomes.

5.4 Player Learning Perception, Experience, and Other Feedback

In addition to the empirical analysis conducted in the previous sections, as part of the post-test procedure, we also asked the participants to rate the game according to their perception on the following questions:

- (1) Did playing the game help you understand and better appreciate the human immune system? (1 - not at all, 5 - very much)
- (2) How would you rate your overall experience in the game? (1 - very bad, 5 - very good)

For the first question, the average response across the subjects was 4.4762. 12 participants (57.1429%) responded with a 5, 7 participants (33.3333%) responded with a 4, while the rest (9.524%) responded with a 3. These results indicate that the subjects perceived the game to be very helpful in learning about the immune system and increasing appreciation for it.

For the second question, the average response across the subjects was 4.3333. 11 participants (52.361%) responded with 5, 8 participants (38.0952%) responded with 4, while the rest (9.524%) responded with 2. Similar to the other questions, these indicate that the subject perceived their experience with playing the game to be very positive. This is important, since as with any serious game, players must first be engaged with the gameplay before the learning objectives can be delivered.

6 CONCLUSION AND FUTURE WORK

In this paper, we presented the design and evaluation of a serious game for raising immune system awareness. The game *Immune Survivors*, is a bullet heaven game aimed at facilitating three learning objectives about the immune system. Quasi-experimental results show the students who played the game demonstrated significant learning gains on immune system concepts based on their pre-test and post-test results. Furthermore, the results show that learning gains were observed across the three learning objectives, and that the largest improvements were observed on those who had a poor initial understanding of immune system concepts.

These findings highlight the potential of serious games as a tool for education. Nonetheless, there is still much work to be done to fully understand and harness this potential. While this study presents an empirical evaluation of the game, there are limitations in its design that can still be addressed in further studies. First, although learning gains were observed, it is not clear which specific elements contributed to the learning. Multiple design choices were made to facilitate learning in the game: the multiple gameplay mechanics, the learning prompts, the reinforcement of the concepts through repeated gameplay, among others. Further studies can investigate the contribution of each component to the learning gains observed. Another future direction is to investigate the effectiveness of a serious game intervention such as this one when compared to a more traditional way of learning, such as by reading a textbook or by listening to a lecture or video. This will help us better contextualize the role of serious games in the education ecosystem and help identify in which context serious games would best be used.

REFERENCES

- [1] [n. d.]. Department of Education Learning Resource Portal. <https://lrmds.deped.gov.ph>
- [2] [n. d.]. Google Trends. <https://trends.google.com/trends/>
- [3] [n. d.]. ImmuneQuest. <https://immunequest.com>
- [4] Toplumsal İmmünoloji Bilinci Anketi. 2015. Social Immunology Awareness Survey. (2015).
- [5] Ryan Sjd Baker, Sidney KD'Mello, Ma Mercedes T Rodrigo, and Arthur C Graesser. 2010. Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive-affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies* 68, 4 (2010), 223–241.
- [6] Johannes Breuer and Gary Bente. 2010. Why so serious? On the relation of serious games and learning. *Journal for computer game culture* 4 (2010), 7–24.
- [7] Antonio Calvo-Morata, Cristina Alonso-Fernández, Manuel Freire, Iván Martínez-Ortiz, and Baltasar Fernández-Manjón. 2020. Serious games to prevent and detect bullying and cyberbullying: A systematic serious games and literature review. *Computers & Education* 157 (2020), 103958.
- [8] Stéphanie Carlier, Sara Van der Paelt, Femke Ongenae, Femke De Backere, and Filip De Turck. 2020. Empowering children with ASD and their parents: Design of a serious game for anxiety and stress reduction. *Sensors* 20, 4 (2020), 966.
- [9] James Cunningham. 2023. Steam Bullet Heaven Fest Kicks Off with Firepower Overload. <https://hardcoregamer.com/articles/opinion/steam-bullet-heaven-fest-kicks-off-with-firepower-overload/477614/>
- [10] Alessandro D'Amico, Gabriele Bernardini, Ruggiero Lovreglio, and Enrico Quagliarini. 2023. A non-immersive virtual reality serious game application for flood safety training. *International journal of disaster risk reduction* 96 (2023), 103940.
- [11] Daniel Fernández Galeote and Juho Hamari. 2021. Game-based climate change engagement: analyzing the potential of entertainment and serious games. *Proceedings of the ACM on Human-Computer Interaction* 5, CHI PLAY (2021), 1–21.
- [12] Paula C Ferreira, Ana Margarida Veiga Simão, Ana Paiva, Carlos Martinho, Rui Prada, Aristides Ferreira, and Francisco Santos. 2021. Exploring empathy in cyberbullying with serious games. *Computers & Education* 166 (2021), 104155.
- [13] Ahmed Hassan, Niels Pinkwart, and Muhammad Shafi. 2021. Serious games to improve social and emotional intelligence in children with autism. *Entertainment computing* 38 (2021), 100417.
- [14] Konstantina Konstantara and Stelios Xinogalos. 2018. Cells of War: a serious game for familiarizing players with the immune system. *Simulation & Gaming* 49, 5 (2018), 567–589.
- [15] Fedwa Laamarti, Mohamad Eid, and Abdulmotaleb El Saddik. 2014. An overview of serious games. *International Journal of Computer Games Technology* 2014 (2014), 11–11.
- [16] MolecularJig. [n. d.]. Immune Attack, ScienceGameCenter. <https://www.sciencegamecenter.org/games/immune-attack>
- [17] Yemima Nestaria Naingolan, Karunia Galih Permadani, and Setiyo Prajoko. 2022. Analysis of student's misconceptions on the material of the immune system using a three-tier diagnostic test. *JPBIO (Jurnal Pendidikan Biologi)* 7, 2 (2022), 158–166.
- [18] N Nazrina M Nazry and Daniela M Romano. 2017. Mood and learning in navigation-based serious games. *Computers in Human Behavior* 73 (2017), 596–604.
- [19] Tina-Simone Neset, Lotta Andersson, Ola Uhrqvist, and Carlo Navarra. 2020. Serious gaming for climate adaptation—Assessing the potential and challenges of a digital serious game for urban climate adaptation. *Sustainability* 12, 5 (2020), 1789.
- [20] Huansheng Ning, Zhangfeng Pi, Wenxi Wang, Fadi Farha, and Shunkun Yang. 2022. A Review on Serious Games for Disaster Relief. In *2022 4th International Conference on Data Intelligence and Security (ICDIS)*. IEEE, 408–414.
- [21] Stacey L Raimondi. 2016. ImmuneQuest: Assessment of a video game as a supplement to an undergraduate immunology course. *Journal of microbiology & biology education* 17, 2 (2016), 237–245.
- [22] Republic of the Philippines, Department of Education 2016. *K to 12 Basic Education Curriculum Senior High School - Core Subject*. Republic of the Philippines, Department of Education. https://www.deped.gov.ph/wp-content/uploads/2019/01/SHS-Core_PE-and-Health-CG.pdf
- [23] Republic of the Philippines, Department of Education 2016. *K to 12 Curriculum Guide for HEALTH (Grade 1 to 10)*. Republic of the Philippines, Department of Education. https://www.deped.gov.ph/wp-content/uploads/2019/01/Health-CG_with-tagged-math-equipment.pdf
- [24] Republic of the Philippines, Department of Education 2016. *K to 12 Curriculum Guide for SCIENCE (Grade 3 to 10)*. Republic of the Philippines, Department of Education. https://www.deped.gov.ph/wp-content/uploads/2019/01/Science-CG_with-tagged-sci-equipment_revised.pdf
- [25] Luis Salvador-Ullauri, Patricia Acosta-Vargas, and Sergio Luján-Mora. 2020. Web-based serious games and accessibility: a systematic literature review. *Applied Sciences* 10, 21 (2020), 7859.
- [26] Melanie Stegman. 2014. Immune Attack players perform better on a test of cellular immunology and self confidence than their classmates who play a control video game. *Faraday discussions* 169 (2014), 403–423.

An Arduino-Based Wearable Gesture Device as a Controller for Quadcopter Drones

Clement Ong
De La Salle University
Manila, Philippines
clement.ong@dlsu.edu.ph

Jon Bryan Dela Cruz
De La Salle University
Manila, Philippines
jb_delacruz@dlsu.edu.ph

Yasmin Audrey Tutanés Datarío
De La Salle University
Manila, Philippines
yasmin_audrey_datario@dlsu.edu.ph

Juan Christian Geronimo
De La Salle University
Manila, Philippines
juan_geronimo@dlsu.edu.ph

Justin Thomas Lanting
De La Salle University
Manila, Philippines
justin_thomas_lanting@dlsu.edu.ph

Lester Andrei Opulencia
De La Salle University
Manila, Philippines
lester_andrei_opulencia@dlsu.edu.ph

ABSTRACT

Drones are typically controlled using a joystick-based controller. Due to its weight and ergonomics, this type of controller requires practice, dexterity, and attention to successfully pilot a drone. Wearable technology has surfaced as an unobtrusive mechanism to provide sensors and tracking devices for varying domains. In this paper, we describe the design and characterization of a camera-free, Arduino-based wearable gesture device that fits onto a person's hand and wrist. The device is designed to track hand movements and finger curl, which are then translated to drone flight commands. Experiments to validate the performance of a prototype drone controller were executed using an obstacle course to test several drone movements. Results show that for a total novice, the prototype controller achieved comparable navigability, and reduced collision frequency, slightly longer test course completion times but less drone crashes.

KEYWORDS

Quadcopter drones, drone controllers, wearable gesture devices, Arduino

1 INTRODUCTION

A drone or unmanned aerial vehicle (UAV) is an aircraft that can be remotely controlled. It can be classified according to its characteristics such as the number of rotors, level of autonomy, size and weight, and energy source [18]. Multirotor systems usually consist of four or more rotary wings or rotors to create lift and to keep them flying. Originally only utilized for military purposes, surveillance, and environmental monitoring [17], drones have found increasing usage in mainstream commercial and personal applications for asset inspection, traffic monitoring, aerial mapping, delivery services, photography [15]. Drones help in disaster management, through weather forecasting of storms, geographic mapping of inaccessible locations, and thermal sensor drones for search and rescue operations [11]. Use of drones for spatial social sciences [6] and in healthcare settings has also seen steady increase in recent years [14].

Drones are operated through remote ground control systems. The most common type are handheld joystick controllers that communicate through radio signals. The joysticks are used to execute motions on the horizontal plane such as forward, backward, left,

and right, and positive or negative rotations around the vertical axis and changes in the altitude of the aircraft [1]. Despite its heavy weight, joysticks are robust and allow users to maneuver the movement of the drone, thus becoming the preferred controller despite alternatives offered by smartphones. Operating joystick-based controllers, however, are complicated and unnatural for humans [9]. They require refined motor skills, practice, and dexterity to maneuver the miniature aircraft around successfully [1]. Those with hand malformations or people with hindrances in learning new things would find it challenging to use a joystick-based controller, especially for first-time users of quadcopter drones.

The utilization of more intuitive inputs such as those afforded by wearable devices as remote drone controllers may lead to ease in piloting these aircraft. Wearable technologies are pieces of clothing or accessories with embedded electronics that can be worn with ease since they can be used hands-free [7, 19]. They carry fully functioning portable computers and can execute a task by detecting gestures. Chanda [3] claims that "the output and the functioning of machines will be more intuitive if they are communicated using human gestures". Thus, transforming controllers into a more natural way of operation using wearable gesture control devices may improve their usability.

2 MOTIVATION

In this paper, we describe our work in designing an Arduino-based prototype wearable gesture device for controlling quadcopter drones - the intent is to develop a controller that allows a novice to fly the aircraft indoors as well as outdoors while reducing the inherent incidences of mishaps. Gestures detected by the wearable device are translated into a set of commands for the drone. We then evaluate the viability of the alternative controller through several quantitative and qualitative measures, including navigability, collision frequency, number of crashes, and course completion time with results comparable to the traditional joystick-based controller.

3 RELATED WORK

Previous research investigated various strategies in controlling drones, from handheld joystick controllers to touchscreen input and body suits that mimic the motions of the human body, to motion sensing cameras for tracking hand gestures. Motion-based gesture controllers have been found useful in controlling devices and robots wirelessly through sensors.

An open-source library for vision-based flight control was developed [13] recognizing eight poses with good accuracy. Although intended as a means to control a drone with a camera facing the user, the system was developed with a camera on a computer. As is typical with vision-based approaches, the recognition accuracy depends highly on environment, i.e. the amount of background clutter and lighting conditions, and the target (hand) being in the FOV at a (system-trained) distance of approximately 1 meter. It is not hard to imagine outdoor situations where such an approach may not work well, such getting glare from the sun, the hand in shadow, or low-light in the early morning or sunset.

A commercially available drone, Spark developed by DJI [1], flies off the user's hand and is controlled through hand gestures. The Leap Motion device from Gubsci and Zsedrovits [5] functions as a motion sensing camera with the palm position being used for determining if the drone should keep its position idle or to specify velocity commands. Similarly, [16] uses the Leap motion controller to pilot an AR Parrot drone, however paper does not elucidate on the gesture to control mapping, except for takeoff and landing sequences. Weaknesses of the LEAP controller include that it is limited to hand and finger gesture, a small working zone and must remain within approximately 1 meter distance of the user's hands. Moreover, because it uses infrared technology, it is susceptible performance degradation in sunlight, and thus cannot be used outdoors.

[20] describes a web-based wireless interactive control system based on hand gestures, which follows the same control commands as the Leap controller but with different hand gestures and does away with the need for any device to control the drone. However, the lack of precision of the solution restricts it to straightforward and specific activities, mainly recreational, where the user must be the focus of the quadcopter's attention. A gesture controller that uses a wearable glove technology instead of an external sensor is reported in [2]. The glove tracks regions of interest based on the trajectory projections of hands.

Motion-based gesture controllers are based on accelerometer technology to control the device wirelessly [3]. These controllers measure the acceleration of a body in space, may it be in the form of movement or vibration. They can also be designed to read body language or any action that a human performs to control several types of technologies. Mapping the motion of human body parts such as fingers, limbs, and torso, for operation and maneuver makes these devices more intuitive to use, requiring less skill and practice to maneuver the miniature devices, such as drones [4]. Still, refined motor skills, practice, and dexterity are needed to feel the ease of using such controllers [1].

Smartphones and tablets integrate well-developed, finger-based "gestures" in the form of various single and multi-finger actions, such as tap, double-tap and drag. Moreover, their touchscreens can also provide a sense of finger pressure information. Drone controllers have been developed around such devices, i.e. [8], which "abstracts the drone and camera into a single flying camera object". Meant for aerial photography, motion and attitude control of the drone/camera are fused into smartphone/tablet's attitude and touchscreen interface. This system's target application is fairly challenging and goes well beyond casual flying. Simpler finger-touch based controllers are installable applications as entry-level flight

controls for several commercially-available drones, such as the Parrot Bebop. The DJI Fly app controls several of DJI's drones, such as the Mini, Air and Mavic¹. The limitations of touchscreen-based controllers include the lack of tactile feedback and the arbitrary mapping of motions such as tap and double-tap and multi-finger / hand requirements, i.e. use of smartphones and tablets, as commercially implemented, is paired with entry-level drones.

Single-hand controllers have also been developed for drone piloting. Normally associated with controlling wheeled RC vehicles, the additional degrees of freedom for drone attitude and movement are input via an embedded accelerometer or inertial measurement unit (IMU) in the controller and joystick, such as the DJI RC Motion 2². A fair number of flight functions are mapped into this controller, with concurrent inputs required for piloting. The joystick is mapped to up/down and left/right motion; the trigger (accelerator) moves the craft forward and backward proportionately. Pitch and yaw is controlled by the attitude of the motion controller. Some motions may not necessarily be intuitive, such as tilting its body 90° to make the drone land. The price of the controller is reflective of its intended application and users.

Wearable gesture devices are often integrated with everyday clothing and accessories to serve varying purposes [19]. Fitness watches, for instance, are designed to track vital body signs such as heart rate and temperature. Gloves and exoskeletons can track body movements and have been applied to control machines and robots. These devices can be programmed to detect gestures that enables controlling other devices [2, 10, 12]

In summary, many approaches to a "better" flight controller have been developed and commercialized. Some designs are meant for indoor use only, with mobility limitations. Image processing and extraction of gestures is compute-intensive, although hardware-specific advancements have almost made this a non-issue. However, challenges remain when environmental conditions make image registration difficult. Commercially-available controllers have advanced features that the novice may not need, and can result in a steeper learning curve, with a price-point that also reflects these advanced features. In our opinion there is room for a controller catered specifically to the novice to allow the pilot to simply enjoy the experience, indoor or outdoor, while reducing the possibility of severe mishaps that could result in expensive repairs or loss of the craft.

4 PROTOTYPE DEVELOPMENT

The design of the wearable controller device to be worn as a glove is depicted in Fig. 1. It incorporates pockets to hold the sensors in place while preventing the sensors from being subjected to outside factors that could cause damage due to extreme exposure. The resulting prototype has three components: (i) the Raspberry Pi (RPI) shown in Fig. 2, (ii) the sensors shown in Fig. 3, and (iii) the Arduino shown Fig. 4.

The component holding the RPI in Fig. 2 is attached to the portion of the arm above the elbow. Using the same materials as the wristband, this part of the device is attached to a smaller band for breathability. A battery pack is added to supply power, which is

¹<https://www.dji.com/downloads/djiapp/dji-fly>

²<https://www.dji.com/rc-motion-2>

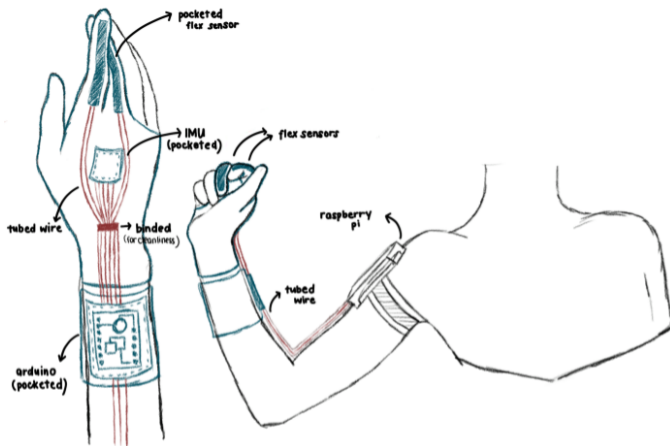


Figure 1: Preliminary design of the wearable device.

placed on the other shoulder alongside the RPI. The actual glove shown in Fig. 3 is cut open on the thumb, ring, and pinky fingers to allow for better ventilation. The glove houses several sensors: the index and middle fingers contain the flex sensors, while the IMU is located on the dorsal side of the hand. Shown in Fig. 4 is the wristband made from leather. It holds the Arduino and utilizes different clasps for its fastening to universally fit the hands of a larger population base. The wristband provides paddings between the Arduino’s shape and the wrists’ natural shape for comfort.

The wearable prototype was then programmed to connect and interact with the drone to allow the latter to be controlled with the poses indicated in Table 1. The Arduino digitizes the flex sensor voltage outputs and communicates with the IMU, then through this data it determines the pose. The pose is communicated to the RPI through the connection between the digital pins on the Arduino



Figure 3: The sensors.

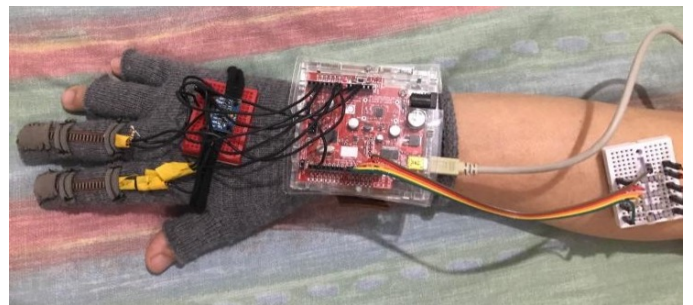


Figure 4: Arduino.



Figure 2: Raspberry PI.

and general-purpose input/output pins on the RPI. When a pose is done, it activates a combination of pins that the RPI recognizes to send the corresponding command to the drone.

5 METHOD

We conducted two types of validation: controller experiment and end-user experiment. Controller experiment involves testing the

Table 1: Hand poses programmed for the wearable drone controller.

Pose	Drone Movement
Hand tilted forward	Pitch forward
Hand tilted backward	Pitch backward
Hand tilted right	Roll right
Hand tilted left	Roll left
Straight hand pointed to the right	Rotate right
Straight hand pointed to the left	Rotate left
Fist	Decrease altitude
Index finger pointed forward	Increase altitude

drone connection with the wearable controller device to determine its recognition accuracy and response time. For each pose, a minimum of five trials were performed to check the pose recognized based on the values from the sensors. End-user experiment involves the collection of quantitative measurements and qualitative feedback. Quantitative measurements include the average response time and the accuracy of the controller in recognizing different hand gestures. Flights on an obstacle course are used to determine the usability of the controller, based on the number of times the drone crashed and the time it took for the drone to be maneuvered from one sector of the obstacle course to another. On the other hand, qualitative feedback are used to determine users' perception of the controller in terms of its intuitiveness and ease of use.

The *Obstacle Course Assessment 5* was developed to help the participants familiarize themselves with the programmed gestures and to track their progress with the specific controls. It has two parts: practice session and main obstacle course. The practice session familiarizes the participant with the eight (8) flight control functions and the response time of the drone to the controller commands. The control functions make the drone to move upward, downward, to the left, right, forward and backward, as well as rotate left and right. Once participants exhibit adequate mastery of these basic flight functions, they are allowed to proceed to the obstacle course.

The obstacle course has an oval-shaped layout with five sectors, each requiring certain combination of drone flight, shown symbolically in , along with the required flight path of the drone. Sector A is a "hall" is made of corrugated carton, forming a passage with a width of 40cm. Targeting the pitch and roll control, this sector assesses the ability to control the drone to move horizontally and vertically. Sector B has two vertical pipe structures, and is navigated by moving forward and tilting at the same time. This evaluates the ability to control the drone to move forward and rotate clockwise and counterclockwise at the same time. Sector C focuses on the combination of simultaneously moving forward while moving left and right, while Sector D determines the drone operator's ability of going through a narrow passage with "backward" (relative to the drone operator) flight pattern. Sector E is composed of three rings that measure 40cm and are placed at differing locations and heights, requiring concentration and skill since the operator must have the ability to move upward and downward while simultaneously moving left and right to "thread" the drone through rings. The time of flight per sector, the average time, and the whole obstacle courses' total time are recorded alongside the number of crashes per sector.

We invited five (5) participants to perform experiments with the traditional and our prototype wearable controller: three (3) used the traditional controller and two (2) used the prototype wearable controller. The participants had minimal to no experience with flying drones. After the experiments, we administered surveys to collect their individual evaluation of the controllers' intuitiveness and usability attributes using a 5-point Likert scale. We also conducted debriefing sessions with the two (2) participants who used the wearable controller. The questions focused on collecting their perception of comfort and overall experience in performing the drone control actions.

6 RESULTS

We present the results from our experiments with the prototype controller, end-user experiments, and users' evaluation of the prototype controller.

6.1 Prototype Controller Tests

We performed three types of tests on the controller: recognition accuracy, program response time, and controller delay.

Table 2 shows a sample of the experiments that were conducted to evaluate the accuracy of recognizing the poses, in this case. For each pose, a minimum of five trials was performed and we recorded the detected pose and the values from the sensors. All poses recorded a recognition accuracy of 100%.

The response time of the components of the wearable controller was also measured for each of the pose. Table 3 shows the sample results collected from measuring the response time in milliseconds for the *forward* drone command. The average time for the pose to be recognized by the Arduino is 3.2 ms, while the overall average response time is 4.0824 ms. However, the response time varies depending on the pose for the RPI to read the data from the Arduino and return from the command task, due to the positioning or order in the code. In this instance, the forward pose is among the bottom at the order in the code thus the time to read is greater than the time to return. On average the read time is 0.8370 ms and the return time is 0.0118 ms.

The delay for each controller was tested as well. The input command was recorded alongside the drone response. The video was then slowed down to check the delay between the input to the response. This was done five times for each controller, the results can be seen in Table 4. The difference between the traditional, 27.60 ms, to the wearable, 28.60 ms, is just 1.00 ms which is insignificant.

6.2 Practice Session

A total of eight (8) drone actions were performed by the participants using the traditional (TRAD) and the wearable (WEAR) controllers. Fig. 6 shows the average times in performing each of the actions. TRAD participants completed the actions faster by an average of 8.45s. The largest time difference is 11.92s for the *roll left* action, while the *rotate left* has the smallest average time difference of 5.68s.

In Fig. 7, it can be seen that TRAD participants were more effective in performing drone actions on the practice session, requiring fewer attempts from the participants using WEAR. An average of 1.375 attempts on each drone action was performed by TRAD participants and none exceeded 2. The WEAR participants, on the other hand, had an average of 2.5 tries.

6.3 Obstacle Course

The results in the obstacle course experiment showed improvements in the performance of WEAR. As seen in Fig. 8, WEAR participants completed the first three sectors in a much shorter time than those using TRAD. For sector 4 however, TRAD was ahead by as much as 20s, while the time difference for sector 5 was 4s, in favor of TRAD. Overall, TRAD was 12s faster than WEAR with an average course completion time of 76s.

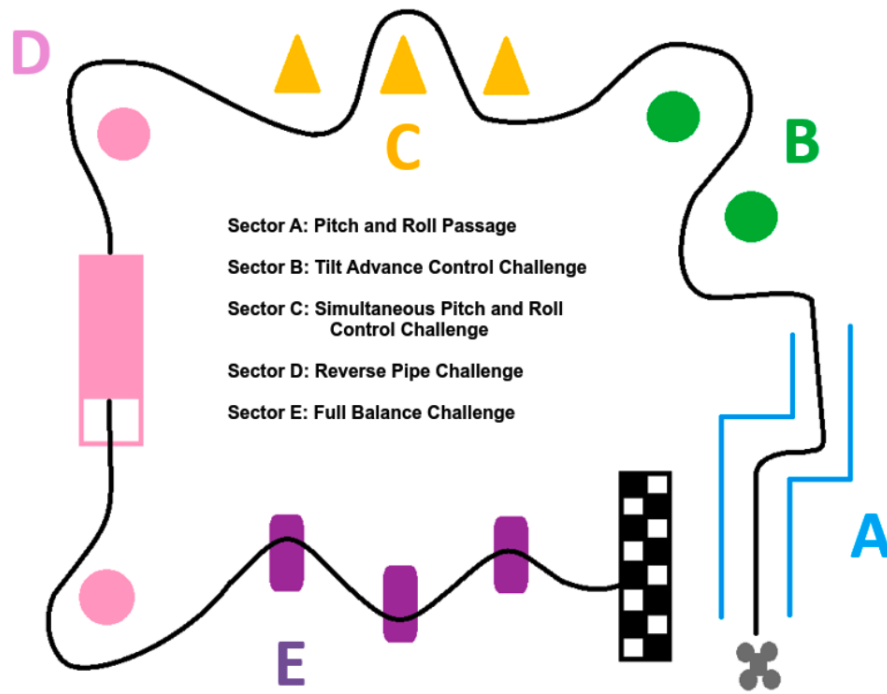


Figure 5: Design of the obstacle course.

Table 2: Sample results of gesture recognition accuracy test for the general purpose input/output connection.

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
Actual pose	Forward/Positive Pitch (Hand Tilted Forward)				
Pose recognized	Forward	Forward	Forward	Forward	Forward
Value of <i>AngleY</i> from IMU	-31	-44	-25	-50	-22
Threshold	AngleY < -20				
Deviation from the threshold	-11	-22	-5	-30	-2

Table 3: Sample results in measuring the program response test (in milliseconds) for the different poses.

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Average
Pose to Arduino recognition	3.0000	3.0000	3.0000	4.0000	3.0000	3.2000
Drone command	Forward/Positive Pitch					
Arduino to RPI drone command	0.8569	0.8263	0.8187	0.8650	0.8154	0.8370
RPI return from command	0.0119	0.0112	0.0126	0.0122	0.0110	0.0118
Total response time	3.866	3.8375	3.8313	4.8772	3.8264	4.0824

Table 4: Results from the controller delay test.

Call to Drone response	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Average
WEAR	29.00	30.00	27.00	30.00	27.00	28.60
TRAD	30.00	28.00	26.00	26.00	28.00	27.60

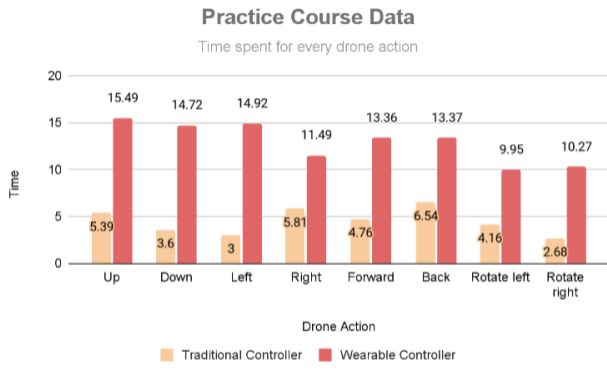


Figure 6: Average time (in seconds) spent on each drone action.

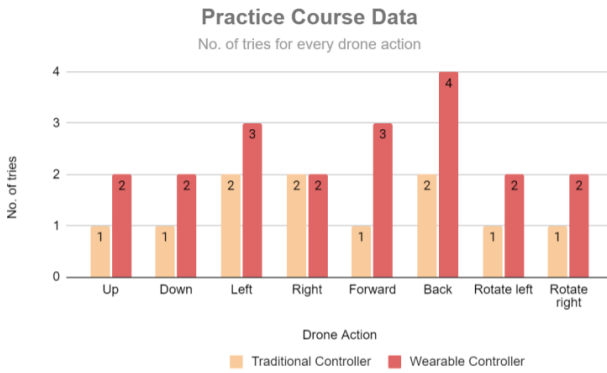


Figure 7: Average number of tries spent on each drone action.

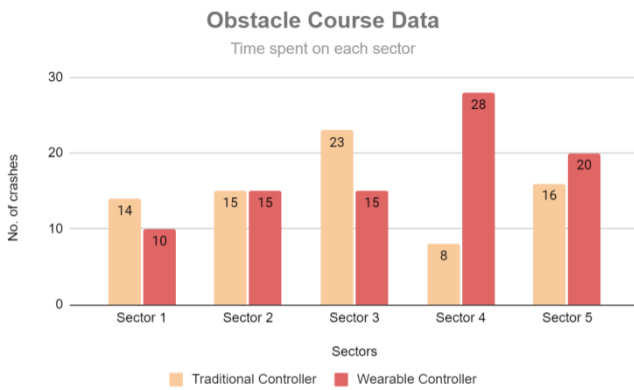


Figure 8: Time (in seconds) spent to complete each sector of the obstacle course.

We also counted the number of crashes and near hits. A *crash* occurs when the drone comes to a complete stop after a collision, while a *near hit* is counted when the drone collides with an object but continues onward. After a crash, the drone will be positioned at

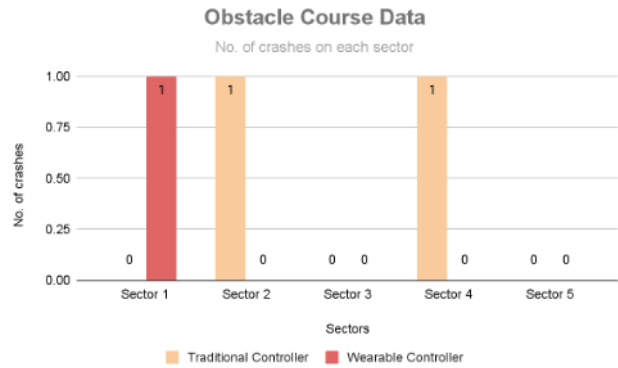


Figure 9: Number of crashes per sector.

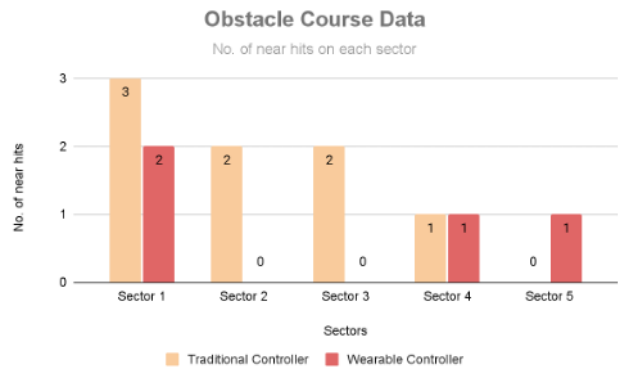


Figure 10: Number of near hits per sector.

the starting point of the current sector. In Fig. 9, it can be observed that a total of two crashes occurred using TRAD while only one crash using WEAR. In Fig. 10, WEAR had a total of four near hits in sectors 1 (2 near hits), 4 and 5 (1 each) as opposed to the eight near hits using TRAD.

6.4 User Evaluation

Fig. 11 shows the participants’ evaluation of the intuitiveness of the two controllers by rating the following items:

- (1) It was easy to learn how to use the controller.
- (2) The controller actions are simple to perform.
- (3) Performing the actions using the controller is natural for me.
- (4) I had no trouble executing the actions and gestures to command the drone.
- (5) I did not notice a delay in the drone’s response to my command.

The wearable controller received a higher rating compared to the traditional controller for all items except item 4: because the prototype controller uses a third-party WiFi radio module to communicate with the drone, WEAR participants reported encountering connection problems during the experiment. Item 3 received the most significant difference of 1.7 in the rating, underscoring the perceived intuitiveness of the wearable controller.

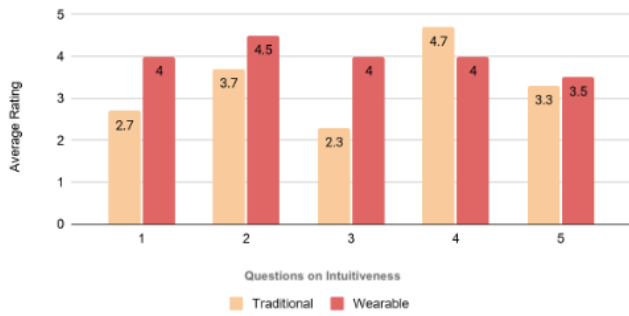


Figure 11: User perception on the intuitiveness of the controllers.

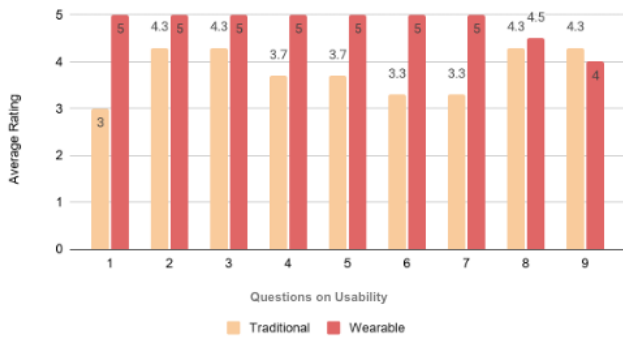


Figure 12: User perception on the usability of the controllers.

Fig. 12 shows the participants’ evaluation of the controller’s usability based on the execution of specific actions. WEAR outperformed TRAD in eight out of nine usability items, with participants successfully executing the actions that instruct the drone to maintain its position (1), go up (2), down (3), forward (4), backward (5), towards the left (6), and towards the right (7). Items 8 and 9 evaluate the ease in using the controller to command the drone to perform rotations, achieving the lowest in commanding the drone to rotate to the right (9).

7 DISCUSSION

Practice sessions took longer to master with the WEAR controller as these are calculated based on the accumulated time of the attempts to complete the required action, i.e. for the "Up" action, TRAD averaged one attempt in 5.39s while WEAR required 2 attempts to do the same. These time differences are significant primarily due to the small number of attempts that were needed to accomplish the action and do not necessarily reflect that the WEAR controller is harder to master. It might also be helpful to clarify that the TRAD controller is an app that runs on a smartphone and displays virtual buttons on the phone’s touch screen that executes the eight poses required for the practice session. It might not be unreasonable to expect that TRAD users will require only one attempt for each action.

Being able to make a drone go to a particular pose (position and orientation) in a specific flight pattern in three-dimensional space requires the pilot to mix the eight basic flight actions from the

practice session, and this is assessed by the obstacle course. Flight times for sectors 1, 2 and 3 show that WEAR is at least as fast, or faster than TRAD, which means that mixed actions are easier to accomplish on the WEAR controller. The time difference for sector 5, considered the most challenging section of the course, is 4s, where WEAR is 25% slower. This sector requires precise 3D positioning of the drone and may indicate that simultaneous up/down and left/right movements are more difficult to encode with the poses chosen for WEAR. It can be noted however that this is the same time difference for sector 1, where WEAR takes a shorter time.

The significant difference between the two controllers is in sector 4 where the drone is required to avoid a vertical pole and fly into a tunnel backward relative to the viewpoint of the pilot. This is a combination of precise 3D positioning while in flight as well as reversing the vision-to-control-requirement. Since sector 5 has shown that 3D positioning is more challenging for the WEAR pilot, the even-larger time discrepancy for sector 4 may point to a larger difficulty in reversing the eye-movement coordination. We posit that this may have been brought about by having a more intuitive (forward movement) interface.

Examining the collision rate for crash and near-hits, WEAR has half as many accidents versus TRAD, 2 vs. 4 and 4 vs. 8, respectively. This lower collision rate indicates that the what the pilot sees the drone doing versus the required flight path (specifically to avoid hitting an object) is more easily corrected using WEAR. It might be argued that the lower collision rate is due to the pilots using WEAR flying slower - however this is countered by the equal or shorter flight times for sectors 1, 2 and 3. Given this, the WEAR controller may be interpreted to be more intuitive to use.

The above assertion is substantiated by the results of the user evaluation on the controller’s intuitiveness, where 4 out of 5 questions had the WEAR controller favored. The results for each of the 8 basic flight actions reflect that the pose required to rotate the drone left, and even worse, right, may have to be recast, or perhaps the thresholds required for recognizing the pose have to be adjusted.

8 CONCLUSION

We developed a prototype wearable controller that recognizes eight hand poses, to control a quadcopter, with the objective of providing a more intuitive interface for flying the aerial vehicle. The hand pose instrumentation uses flex sensors positioned on the index and middle fingers, and an inertial measurement unit at the back of the hand. The pose is recognized by an Arduino and sent to a battery-powered Raspberry Pi which then sends the flight control command via WiFi to the drone.

Results show that users typically take two to four attempts to learn the proper pose for each flight action. The wearable controller exhibited comparable albeit slightly longer completion times against a traditional controller for an obstacle course whose sectors require various combined flight actions to accomplish, while having half the number of mishaps. Moreover, users reported a positive experience with the wearable controller, scoring higher in both intuitiveness and usability versus the traditional controller. The wearable controller however did require a longer time to accomplish individual flight actions during familiarization. Despite this,

overall we believe the controller has good potential, in line with the use-case stated in the Motivation section.

Future work will include refinement of recognizing existing poses and tuning the magnitude of the response to the pose to possibly accomplish more responsiveness without losing controllability. A larger set of participants for testing is desirable, and these can be carried out in an area that has less radio-frequency interference to avoid connectivity issues originally experienced during the development of this prototype.

REFERENCES

- [1] 2018. *UAV for Everyone: An Intuitive Control Alternative for Drone Racing Competitions*. <https://doi.org/10.1109/TISHW.2018.8559538>
- [2] Vladislav Berezhnoy, Dmitry Popov, Ilya Afanasyev, and Nikolaos Mavridis. 2018. The Hand-Gesture-Based Control Interface with Wearable Glove System. In *Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics*, Vol. 2. 448–455. <https://doi.org/10.5220/0006909304480455>
- [3] Premangshu Chanda, Pallab Kanti Mukherjee, Subrata Modak, and Asoke Nath. 2016. Gesture Controlled Robot using Arduino and Android. *International Journal of Advanced Research in Computer Science and Software Engineering* 6, 6 (June 2016).
- [4] Mesut Cicek. 2015. Wearable Technologies and Its Future Applications. *International Journal of Electrical, Electronics and Data Communication* 3, 5 (April 2015).
- [5] Gergely Gubsci and Tamas Zsedrovits. 2018. Ergonomic Quadcopter Control Using the Leap Motion Controller. In *2018 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*. 1–5. <https://doi.org/10.1109/SECONW.2018.8396348>
- [6] Ola Hall and Ibrahim Wahab. 2021. The Use of Drones in the Spatial Social Sciences. *Drones* 5, 4 (2021). <https://doi.org/10.3390/drones5040112>
- [7] Seok Chan Jeong, Sang-Hyun Kim, Ji Yeon Park, and Beomjin Choi. 2017. Domain-specific Innovativeness and New Product Adoption: A Case of Wearable Devices. *Telematics and Informatics* 34, 5 (August 2017), 399–412.
- [8] Hao Kang, Haoxiang Li, Jianming Zhang, Xin Lu, and Bedrich Benes. 2018. Fly-Cam: Multitouch Gesture Controlled Drone Gimbal Photography. *IEEE Robotics and Automation Letters* 3, 4 (2018), 3717–3724. <https://doi.org/10.1109/LRA.2018.2856271>
- [9] Guan-Chun Luh, Heng-An Lin, Yi-Hsiang Ma, and Chien Jung Yen. 2015. Intuitive Muscle-Gesture Based Robot Navigation Control using Wearable Gesture Armband. In *2015 International Conference on Machine Learning and Cybernetics*. IEEE. <https://doi.org/10.1109/ICMLC.2015.7340953>
- [10] Leonardo Meli, Davide Barcelli, Tommaso Lisini Baldi, and Domenico Prattichizzo. 2017. Hand in Air Tapping: A Wearable Input Technology to Type Wireless. (2017).
- [11] Sharifah Mastura Syed Mohd Daud, Mohd Yusmialdil Putera Mohd Yusof, Chong Chin Heo, Lay See Khoo, Mansharan Kaur Chainchel Singh, Mohd Shah Mahmood, and Hapizah Nawawi. 2022. Applications of Drone in Disaster Management: A Scoping Review. *Science Justice* 62, 1 (2022), 30–42. <https://doi.org/10.1016/j.scijus.2021.11.002>
- [12] Morais. 2016. Application of Myo Armband System to Control a Robot Interface. In *9th International Conference on Bio-inspired Systems and Signal Processing (Biosignals 2016)*. <https://doi.org/10.5220/0005706302270231>
- [13] Kathiravan Natarajan, Truong-Huy D. Nguyen, and Mutlu Mete. 2018. Hand Gesture Controlled Drones: An Open Source Library. In *22018 1st International Conference on Data Intelligence and Security*. 168–175. <https://doi.org/10.1109/ICDIS.2018.00035>
- [14] Mario Poljak and Anja Šterbenc. 2020. Use of drones in clinical microbiology and infectious diseases: current status, challenges and barriers. *Clinical Microbiology and Infection* 26, 4 (2020), 425–430. <https://doi.org/10.1016/j.cmi.2019.09.014>
- [15] James Rennie. 2022. What is a drone used for? Commercial uses and applications in various industries. <https://www.auav.com.au/articles/what-is-a-drone-used-for/>.
- [16] Ayanava Sarkar, Ketul Arvindbhai Patel, R.K. Ganesh Ram, and Geet Krishna Capoor. 2016. Gesture control of drone using a motion controller. In *2016 International Conference on Industrial Informatics and Computer Systems*. 1–5. <https://doi.org/10.1109/ICCSII.2016.7462401>
- [17] Alessia Vacca and Hiroko Onishi. 2017. Drones: military weapons, surveillance or mapping tools for environmental monitoring? The need for legal framework is required. *Transportation Research Procedia* 25 (2017), 51–62. <https://doi.org/10.1016/j.trpro.2017.05.209>
- [18] B. Vergouw, H. Nagel, G. Bondt, and B. Custers. 2016. Drone Technology: Types, Payloads, Applications, Frequency Spectrum Issues and Future Developments. *The Future of Drone Use* (2016), 21–45. https://doi.org/10.1007/978-94-6265-132-6_2
- [19] Wright and Keith. 2014. Wearable Technology: If the Tech Fits, Wear It. *Journal of Electronic Resources in Medical Libraries* 11, 4 (2014), 204–216. <https://doi.org/10.1080/15424065.2014.969051>
- [20] Zhenfei Zhao, Hao Luo, Guang-Hua Song, Zhou Chen, Zhe-Ming Lu, and Xiaofeng Wu. 2018. Web-based interactive drone control using hand gesture. *Review of Scientific Instruments* 89, 1 (2018). <https://doi.org/10.1063/1.5004004>

Halogen: A 2D Simulation Game on Dental Health Challenges in the Philippines

Julliana Zarah Cruz
Department of Information
Systems and Computer Science
Ateneo de Manila University
Quezon City, NCR, Philippines
julliana.cruz@student.ateneo.edu

Norberto Tadeo
Department of Information
Systems and Computer Science
Ateneo de Manila University
Quezon City, NCR, Philippines
norberto.tadeo@student.ateneo.edu

John Ryan Tamano
Department of Information
Systems and Computer Science
Ateneo de Manila University
Quezon City, NCR, Philippines
john.tamano@student.ateneo.edu

ABSTRACT

The thesis is about a 2D top-down educational medical simulation game from the perspective of a dentist participating in a dental mission. The main goal of the thesis is to educate players on the general practice of dentistry while raising awareness on the relevant issues related to Philippine dental health care such as inaccessible and scarce dental care, especially in far-flung areas, lack of dental health awareness, and poverty that exacerbates said inaccessibility. The researchers utilized a custom-made game to immerse the player within the medical simulation, with the intent of increasing their interest in the subject. Game testing was conducted to gather data regarding player experience and the effectiveness of the interactive and simulation game in teaching and raising awareness about dental health challenges. The results show that the game was effective in teaching dental health hygiene, operations, and concepts, but lacked a number of features that would make the game more fun for players, including minigames and indicators. This suggests that games are an effective way to teach dental health, but would need a lot more elements, features, and mechanics to make them enjoyable as well as educational for players in some areas.

CCS CONCEPTS

• Interaction Design • Life and Medical Sciences • Learning Education • Game Design

KEYWORDS

Medical simulation game, Dental hygiene, Philippines, Dental Health

1 INTRODUCTION

Dental health care in the Philippines is often overlooked but crucial for overall health and quality of life. Statistics show high rates of dental issues like caries, gum disease, and tooth decay among Filipinos [1, 2]. Accessibility to

healthcare providers and facilities is limited, especially in rural areas [3]. The presence of "faith healers" adds complexity to the healthcare landscape [4]. There have been multiple government initiatives and advocacies to bring dental health services to strengthen awareness and provide support but said efforts are unfortunately still lacking [5,6].

There is a lack of existing literature and knowledge in the capabilities of games in educating regarding dental health and hygiene, especially in the Philippine context. The study aims to determine the effectiveness of using serious games for dental health education in the Philippines. The game developed, Halogen, simulates dental procedures from a dentist's perspective during a medical mission. Key questions and objectives of the study include the effectiveness of game elements, teaching basic dentistry, highlighting industry challenges, and assessing player enjoyment.

The game was developed using Unity Game Engine 2021, targeting Windows 10 and 11 Personal Computers (PCs). The target demographic of the study are young adults aged 18-25, primarily university students, with no prior knowledge requirement. Limitations include focus on core dental procedures like check-ups and cleanings, prioritizing foundational knowledge and player engagement. Being targeted towards young adults, the effectiveness for other age groups may need further exploration.

2 REVIEW OF RELATED LITERATURE

2.1 A Video-Game-Based Oral Health Intervention in Primary Schools—A Randomised Controlled Trial

A Video-Game-Based Oral Health Intervention in Primary Schools—A Randomised Controlled Trial is a study by

Aljafari, A., Elkarmi, R., Nasser, O., Atef, A., Hosey, M. T. [8]. This study made use of two groups, one of which played a standalone dental health game while the other received no intervention. Afterward, their dietary knowledge, plaque levels, and toothbrushing habits were compared.

The results showed that there was a significant improvement in the intervention group's knowledge of dental hygiene and dietary choices, but it did not translate to improved dental practices. The authors conclude that though the game educated the students, other school interventions are necessary to cement the proper behaviors in children. They also highlighted the need to include parents in educating and building proper dental hygiene habits for children, and the need for digital, child-friendly oral health education games that could be used for online education

2.2 Current State of Serious Games in Dentistry: A Scoping Review

Current State of Serious Games in Dentistry: A Scoping Review is an article by Zaror, Mariño, and Atala-Acevedo in 2021 [7]. This article reviews research on serious games in dentistry (video games used for education or training). Researchers found two main uses: training dental professionals and teaching kids about oral health. Dental training games focused on skills and patient management, while children's games aimed to raise awareness and build healthy habits.

The studies mentioned in the review used different devices (computers, phones) and game elements (quizzes, rewards). There was no strong evidence to prove how well the games worked, but some studies showed promise. For example, children's games helped them identify unhealthy foods and improve brushing. Dental professional games may improve skills, but there is not enough evidence yet to say for sure if they are effective. The researchers concluded that more studies are needed to see if these games can truly improve oral health education.

2.3 Previous Games

The researchers also reviewed previous games that were relevant to the subject matter and/or game design. These games range from free simulation games to role-playing games. Among these games, there were medical simulation games such as Surgeon Simulator, Dentist for Children, and Toothsavers. These games were reviewed to examine the educational aspect and appeal of medical and dental simulation games. The researchers borrowed elements like level design and dental procedures from said games. Furthermore, there were also nonmedical simulation games that influenced the game such as 1.) Super Mario Bros. 3, for the game's overworld map, 2.) Pokemon Red and White, for its non-player character (NPC) interaction mechanics,

and 3.) Cooking Dash for its level design and interaction with tools.

3 METHODOLOGY

3.1 Game Design Principles

In developing the game in line with the goals of the study, multiple principles have been considered. These served as a guide not only for the development of the game, but also for emphasizing the topic in the game and assessing the effectiveness of the game in doing its intended purpose after doing participant testing.

Game Elements and Mechanics for Meaningful and Engaging Representation of Dental Operations

The main gameplay loop will have the player treating multiple patients with varying dental-related ailments with increasing complexity throughout each stage. These operations will be represented in the game through an interface that allows the player to interact with the patient's teeth using the various tools at their disposal to fully accomplish each operation. This interface will show both rows of upper and lower teeth in a neatly arranged 2D sub-screen. The player will be interacting with the patient's teeth by first selecting the applicable tool and using the said tool by selecting the specific tooth that the player will be interacting with. For example, the player can select the sickle probe and interact with the teeth to remove any initial plaque from the teeth before proceeding with a thorough cleaning.



Figure 1: Dental View and main gameplay. Here, the player is cleaning the patient's mouth, separated into sections, with a dental tool.

Effectiveness of the Game in Educating the Young Adult Audience about Dentistry and Oral Health

To measure effectiveness for the study, a pre-test and post-test was utilized. The test used is a Knowledge Assessment Questionnaire which contains seventeen (17) objective questions, aimed at assessing the knowledge of the participants before and after the testing of the game. The content of this questionnaire is also discussed in the

game, through dialogue and mechanics, and was validated by the study's Subject Matter Expert (SME). This helped determine if the game is effective in teaching dentistry and oral health. The pre-testing and post-testing scores were analyzed using descriptive statistics (mean) to measure the difference in scores before and after the game testing with the given sample size, and thus the effectiveness of the game in teaching dental health concepts.



Figure 2: Part of the intro cutscene of the game, where the player character arrives at a town where the dental mission is.

Utilizing the Game to Shed Light on Issues in the Philippine Dental Industry

The primary premise of the game focuses on the player taking on the role of a dentist undergoing a dental mission. This dental mission will take place in a remote poverty-stricken rural town where the populace does not have easy access to dental care. Throughout the game, dialogue via NPC interaction and cutscenes show the problems plaguing the town, which in turn have caused them to have difficulties in accessing basic dental care. This is how the player would experience and learn about the issues that unfold as the game progresses, more levels are completed, and more of the populace is interacted with.



Figure 3: A portion of the overworld map of the game with an NPC dialogue on-screen. The player can explore the

area and converse with its residents, learning more about them and the town

Assessing the Enjoyability Factor of the Game

To assess if measures and features added to the game were fun and interesting for the player, the researchers made use of a Game Experience Questionnaire (GEQ). This questionnaire has three variations, but for the study the core module is sufficient[9]. The GEQ does not draw on a specific psychological theory and is based on conceptual accounts of player experience. The questionnaire makes use of Likert scales (1-5) intended to measure positive affect, negative affect, frustration, flow, challenge, immersion, and competence. In particular, components such as positive affect, flow, and immersion are indicators of enjoyment. Challenge, tension, and competence can also be included in assessing enjoyment. Negative affect can be used to denote dissatisfaction or negative experiences with the game, the opposite of "fun". Descriptive statistics (mean) was implemented in the mentioned components to get their component scores. This was followed by a qualitative response into what makes the game enjoyable or otherwise for feedback and future improvements.



Figure 4: Scoring system in the game.

3.2 Game Design

The game has two main game modes. The first is the core gameplay loop involving the simulation of various dental operations which will be the primary means with which players interact with their patients during each stage. The second is the overworld traversal that is present in between stages.

3.2.1 In-stage Gameplay Mechanics and Core Gameplay Loop

During each stage, the player is tasked with giving treatment to various patients who enter the 'clinics' for free treatment. The dental operations that have been included in the game are the following: Cleaning and Tooth Extraction.

The player earns a corresponding number of points based on their performance during treatments. During stages, a line system is implemented to automatically seat each NPC in a queue-type fashion, one after the other. Additionally, each level has a short time limit to give the player a sense of urgency when completing the game.

For the player to perform operations, the player will have to obtain the required items before interacting with the patient. These items are placed on a table for players to use as they please. Items are divided into two categories, reusable and non-reusable. The two categories have separate inventories. Reusable items include the following items: Sickle Probe (also known as the Scaler), Mirror Probe, and Dental Pliers. While non-reusable items include the following items: Toothbrush, Mouthwash, and Anesthesia. Reusable items can be reused, while Non-reusable items will have to be disposed of via the trash can. Each item also has unique features which allow them to interact with the patient.

When the player has obtained the necessary items, the player can then move on to the diagnosis and treatment. To begin, the player must interact with the seated patient in order to open what the developers call the 'Dental View'. This interface is the primary medium that allows the players to directly access the patient's teeth and interact with the teeth using whatever tools are necessary. When viewing the teeth from the dental view, the player will first have to choose which section to view, the left, right, and middle sections of the patient's teeth.



Figure 5: The main gameplay, with the complete UI

The separation of the teeth into sections allows the players to view the teeth neatly and accurately. It also emphasizes the various ailments that each patient may be suffering from. In the Dental View, the player is able to perform a number of operations provided they have the necessary tools in their inventory. This includes, cleaning or tooth extraction.

Cleaning is accomplished with the toothbrush and sickle probe. It involves using the toothbrush on the patient's yellowed teeth and using the sickle probe to fully clean it afterward.

Tooth Extraction is done with the dental pliers, anesthesia, and mouthwash. The player must identify the tooth or teeth to be extracted (these are teeth that are blackened), use

mouthwash before extraction, apply anesthesia in the gums near said tooth, and finally, extract the tooth with the dental pliers. As a final step, the player must use mouthwash again after extraction.

When all required operations are done (full cleaning and all decayed teeth extracted), the player must select the finish patient button, after which the player will gain a large number of points and the next patient will be automatically seated.

3.2.2 Overworld Traversal and Stage Select

After clicking "Play" from the main menu, the players will be greeted with a short introduction cutscene and will be transitioned into the overworld stage. The overworld acts as a hub and stage select area. Here, the player can walk around, interact with NPCs, and explore the small town where the dental mission is being conducted. Movement is limited to a small invisible border around the town. Interaction is limited to accessing a stage and initiating dialogue with the NPCs.



Figure 6: Complete overworld play area

The gold tooth icons will be where the player accesses the levels. The player will be restricted to accessing the tutorial level at the start and will unlock another stage after completing the tutorial. The current game has a total of 2 levels, the tutorial and the first stage.

3.2.3 Aesthetics and Visual Design

The game takes place in a rural setting, immersing players in a remote town far from urban areas. The visuals depict a lack of access to many conveniences, implying the potential consequences of neglecting dental health in such an environment. This design choice aims to engage players with the realities of limited resources.



Figure 7: Game Concept art and Title Screen background

The assets in particular were made with an animated sprite and pixel art tool, Aseprite. These assets include sprites, backgrounds, cutscenes, etc. The researchers envisioned the game’s aesthetic to be reminiscent of a rural scene in the Philippines. As such the assets chosen were made to further the immersive experience of the player.

3.3 Game Development

The game is being developed in Unity Game Engine. The initial experience of the researchers with Unity prior to this thesis project was rather limited. The primary programming language of choice for this project is C# as that is what Unity uses for its scripts.

Large portions of the programming were done with modularity in mind. A lot of the code has been rebuilt and remade to be more dynamic in that its reusability across differing scenes in the game is much more friendly.

4 RESULTS

The game underwent two types of testing: alpha-testing and beta-testing. Alpha-testing made use of an earlier version of the game, which did not have NPC dialogue except for a short tutorial for the controls, three patients to clean, and a lack of a timer feature. Feedbacks from alpha-testing would be implemented and improved in a later version, which was then used for beta-testing.

4.1 Alpha-Testing

Alpha testing was done supervised and with the participants consisting of two (2) interns from the University of the Philippines Integrated School (UPIS) who were undertaking their internship in Ateneo de Manila University, and the Subject Matter Expert (SME) of the researchers. The former group were not specifically chosen by the researchers, nor were their identities disclosed before testing, and were simply chosen for their availability for testing as part of their internship. Participants were tasked to fill out a form asking

for their comments regarding aspects of the game such as concept and game premise, interface design and story, gameplay, bugs encountered, and other additional comments they had.

4.1.1 Key Takeaways from Alpha-testing

User	UPIS Interns	SME
Strengths	<ul style="list-style-type: none"> - Interesting concept and premise - Easy to understand gameplay, suitable for all ages - Original mix of mechanics (Cooking Mama & Surgeon Simulator) - Aesthetically pleasing 2D pixel style - Entertaining, enlightening, and good cutscenes - Simple and easy-to-understand controls - Enjoyable exploration and character interaction 	<ul style="list-style-type: none"> - Good representation of dental operations - Aesthetics well-represent the context - Educational merit in teaching dental concepts
Weakness	<ul style="list-style-type: none"> - Average user interface - Lack of pointers/markers for completing goals - Missing in-game instructions and tutorials - No competitive elements or minigames 	<ul style="list-style-type: none"> - Inaccurate order of operations (procedures) - Gameplay doesn't reflect real-world practices (toothbrush usage) - Skepticism about effectiveness for teaching (limited experience with simulation learning)

User	UPIS Interns	SME
Areas for Improvement	- Enhance user interface with pointers/markers	- Correct order of dental operations
	- Include in-game instructions and tutorials	- Refine gameplay practices to reflect real-world dentistry
	- Add competitive elements and minigames	- Variety and replay value

Table 4.1 Summary of Feedback from Alpha-Testing

Overall, participants found that the game successfully raised awareness about dental hygiene and taught basic dental concepts, such as procedures and tools. However, opinions were mixed on whether the game effectively conveyed the dental health situation and context in the Philippines. While the SME believed the game could effectively teach this narrative, UPIS interns felt the storytelling aspect needed improvement. These insights were taken into account for the development of a later version of the game for Beta-testing.

4.2 Beta-Testing

Beta-testing was conducted with five (5) participants, young adults between the ages of 18 to 25 years old. Participants were recruited through face-to-face communication, and were chosen randomly. Testing was conducted with the participants one-by-one and with the presence of the researchers. The average duration of testing was 30 minutes. Unlike in alpha-testing, beta-testing made use of the pre-test and post-test Knowledge Assessment Questionnaire, as well as the Game Experience Questionnaire for feedback on the game to consider specific aspects of the game that were effective.

4.2.1 Key Takeaways from Beta-Testing

Results from the Knowledge Assessment Questionnaire in both pre-test and post-test show that the game had mixed results in terms of teaching dental health concepts, effects, and issues. 60% (3) of the participants had better scores after playing the game, but 20% (1) of the participants had a worse score after playing the game, and another 20% (1) of the participants had the same score. The summary of the results can be seen below:

Knowledge Assessment Questionnaire Results

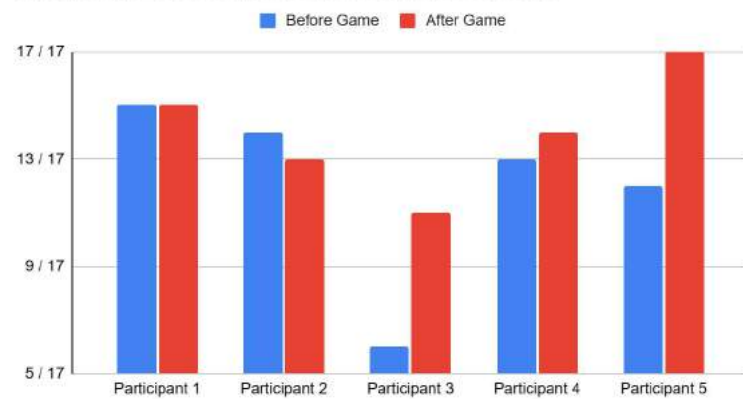


Figure 8: Knowledge Assessment Questionnaire Results Before and After Game Testing

4.3 Game Experience Questionnaire Results

Results from the Game Experience Questionnaire (GEQ) indicate high consistently high scores in components such as Competence, Immersion, Flow, and Positive Affect. The scale used for the GEQ is as follows: Not at all (1), Slightly (2), Moderately (3), Fairly (4), Extremely (5):

The average Competence component score stands at 4.2, suggesting that participants felt reasonably proficient in the game and subject matter both during and after gameplay. As for immersion, the average component score is 3.9, indicating a moderate level of engagement with the game's art and world, though not reaching a high level of immersion. Regarding Flow, the component score of 3.5 suggests that participants were moderately engaged and focused on the gameplay experience.

User	Competence	Immersion	Flow	Positive Affect
Participant 1	4.4	3.5	2.4	4.0
Participant 2	4.0	3.5	5.0	4.6
Participant 3	3.2	4.0	2.8	3.8
Participant 4	4.4	3.5	3.8	4.2
Participant 5	5.0	4.8	3.4	4.8
Average	4.2	3.9	3.5	4.3

Table 4.2 GEQ Results for Competence, Immersion, Flow, and Positive Affect

And finally, for Positive Affect, the overall component score was 4.3, indicating that the participants felt fairly content and happy with the game, and had enjoyed it.

<i>User</i>	<i>Annoyance</i>	<i>Challenge</i>	<i>Negative Affect</i>
Participant 1	2.0	3.6	1.0
Participant 2	1.0	3.2	1.0
Participant 3	2.3	2.4	2.0
Participant 4	2.0	2.2	1.8
Participant 5	1.3	1.4	1.5
Average	1.7	2.6	1.5

Table 4.3 GEQ Results for Annoyance, Challenge, and Negative Affect

The Tension or Annoyance component yielded an overall score of 1.7, suggesting that participants experienced minimal irritation or frustration during gameplay. Regarding the Challenge component, it received an overall score of 2.6, signifying that participants found the game slightly challenging above the average level, yet not to the extent of being considered moderately challenging. Lastly, the Negative Affect component garnered an overall score of 1.5, indicating that participants did not report feelings of boredom, fatigue, or other negative emotions while engaging with the game.

This is supported by the participants' qualitative feedback on the game. Many noted that the game mechanics allowed them to understand dental operations, knowledge about infections, promotion of better hygiene, along with the engaging gameplay loop, art utilized, NPCs and dialogue, as well as the implementation of the scoring system for competitiveness and measurement of learning and thus creating an enjoyable experience.

Overall, the component scores and qualitative feedback indicate that the participants thoroughly enjoyed the game, its world, and its content, and that the game made them feel that they were fairly competent, and only encountered minimal annoyance, challenges, and negative emotions. These results suggest a positive reception and engagement with the game.

5 CONCLUSION

After conducting alpha-testing and beta-testing, considerable conclusions, suggestions, and recommendations can be derived at this point in time to

improve future works and research pertaining to the use of games in teaching dental health and hygiene. The results of the Knowledge Assessment Questionnaire is able to show that Halogen is able to teach, however there needs to be further studies and improvements in order to conclude this as fact, as the study had only made use of 5 participants, and only 3 of them displayed learning. The results of the GEQ, however, has proven that the use of serious games to teach dental health concepts and issues was fun and enjoyable for the participants, and thus they were able to engage with the material. These findings underscore the potential of serious games in effectively engaging learners in dental health education, while also highlighting the need for further research and development to ensure broader efficacy and impact.

5.1 Future Work, Suggestions, and Areas of Improvement

After the testing phases,, the researchers have determined that while the thesis project has a lot of merit already in its potential as a learning tool, there is still a lot of future work that can be done to further improve it.

One of the research goals of the study was to determine, identify, and implement elements and mechanics that would make the game engaging and enjoyable. While to an extent this has been fulfilled, it should be noted that this term is highly subjective, and the researchers believe that there is a lot to be desired in this aspect of the simulation game. New challenges and new obstacles for the player to overcome would help keep the player's attention and help add replay value to the game. The current problem that the researchers believe needs to be addressed is the lack of challenge. As suggested by the SME of the study, adding a third operation will help add replay value to make the game more interesting. A suggestion by the participants on the other hand was to add a guessing game of sorts before the operation proper where the player must inspect the teeth of the patient and determine the correct choice among a set of options as to what the correct operation needs to be.

Another suggestion was to incorporate a more interactive tutorial. In this way the player is more engaged as they learn the game's mechanics by using a more visual and interactive tutorial with minimal dialogue. Adding more operations, different gameplay styles, and enhancing the tutorial stage would improve the variety of activities in the game, thus improving player retention and enjoyment. Along with that, another suggestion was to make elements clear and visible to the users, adding more visual response to actions to determine that they register or that they happened, like when anesthesia has been applied, or if certain items are unavailable and are already used, like the

disposable items anesthesia, mouthwash, and toothbrush. Halogen also had indicators for when a tooth was clean, however players suggested to make it more distinct, as there were instances where the participants were left thinking they had correctly finished cleaning, but were unable to use the explorer to clean the teeth fully. A suggestion was to implement particle effects to make the game be closer to actual brushing and to have visual indicators.

Another element that needs future work is the accuracy of dental operations. The researchers believe that there is room for improvement in this regard. Some reworking of the internal systems may be required to help make the game more accurate to real-life standards. However, it should be noted that since the game was not developed to aid the education of medical practitioners, it does not have to be accurate to the finest detail, as long as it represents the proper flow of the dentistry operations alongside teaching oral hygiene.

As this study only had 5 participants, another area of improvement would be to increase the demographics in this field of study. Considering that this study brought mixed conclusions to the effectiveness of serious games in teaching dental health, future studies should include a greater number of participants to further shed light into this body of knowledge.

Other areas of improvement include exploring the capabilities of other game development platforms for developing games primarily for teaching. Unreal engine is a powerful game engine known for stunning visuals and advanced features such as real-time ray tracing and physically based rendering. Godot is also another free and open-source game engine that is becoming increasingly popular for its ease of learning and use. As such, it is a great choice as it is a powerful and flexible engine. GameMaker Studio 2 is another popular game engine designed specifically for 2D Game Development, that makes use of a scripting language instead of coding. This study made use of the Unity game engine primarily out of familiarity, but exploring other game engines specializing on desired results could optimize performance, as well as streamline projects leading to a more polished gameplay experience.

Minor improvements could include the addition of language options to the game. The dialogue of the game is primarily in English. English is considered a secondary language in the Philippines. However, if the goal of the study is to teach and bring awareness to dental health and issues in the Philippines, then it would be beneficial to also include the primary and native language, Filipino, as the primary

language option for the dialogue in the game. This could help emphasize the teachings and points made regarding dental health to the demographics that it matters to the most: those who live in far-flung areas.

5.2 Concluding Remarks/Final Points

In conclusion, the results suggest that the use of serious games in the education of medical concepts may prove to be effective. To be more specific, the results have shown that participants, after playing the game, learned information regarding dental health, processes, operations, and tools. While it is lacking in areas related to dental hygiene in the Philippine context and visual cues in mechanics, the game was successful in being fun, thus fulfilling an objective of this study, yet only partially successful in teaching as shown by the mixed results in the knowledge assessment indicating that further studies need to be conducted in methods that will be better at these areas. Future development should address these areas to enhance engagement and ensure the game effectively tackles dental hygiene in the Philippine context, and could involve incorporating Philippine-specific information and enhancing the visual clarity of the game mechanics as well as investigate the long-term knowledge retention and behavioral changes for effectiveness.

REFERENCES

- [1] DEPARTMENT OF HEALTH. Dental Health Program: Department of Health Website. Department of Health website. URL=<https://doh.gov.ph/dental-health-program>
- [2] Philippine Dental Sector. 2022. International Trade Administration. URL=<https://www.trade.gov/market-intelligence/philippine-dental-sector>
- [3] Department of Health. National eHealth Context (Overview) URL=<http://ehealth.doh.gov.ph/index.php/transparency/overview>
- [4] COLLADO, Z. 2019. Challenges in public health facilities and services: evidence from a geographically isolated and disadvantaged area in the Philippines. *Journal of Global Health Reports*. URL=<https://www.joghr.org/article/11962-challenges-in-public-health-facilities-and-services-evidence-from-a-geographically-isolated-and-disadvantaged-area-in-the-philippines>
- [5] CERVANTES, F. 2019. Senator pushes for creation of dental units in rural areas. URL=<https://www.pna.gov.ph/articles/1084138>
- [6] TUBADEZA, C. 2023. Abra Dental Chapter brings dental services to remote barangay.
- [7] ZAROR, C., MARINO, R., & ATALA-ACEVEDO, C. (2021, March 29). Current State of Serious Games in Dentistry: A Scoping Review. *Games for Health Journal*, 10(2), 95-108. URL=<https://doi.org/10.1089/g4h.2020.0042>
- [8] ALJAFARI, A., ELKARMI, R., NASSER, O., ATEF, A., HOSEY, M. T. 2022. A video-game-based oral health intervention in primary schools-a randomised controlled trial. *Dentistry journal*. URL=<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9139898/>
- [9] IJSSELSTEIN, W. A., DE KORT, Y. A. W., & POELS, K. (2013). The Game Experience Questionnaire. Technische Universiteit Eindhoven. URL = https://pure.tue.nl/ws/portalfiles/portal/21666907/Game_Experience_Questionnaire_English.pdf

Formal Verification of Shortest Job First Scheduling Algorithm in Coq

Jian Lawrence Luteria
University of the Philippines
jgluteria@up.edu.ph

Earl Wilbur Nogra
University of the Philippines
eanogra@up.edu.ph

Andrei Tiangco
University of the Philippines
altiangco@up.edu.ph

Alfonso B. Labao
University of the Philippines
ablalabao@up.edu.ph

Henry N. Adorna
University of the Philippines
hnadorna@up.edu.ph

ABSTRACT

Schedulers play a critical role in operating systems, particularly in real-time systems where timing guarantees are vital. This paper formally verifies the non-preemptive Shortest Job First (SJF) scheduling algorithm using the Coq proof assistant. Our focus is on two key properties: permutation (ensuring each job appears once in the schedule) and sortedness (scheduling jobs with the shortest burst times first). Through Coq verification, we establish the SJF algorithm's reliability. While this work explores the non-preemptive SJF variant, our methodology can extend to more complex scheduling algorithms and real-world scenarios with dynamic job arrivals and preemption capabilities.

KEYWORDS

Formal verification, correctness, scheduling, operating system

1 INTRODUCTION

An operating system (OS) serves as a fundamental software layer that manages computer hardware and provides a platform for various applications. Central to the functioning of an OS is the scheduler, which determines the allocation of the CPU to different processes [12]. Scheduling is the process of efficiently managing and prioritizing tasks in a computing system.

Different scheduling algorithms, such as First-In-First-Out (FIFO) and Round Robin, contribute to this process by implementing distinct strategies for task execution. FIFO follows a first-come, first-served approach, while Round Robin ensures fair CPU time distribution by allocating fixed time slices to each process in a circular manner.

Another common algorithm used for scheduling is the Shortest Job First (SJF) algorithm, which selects the waiting process with the smallest execution time to execute next [11]. It is particularly suitable for jobs with predetermined run times [10]. In this approach, each process in the ready queue is executed based on its shortest burst time, minimizing waiting times for individual processes and consequently reducing the overall average waiting time.

As operating systems evolve into complex programs, ensuring their adherence to intended functionality becomes crucial [4]. Kumar et al. [7] highlighted the complexity of safety-critical systems such as nuclear power plant control software, emphasizing the necessity to model these systems prior to implementation. Similarly, Juvva K. [5] emphasized the crucial nature of timely processing in real-time systems such as air traffic control, stating that “a missed

deadline in hard real-time systems is catastrophic”. These case studies reinforce the need of a verified process scheduling algorithm to minimize the risk of failure and enhance system safety.

The method of formal verification can be used to provide high assurance to critical software [4, 6]. It involves rigorously proving that a system accords with its formal specifications or properties through the use of automated tools. The proof assistant Coq provides a formal language for expressing logical statements and a mechanism for constructing machine-checked proofs [13]. It ensures elimination of human errors and comprehensive consideration of all pertinent cases.

Formal verification has ensured reliability across the domain of scheduling and algorithm design [3, 8]. A recent formal correctness proof in Coq focused on verifying a similar scheduling algorithm via the Earliest Deadline First (EDF) scheduler working in real-time systems [14]. The proof methodology is organized into distinct modules, addressing aspects such as the *election function*, which governs the selection of the next task to be scheduled based on certain criteria; the driver or *backend* code, responsible for the main execution of the scheduling algorithm; and *assumptions*, which encompass the foundational conditions crucial for the correct functioning of the scheduler. The authors arguably become the first to showcase a formally proved correct implementation of an EDF scheduler, a methodology they assert to be “general enough to be applied to other schedulers or other types of system code”.

In another case study, Bedarkar et al. [1] verified the response-time analysis of First-In First-Out (FIFO) scheduling in Coq, showcasing the feasibility of formal verification in the domain. Their findings emphasize the importance of domain-specific libraries such as PROSA. This reinforces the idea that formal verification is not only achievable but efficient for real-time scheduling theory.

In recent years, formal verification has gained prominence as a reliable method for validating system correctness. This paper addresses the challenge of ensuring correctness in systems and critical software by leveraging formal verification techniques. The specific focus of this paper is to formally verify the SJF algorithm using the functional programming language in Coq.

1.1 Contributions

This study addresses the growing need for assurance in operating system scheduling, particularly for safety-critical applications where timing guarantees are essential. By employing the Coq proof assistant, we present a formal verification of a non-preemptive SJF scheduling algorithm. This formal approach mathematically

verifies the algorithm’s correctness. This work thus contributes to the field of reliable scheduling by establishing a methodology for the SJF algorithm’s behavior, offering valuable insights for the development and verification of more complex scheduling policies.

2 BACKGROUND

This study explores the formal verification of the SJF scheduling algorithm, specifically its correctness using Coq. The concept of correctness pertains to the subproperties of permutation and sort-*edness*.

- (1) *permutation*: the exact correspondence of jobs in the output set with those in the input set
- (2) *sorted*: the actual intended behavior of the algorithm, i.e. the ascending order of burst times in the scheduled jobs

Real-world scheduling implementations must contend with factors such as context switching overhead, which incurs additional time for the processor to switch between tasks [9]. However, to simplify the analysis and focus on the core principles of SJF, we assume negligible time for context switches. Additionally, the verification of the algorithm assumes that all incoming tasks arrive at time $t = 0$ with potentially same burst times. Lastly, this verification process is conducted in an offline mode, where process data is provided in predefined batches rather than in real-time.

3 METHODOLOGY

We detail the formalization of the SJF algorithm by defining the algorithmic steps and properties using Coq’s syntax with natural language correspondences.

An informal representation of the SJF scheduling algorithm serves as the preliminary step in our methodology. The use of a pseudocode offers a high-level, human-readable description of the algorithm’s logic, enabling clear comprehension of its operational steps [2]. This pseudocode serves as a blueprint for the subsequent translation of the algorithm into the Coq proof assistant as shown in Figure 1. Through this translation process, we are able to encode the algorithm’s logic into Coq.

3.1 Formalization of SJF Algorithm

Definition of Jobs. We begin by defining a *job* data type that encapsulates the essential properties of a job, including its ID and burst time. Hereafter, the list of jobs coming in and out of the scheduler is referred to as *joblist* which denotes a sequence of job elements, building upon Coq’s built-in list structure.

```
Inductive job : Type :=
  | taskj (id : nat) (burst_time : nat).
```

```
Definition joblist := list job.
```

SJF Scheduler. We define the SJF scheduler function, which takes a list of jobs as input and produces a sorted list according to ascending burst times. This function employs the insertion sort subroutine, ensuring that shorter jobs are prioritized over longer jobs.

```
Fixpoint insert_job (j : job) (l : joblist) :=
  match l with
  | nil => j :: nil
  | h :: t => if leb (get_burst j) (get_burst h)
  then j :: h :: t else h :: insert_job j t
```

```
end.
```

```
Fixpoint sjf (lst : joblist) : joblist :=
  match lst with
  | nil => nil
  | h :: t => insert_job h (sjf t)
  end.
```

3.2 Correctness Proof of SJF Algorithm

The correctness proof of the algorithm is subdivided into proving its permutation and sorted properties.

Permutation Property. We prove that applying the SJF scheduler function to a list of jobs results in a permutation of the original list. This property ensures that the scheduler produces a valid schedule without discarding or duplicating jobs.

LEMMA 3.1. *For all x and l , the permutation of inserting x into list l is equivalent to x followed by the result of inserting x into the rest of the list.*

PROOF. We prove by induction.

Base case: If l is empty, the permutation of inserting x into an empty list is x followed by the empty list itself, which is reflexive.

Inductive case:

- If the burst time of x is less than or equal to the burst time of the first element of l , then inserting x into l results in the permutation being the same as the original list.
- Otherwise, if the burst time of x is greater than the burst time of the first element a of l , then the permutation of inserting x into l is equivalent to swapping x with a , then inserting x into the rest of the list.

□

```
Lemma insert_perm: forall x l, Permutation
(x :: l) (insert_job x l).
```

```
Proof.
```

```
  induction l.
```

```
  - simpl. apply Permutation_refl.
```

```
  - simpl. bdestruct (leb (get_burst x)
(get_burst a)).
```

```
    ++ apply Permutation_refl.
```

```
    ++ assert (R: Permutation (x :: a :: l)
(a :: x :: l)).
```

```
      { apply perm_swap. }
```

```
      rewrite -> R. apply perm_skip. apply IHl.
```

```
    Qed.
```

With the aid of the Permutation module from Coq’s standard library, the lemma `insert_perm` is provided to inductively show that inserting a job into a list preserves the permutation of the list elements. In the base case, when the list is empty, the lemma trivially holds as inserting a job into an empty list results in a singleton list, which is permutation-equivalent to itself. For the inductive step, when the list contains elements, the lemma utilizes the `bdestruct` tactic to analyze whether the burst time of the job to be inserted is less than or equal to the burst time of the first element in the list. If this condition holds, the lemma again trivially holds as the insertion

```

// Initialize variables
ready_queue <- queue for processes ready
to execute
current_process <- None
// Sort the ready queue from its shortest
burst time to longest burst time
through
// insertion sort
for j <- 2 to length of ready queue do
  key <- ready_queue[j]
  i <- j - 1
  while i > 0 and ready_queue[i] > key
  do
    ready_queue[i + 1] <-
      ready_queue[i]
    i <- i - 1
  end while
  ready_queue[i + 1] <- key
end for

```

```

Fixpoint insert_job (j : job) (l : joblist) :=
  match l with
  | nil => j :: nil
  | h :: t => if leb (get_burst j) (get_burst
    h) then j :: h :: t else h ::
    insert_job j t
  end.

Fixpoint sjf (lst : joblist) : joblist :=
  match lst with
  | nil => nil
  | h :: t => insert_job h (sjf t)
  end.

```

Figure 1: Side by side comparison between pseudocode and Coq implementation of SJF algorithm

operation does not alter the relative order of elements. However, if the condition is false, the lemma utilizes the `perm_swap` lemma to swap the positions of the job to be inserted and the first element in the list, establishing a permutation between the list before and after the insertion operation. Finally, the lemma recursively applies the induction hypothesis to the remaining elements of the list, ensuring that the permutation property is preserved throughout the insertion process.

LEMMA 3.2. *For any job a , and joblists x and y , if x is a permutation of y , then inserting a into x yields the same permutation as inserting a into y .*

PROOF. We prove by cases.

- (1) If the lists x and y are the same, inserting a into both lists results in the same list.
- (2) If x is a permutation of x_0 and x_0 is a permutation of y , then inserting a into x is equivalent to inserting a into x_0 twice.
- (3) If x is obtained from x_0 by swapping two elements and x_0 is a permutation of y , then inserting a into x is equivalent to swapping a with those two elements in y , followed by inserting a into the rest of the list.
- (4) If x is obtained from x_0 by swapping two elements and x_0 is obtained from y by swapping two different elements, then inserting a into x is equivalent to inserting a into x_0 followed by applying the same swaps.

Thus, in all cases, inserting a into x yields the same permutation as inserting a into y . \square

```

Lemma lem2 : forall a x y, Permutation x y ->
  Permutation (insert_job a x) (insert_job a y).
Proof.
  intros. simpl. inversion H.
  - apply Permutation_refl.

```

```

- simpl. bdestruct (leb (get_burst a)
(get_burst x0)).
+ apply perm_skip. apply perm_skip. apply H0.
+ apply perm_skip. rewrite <- insert_perm.
  rewrite <- insert_perm. apply perm_skip.
  apply H0.
- rewrite <- insert_perm. rewrite <-
insert_perm.
rewrite -> H0. rewrite -> H1. apply perm_skip.
apply H.
- rewrite <- insert_perm. rewrite <-
insert_perm.
apply perm_skip. apply H. Qed.

```

Another lemma, `lem2`, establishes that inserting a job into two permutation-equivalent lists results in lists that are also permutation-equivalent. The proof begins by applying induction on the permutation between the two lists. In the base case, where the permutation is reflexivity, the lemma trivially holds as inserting a job into lists that are already permutation-equivalent yields lists that remain permutation-equivalent. For the inductive steps corresponding to transitivity and symmetry of permutations, the lemma employs the `insert_perm` lemma to ensure that inserting a job into the corresponding lists preserves their permutation equivalence. Additionally, when the permutation involves swapping adjacent elements, the lemma utilizes the `perm_skip` tactic to insert the job while maintaining the permutation property.

THEOREM 3.3. *For any joblist l , l is a permutation of $(sjf\ l)$.*

PROOF. We prove by induction.

Base Case: If the list l is empty, then l is a permutation of itself, which is reflexive.

Inductive Case: If l is obtained by inserting an element into the list l_0 and l_0 is a permutation of $(sjf l_0)$, then l is also a permutation of $(sjf l)$.

By induction hypothesis, l_0 is a permutation of $(sjf l_0)$. Then, by Lemma lem2, inserting an element into l_0 yields the same permutation as inserting it into $(sjf l_0)$. Hence, l is a permutation of $(sjf l)$. Thus, for all cases, l is a permutation of $(sjf l)$. \square

```
Theorem sjf_perm: forall l, Permutation l (sjf l).
Proof.
  intros. induction l.
  - simpl. apply Permutation_refl.
  - simpl. rewrite -> insert_perm. apply lem2.
    apply IHl. Qed.
```

Now we prove the permutation property through induction: in the base case, it demonstrates that applying the SJF scheduler to an empty list results in the identity permutation. In the inductive step, the proof utilizes the `insert_perm` lemma and recursively applies the induction hypothesis to establish the preservation of permutation when scheduling non-empty lists.

Sorted Property. We prove that the sorted list produced by the SJF scheduler satisfies the SJF criteria, wherein jobs with shorter burst times precede those with longer burst times.

```
Inductive sorted : joblist -> Prop :=
|sorted_nil : sorted [ ]
|sorted_1 : forall x, sorted [x]
|sorted_cons : forall x y l, get_burst x <=
  get_burst y -> sorted (y::l) -> sorted (x::y::l).
```

First we define the concept of a sorted list of jobs within Coq, inductively defined by the `sorted` predicate. This predicate asserts that a list of jobs is sorted if it satisfies one of three conditions: either it is an empty list, denoted by `sorted_nil`; it contains a single job, represented by `sorted_1`; or for any two consecutive jobs x and y along with the remaining list l , if the burst time of job x is less than or equal to that of job y , and the rest of the list l , is also sorted, then the entire list consisting of x , y , and l is considered sorted.

LEMMA 3.4. *For any job a and any joblist l , if l is sorted in ascending order of burst times, then inserting job a into l while preserving the sorted order results in a new sorted list.*

PROOF. We begin by induction on the list l .

Base Case: When l is an empty list, inserting a creates a singleton list containing only a , which is trivially sorted.

Inductive Step: Suppose l is non-empty and consists of a head job x and a tail list of jobs l' . We assume that inserting a into l' while preserving the sorted order yields a new sorted list.

Now, we consider two cases based on the burst time comparison between a and x :

- If the burst time of a is less than or equal to the burst time of x , then a should be inserted before x to maintain the sorted order. We insert a at the beginning of l and recursively apply the induction hypothesis to the tail list l' .
- If the burst time of a is greater than the burst time of x , we continue recursively with the induction on the tail list l' .

```
Lemma insert_sorted : forall a l, sorted(l) ->
sorted (insert_job a l).
Proof.
  intros a l S. induction S; simpl.
  + apply sorted_1.
  + bdestruct (leb (get_burst a) (get_burst x)).
    - apply sorted_cons.
      ++ apply H.
      ++ apply sorted_1.
    - apply sorted_cons.
      ++ lia.
      ++ apply sorted_1.
  + bdestruct (leb (get_burst a) (get_burst x)).
    - apply sorted_cons.
      ++ apply H0.
      ++ apply sorted_cons.
        -- apply H.
        -- apply S.
    - bdestruct (leb (get_burst a) (get_burst x)).
      ++ bdestruct (leb (get_burst a) (get_burst y)).
        -- apply sorted_cons.
          +++ lia.
          +++ apply sorted_cons.
            ---- apply H2.
            ---- apply S.
        -- lia.
      ++ bdestruct (leb (get_burst a) (get_burst y)).
        -- apply sorted_cons.
          +++ lia.
          +++ apply sorted_cons.
            ---- apply H2.
            ---- apply S.
        -- apply sorted_cons.
          +++ apply H.
          +++ simpl. unfold insert_job in IHS.
            bdestruct (leb (get_burst a)
              (get_burst y)).
              ---- lia.
              ---- apply IHS. Qed.
```

Next we define the `insert_sorted` lemma, which states that inserting a job into a sorted list while preserving its sorted order results in another sorted list. The proof of this lemma is carried out through induction on the input job list. In the base case, when the input list is either empty or consists of a single job, the lemma trivially holds as inserting a job maintains the sorted property. For the inductive step, we consider the case where the input list contains multiple jobs, and we demonstrate that inserting a new job into this list while preserving its sorted order ensures that the resulting list remains sorted. This proof involves utilizing boolean destruction or the `bdestruct` tactic to handle cases involving comparisons of burst times between jobs.

THEOREM 3.5. *For any joblist l , the schedule generated by $(sjf l)$ will be sorted.*

PROOF. Base case: If the list l is empty, there's nothing to sort, so it is trivially considered sorted.

Inductive case:

- Assume that when applying the sjf function to a joblist l , the resulting schedule will be sorted. In other words, $(sorted (sjf l))$ is true.
- The sjf function schedules jobs in accordance to their burst times. It needs to be proven that when job a is inserted into l , the overall schedule remains sorted. Through using the lemma $insert_sorted$, this shows that the entire schedule with the job a inserted remains sorted.

□

Theorem sjf_sorted : forall l, sorted (sjf l).

Proof.

```
intros. simpl. induction l.
- simpl. apply sorted_nil.
- simpl. apply insert_sorted. apply IHl. Qed.
```

As in the permutation proof, we now verify through induction that that the SJF algorithm produces a sorted list of jobs according to ascending burst times. In the base case, it establishes that the scheduler produces a sorted list when applied to an empty list via the $sorted_nil$ case. In the inductive step, the proof simplifies the list and recursively applies the $insert_sorted$ lemma, which ensures that inserting a job into a sorted list maintains its sorted property.

Correctness Property. Building upon the individual proofs of permutation and sortedness, we define a predicate is_sorted_perm that captures both properties for the given scheduling function. This predicate asserts that for any input list of jobs, applying the function results in a permutation of the original list and produces a sorted list according to the SJF criteria. By unifying these properties into a single theorem $sjf_correct$, we formally verify that the SJF scheduler satisfies both permutation and sorted requirements simultaneously.

THEOREM 3.6. *Asserts that the sjf function is correct in the sense that it produces a schedule that fulfills two requirements:*

- (1) *Theorem 3.3: All jobs from the original list are included in the schedule exactly once. There are no missing or duplicate jobs.*
- (2) *Theorem 3.5: The jobs in the schedule are ordered by their burst times, with the shortest jobs scheduled first.*

PROOF. sjf_perm guarantees the permutation property, and sjf_sorted guarantees the sortedness property. By applying these theorems, we establish that the sjf function satisfies both requirements for a correct SJF implementation. □

```
Definition is_sorted_perm (f : joblist -> list
job) :=forall al, Permutation al (f al) /\
sorted (f al).
```

Theorem $sjf_correct$: $is_sorted_perm\ sjf$.

Proof.

```
unfold is_sorted_perm. intros al.
split.
```

```
+ apply sjf_perm.
+ apply sjf_sorted.
Qed.
```

4 DISCUSSION

The methodology outlined utilizes the Coq proof assistant to formally verify the correctness of a non-preemptive SJF scheduling algorithm. This approach offers several advantages over traditional testing methods.

By converting the SJF algorithm into Coq, we ensure a precise and unambiguous representation of its logic. This eliminates potential ambiguities that could arise from natural language descriptions or pseudo-code. Secondly, by verifying that the algorithm outputs a permutation of the input job list, we establish that all jobs are scheduled exactly once, avoiding issues like job starvation. Finally, by verifying that the jobs are sorted in ascending order of burst times within the schedule, we formally guarantee that the benefit of the SJF algorithm is adhered to—minimizing the average wait time for jobs in the system.

However, it is important to acknowledge the limitations of this approach. The verification focuses on a simplified, offline version of the SJF algorithm. Real-world scheduling often includes dynamic job arrivals and potentially more complex scheduling criteria. Future work could explore the verification of a less restrictive system, such as considering variable burst times, to extend the verification method to dynamic scheduling environments.

Furthermore, while Coq offers a high degree of assurance, the quality of the verification ultimately depends on the correctness of the initial formalization of the algorithm. Hence, diligent validation of the formalization is essential to ensure the overall validity of the formal verification process.

5 CONCLUSION

Throughout this paper, we have provided our methodology for creating a formal correctness proof for the Shortest Job First algorithm in Coq. Our approach involves defining data types for jobs, implementing the SJF scheduler function, and providing proof of correctness via permutation and sortedness properties. By directly embedding the SJF algorithm in Coq and conducting rigorous correctness proofs, we ensure the reliability and predictability of the scheduler's behavior. Furthermore, we have shown how our formalization serves as a foundation for future research and development in scheduling algorithms. We invite readers to examine the provided SJF code and accompanying proofs to verify our findings and contribute to the advancement of formal methods in computer science.

REFERENCES

- [1] Kimaya Bedarkar, Mariam Vardishvili, Sergey Bozhko, Marco Maida, and Björn B Brandenburg. 2022. From intuition to coq: A case study in verified response-time analysis 1 of FIFO scheduling. In *2022 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 197–210.
- [2] Andrew Dessler. 2024. *ATMO 321 PYTHON FOR ATMOSPHERIC SCIENCES*. LibreTexts.
- [3] Simon Doherty, Lindsay Groves, Victor Luchangco, and Mark Moir. 2004. Formal Verification of a Practical Lock-Free Queue Algorithm. (2004).
- [4] Yalin Hu and Robert Armstrong. 2011. A Survey of Formal Verification in Mission-critical, High-consequence Applications. (2011).

- [5] Kanaka Juvva. 1998. Real-Time systems. (1998). <https://users.ece.cmu.edu/koopman/dess99/realtime/>
- [6] Moez Krichen. 2023. A survey on formal verification and validation techniques for internet of things. *Applied Sciences* 13, 14 (2023), 8122.
- [7] Pramod Kumar, Lalit Kumar Singh, and Chiranjeev Kumar. 2020. Performance evaluation of safety-critical systems of nuclear power plant systems. *Nuclear Engineering and Technology* 52, 3 (2020), 560–567.
- [8] Peng Li, Binoy Ravindran, and Syed Suhaib. 2004. A Formally Verified Application-Level Framework for Real-Time Scheduling on POSIX Real-Time Operating Systems. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING* (2004).
- [9] Divi Pruthvi. 2013. Reducing Context Switching Overhead by Processor Architecture Modification. *Advances in Electronic and Electric Engineering* (2013).
- [10] Tri Dharma Putra. 2020. Analysis of Preemptive Shortest Job First (SJF) Algorithm in CPU Scheduling. *International Journal of Advanced Research in Computer and Communication Engineering* 9, 4 (2020), 41–45.
- [11] Syed Shah, Ahmad Mahmood, and Alan Oxley. 2009. Hybrid Scheduling and Dual Queue Scheduling. (2009).
- [12] Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne. 2012. Operating system concepts. [SI].
- [13] The Coq Development Team. 2021. The Coq Proof Assistant Reference Manual – Version V8.13.1. (2021). <http://coq.inria.fr>
- [14] Florian Vanhems, Vlad Rusu, David Nowak, and Gilles Grimaud. 2022. A Formal Correctness Proof for an EDF Scheduler Implementation. (2022), 281–292.

AI Skill Check

An Examination of ChatGPT’s Consistency in Linguistics and Mathematics using the SAT

Ma. Elisha Kaye B. Nazario
De La Salle University Integrated School
Manila, Philippines
maria_elisha_nazario@dlsu.edu.ph

Eliza Janel L. Tan
De La Salle University Integrated School
Manila, Philippines
eliza_janel_tan@dlsu.edu.ph

Steven Vincent D. Lim
De La Salle University Integrated School
Manila, Philippines
steven_vincent_lim@dlsu.edu.ph

Keith Alexandre L. Ramos
De La Salle University Integrated School
Manila, Philippines
keith_ramos@dlsu.edu.ph

Shirley B. Chu
De La Salle University
Manila, Philippines
shirley.chu@dlsu.edu.ph

ABSTRACT

In the era of technological boom, the use of artificial intelligence is becoming increasingly intertwined with everyday societal systems. This includes the integration of Open AI’s ChatGPT into the way students learn today. As a result, the academic community has continuously assessed the performance of ChatGPT in various domains, especially in linguistics-related and mathematics-related fields. However, there still exists a gap to allow a more diverse understanding of ChatGPT’s consistency in answering questions on these domains at a high-school level. Therefore, this study aims to quantify the extent to which ChatGPT may generate inconsistent responses to high-school-level linguistics and mathematics questions when fed standardized SAT questions commonly used to evaluate a high-school student’s knowledge and aptitude. By extension, it provides a more detailed analysis of ChatGPT’s potential as a learning tool. Through this investigation, it was observed that ChatGPT generally demonstrates greater consistency in linguistics compared to mathematics, with different levels of reliability across distinct SAT subareas.

KEYWORDS

ChatGPT, SATs, consistency, high-school, learning tool

ACM Reference Format:

Ma. Elisha Kaye B. Nazario, Eliza Janel L. Tan, Steven Vincent D. Lim, Keith Alexandre L. Ramos, and Shirley B. Chu. 2024. AI Skill Check: An Examination of ChatGPT’s Consistency in Linguistics and Mathematics using the SAT. In *Proceedings of Philippine Computing Science Congress (PCSC2024)*. Laguna, Philippines, 8 pages.

1 INTRODUCTION

1.1 Background of the Study

The rapid development of technology over history has led to the rise of artificial intelligence (AI) that continues to unlock new opportunities in the education sector, benefitting both educators and life-long learners as affirmed by the United Nations Educational,

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

PCSC2024, May 2024, Laguna, Philippines

© 2024 Copyright held by the owner/author(s).

Scientific and Cultural Organization or UNESCO, on Sustainable Development Goal 4, focusing on Quality Education [14]. However, despite the convenience it may bring, the use of AI also prompted concerns and risks regarding its ethical usage, security, and long-term implications for student learning among its users. One such discourse revolves around ChatGPT. Its popularization in usage among students has been evident since its release in November 2022. [8] noted ChatGPT’s instant feedback capability prompts excessive reliance on it as a learning tool for Linguistics and Mathematics (subjects often taken at the high-school level). This is despite the risk of unintentional misinformation.

ChatGPT works by drawing from datasets up to September 2021. It then refines its responses using human feedback. This ability of ChatGPT prompted studies like [4] and [13] to analyze ChatGPT’s proficiency in test-taking scenarios. However, more scrutiny must still be done on its performance in high school contexts.

This highlights the significance of the College Board’s SATs. This standardized test gauges high school students’ college readiness, with over a million students taking it annually [1, 9, 12]. As students start utilizing ChatGPT for SAT preparations, understanding its inconsistencies, aside from its accuracy, is important to understand its effectiveness as a learning tool [2, 10].

This study aims to quantify inconsistent responses generated by ChatGPT when presented with high-school-level linguistics and mathematics questions.

1.2 Scope and Limitations

This study evaluates ChatGPT’s performance on selected data sets involving linguistics and mathematics SAT practice test questions gathered from both offline and online sources. Only questions with a defined answer, such as multiple-choice linguistics questions, multiple-choice mathematics questions, and free-response mathematics questions, are considered. The analysis of ChatGPT’s performance in this study mainly focuses on its responses’ consistency, to determine if it can consistently answer high-school level mathematics and linguistics questions correctly or incorrectly. Due to ChatGPT’s limitation involving images and geometric shapes, questions involving these are excluded.

This study also only covers understanding the performance of GPT-3.5 as this is the only version that is accessible for free by high school students.

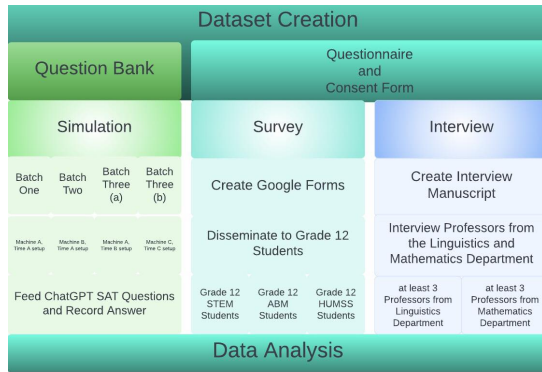


Figure 1: Schematic Representation of Study's Methodology

2 RELATED WORKS

A number of studies have delved into ChatGPT's performance and limitations across various academic disciplines. However, most focus on ChatGPT's accuracies in mathematics and linguistics in the context of post-secondary level. In mathematics, [11] highlights ChatGPT's ability to solve mathematical word problems. [6] explores its proficiency in advanced math, revealing both accurate and partially incorrect answers. On the other hand, [3] and [5] reveal that ChatGPT's linguistics proficiency is comparable to an average student in producing clear, concise responses in academic writing. Both cases show that ChatGPT has the ability to comprehend linguistic and mathematical expressions.

While there is limited research that directly studies the consistency of linguistics and mathematics, [6] has noted that while ChatGPT can indeed demonstrate enhanced language understanding abilities and deductive reasoning ability, it can still make mistakes that violate logical properties and that it can sometimes change its answer if a question is paraphrased.

3 METHODOLOGY

3.1 Research Design

While the complete study utilizes a mixed-method triangulation approach, this paper will focus on the quantitative approach, specifically testing ChatGPT through simulations. The entire procedure has been illustrated in Figure 1.

3.1.1 Dataset Creation. The dataset creation phase involves the compilation of a question bank to store and categorize SAT questions for linguistics and mathematics sets from different online and offline sources and record ChatGPT's response to each question. Questions are categorized and labeled by source and subject.

These datasets are randomly split into three batches of 40 questions for each domain. Each batch is divided equally among subareas of each domain.

In linguistics, questions are sourced from the Reading Test, and Writing and Language Test sections of the SAT. In each batch, ten questions come from each of the following subareas:

- (1) STANDARD ENGLISH CONVENTIONS, which focuses on sentence structure, usage, and punctuation;

Table 1: Example Revisions Made to Linguistics Questions for the Simulation

Original	Input to ChatGPT
A goat ingests the vegetation particular to the meadow in which it grazes, which, along with other environmental factors such as altitude and weather shapes the cheese's taste and texture.	A goat ingests the vegetation particular to the meadow in which it grazes, which, along with other environmental factors such as altitude and weather shapes the cheese's taste and texture.
A) NO CHANGE B) factors, such as altitude and weather, C) factors such as, altitude and weather, D) factors, such as altitude and weather	What improvements can be made to "factors such as altitude and weather"? A) NO CHANGE B) factors, such as altitude and weather, C) factors such as, altitude and weather, D) factors, such as altitude and weather

- (2) EXPRESSION OF IDEAS, which touches upon topic development, organization, and rhetorically effective use of language;
- (3) RELEVANT WORDS IN CONTEXT, which focuses on addressing word/phrase meaning in context and rhetorical word choice;
- (4) COMMAND OF EVIDENCE, which assesses the interpretation and usage of evidence found in passages and informational graphics (e.g. graphs, tables, and charts).

For mathematics, questions encompass two types: multiple choice and response. Ten questions from each of the following subareas in the mathematics section form each batch of dataset:

- (1) HEART OF ALGEBRA, which involves linear equations and inequalities questions;
- (2) PROBLEM SOLVING & DATA ANALYSIS, which tests quantitative reasoning and the interpretation of data (ratio and percentages);
- (3) PASSPORT TO ADVANCED MATH, which focuses on understanding expression structure, reasoning with more complex equations, and interpreting and building functions;
- (4) ADDITIONAL TOPICS IN MATHEMATICS, which focuses on other questions, including trigonometry and geometry.

Since input to ChatGPT is limited to text only, some questions are modified. Additional instructions are included to replace the underlined words or phrases in the original texts (see Table 1). \LaTeX was used for typesetting mathematical expressions that have complex symbols and structures. A sample is shown in Table 2.

3.1.2 Simulation. The simulation phase involves inputting the questions from the dataset to ChatGPT and recording its responses. Inspired by methodologies in [5, 11, 15], ChatGPT's consistency in a topic is examined by having it answer each question four times under different conditions or on different machines. The setups

Table 2: Example Revisions Made to Mathematics Questions for the Simulation

Original	Input to ChatGPT
The volume of a sphere is given by the formula $V = \frac{4}{3}\pi r^3$ where r is the radius of the sphere. Which of the following gives the radius of the sphere in terms of the volume of the sphere?	The volume of a sphere is given by the formula $V = \frac{4}{3}\pi r^3$ where r is the radius of the sphere. Which of the following gives the radius of the sphere in terms of the volume of the sphere?
A) $\frac{4\pi}{3V}$	A) $\frac{4\pi}{3V}$
B) $\frac{3V}{4\pi}$	B) $\frac{3V}{4\pi}$
C) $\sqrt[3]{\frac{4\pi}{3V}}$	C) $\sqrt[3]{\frac{4\pi}{3V}}$
D) $\sqrt[3]{\frac{3V}{4\pi}}$	D) $\sqrt[3]{\frac{3V}{4\pi}}$

Table 3: Independent Variables per Setup for Comparison

	Setup A (CONTROL) Time	Independent Location
Setup B		✓
Setup C	✓	
Setup D	✓	✓

include a Control Group (**Machine A, Time A**) for benchmarking, Machine Variation (**Machine B, Time A**) to assess machine-dependent responses, Temporal Variation (**Machine A, Time B**) for understanding temporal stability, and Location and Time Variation (**Machine C, Time C**) to explore external factors’ influence on performance. See Table 3 for a comparison of independent variables per setup.

During the simulation, ChatGPT was manually fed SAT questions. However, each batch utilized additional prompts to test the effects of the prompts on ChatGPT’s consistency. A “Give answer only” prompt was added to some questions in Batch 1, not included in Batch 2, and was added to all questions in Batch 3. Moreover, there are instances when ChatGPT changes its answer halfway through its explanation, i.e. its answer at the beginning, is different from its answer at the end of its explanation or its solution. In such cases, ChatGPT’s initial response is recorded.

3.2 Data Analysis

In order to compare and examine ChatGPT’s consistency in answering questions correctly for each SAT subarea, a manual counting of ChatGPT’s answers is performed. The number of times ChatGPT answered correctly in one setup, two setups, three setups, and all setups, or none of the setups, is also noted.

Additionally, to compare the consistency between linguistics and mathematics, the standard deviation of all setups per batch is calculated along with the total mean standard deviation.

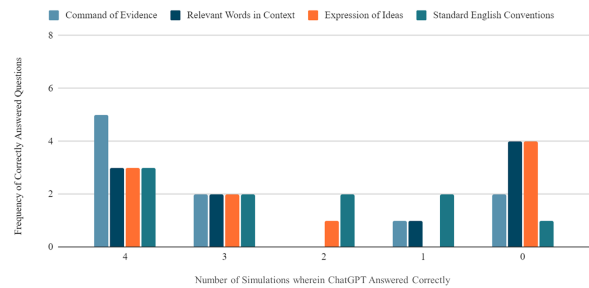


Figure 2: Batch 1 Results

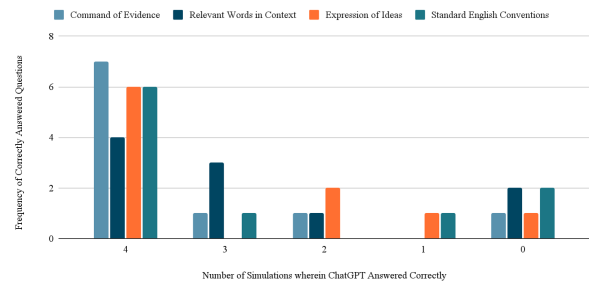


Figure 3: Batch 2 Results

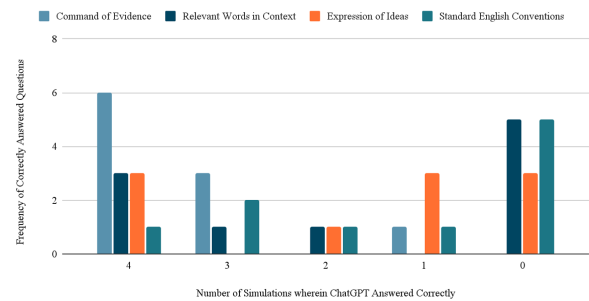


Figure 4: Batch 3a Results (with prompt)

4 RESULTS AND DISCUSSION

4.1 Linguistics

This chapter summarizes ChatGPT’s consistency in each of the SAT subareas for each batch and its overall consistency in each domain through the calculated mean standard deviation. The values gathered were based on ChatGPT’s raw accuracy score when it was fed SAT questions for each setup and each batch.

Consistency is defined as ChatGPT’s tendency to generate the same response when given the same prompt. Hence, the terms *consistently correctly* and *consistently incorrectly* shall be used in instances where ChatGPT was able to get the answer to a question correctly or incorrectly for all four setups of each batch.

Figure 2 shows that in Batch 1, ChatGPT is *consistently correct* in COMMAND OF EVIDENCE, being able to answer five questions

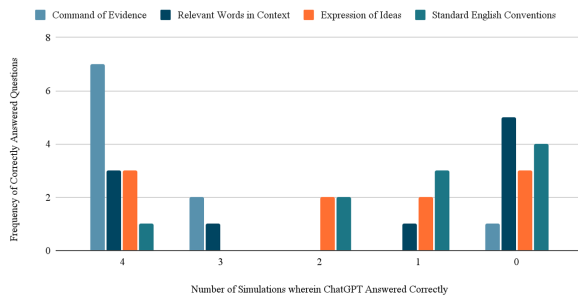


Figure 5: Batch 3b Results (without prompt)

Figure 6: ChatGPT’s Consistency in the SAT Linguistics Subareas

correctly in all four machines. The data also shows that ChatGPT is *consistently incorrect* in the subareas of RELEVANT WORDS IN CONTEXT and EXPRESSION OF IDEAS. This indicates that during the simulation of the first batch of questions, the software for all machines was more capable of noting details and answering questions about provided texts. However, it is relatively weak in determining the meanings of words and effectively expressing complete thoughts. Additionally, results show that in the STANDARD ENGLISH CONVENTIONS subarea, ChatGPT answered more questions *consistently correctly* than incorrectly in all machines.

Figure 3 shows that in Batch 2, ChatGPT answered questions *consistently correctly* more than incorrectly. This also shows ChatGPT’s improvement in all of the four subareas. Based on the Batch 2 results, ChatGPT answers most *consistently correctly* questions under COMMAND OF EVIDENCE. In this simulation, it answered seven questions correctly in all machines compared to five questions in the previous simulation. There is also an increase in correctly answered questions compared to the previous batch, which shows ChatGPT’s strength in answering questions about details in texts. Following COMMAND OF EVIDENCE, ChatGPT consistently answers six questions correctly in EXPRESSION OF IDEAS and STANDARD ENGLISH CONVENTIONS. This improvement from the previous batch’s results shows that ChatGPT is able to effectively utilize the English language to convey thoughts and identify grammatical errors in texts. Lastly, in all simulations, ChatGPT answers with the fewest questions correct in the RELEVANT WORDS IN CONTEXT subarea.

Figure 4 shows Batch 3a results. The “Give the answer only” prompt is included in all questions in this simulation. With the additional prompt, ChatGPT’s results changed significantly compared to the previous two batches. While ChatGPT maintains the highest consistency of correctly answered questions in the subarea of COMMAND OF EVIDENCE, results in the other three subareas begin to show its weaknesses. Firstly, it answered more questions *CONSISTENTLY INCORRECTLY* in the subareas of RELEVANT WORDS IN CONTEXT and STANDARD ENGLISH CONVENTIONS. Its worst performance is in the latter, wherein only one question in all machines is answered correctly. The batch’s results seem to show that restricting ChatGPT’s response to a selection between four options without giving it the freedom to explain its answers affects its

answers significantly. This is especially evident in the subareas of RELEVANT WORDS IN CONTEXT and STANDARD ENGLISH CONVENTIONS, wherein any form of explanation would be needed to understand the thought process that led ChatGPT to determine the definitions of words or the errors in sentence structure. In the EXPRESSION OF IDEAS subarea, ChatGPT neither consistently answers more questions correctly nor incorrectly.

When testing the same set of questions without the “Give answer only” prompt, there is no significant difference with regard to ChatGPT’s consistency (Figure 5). ChatGPT only consistently answers one additional question correctly across all batches under COMMAND OF EVIDENCE and is able to answer more questions correctly in some batches under the STANDARD ENGLISH CONVENTIONS subarea. This suggests that in linguistics, either prompt does not significantly affect consistency or a more appropriate prompt must be used.

4.1.1 *Consistency for each Subarea.* After analyzing the consistency of ChatGPT’s answers for the linguistics simulations, the following can be observed:

- (1) ChatGPT answers most *consistently correctly* in questions under the COMMAND OF EVIDENCE subarea. This is seen in all four simulations. This demonstrates ChatGPT’s capabilities in comprehending, noting details from, and answering questions about provided texts, a competency observed by other studies that tested its performance in answering reading comprehension questions, such as [5].
- (2) ChatGPT answers most *consistently incorrectly* in the RELEVANT WORDS IN CONTEXT subarea in all simulations. This indicates ChatGPT struggled in discerning the definitions of words used in sentences. This observation is quite ironic, as researchers such as [7] have noted its remarkable ability to generate accurate definitions for various words similar to the Collins Birmingham University International Language Database (COBUILD). Considering that some of the questions were offline-sourced and that limitations of text inputs meant that the words asked could not be highlighted in the questions themselves, there are still some factors that contribute to ChatGPT’s weakness in answering the questions under this subarea. While prompt engineering is outside the scope of this paper, other prompts may be identified to increase ChatGPT’s consistency.
- (3) Adding the “Give the answer only” prompt to the questions in Batch 3 does not make a difference. This only implies that ChatGPT’s consistency is not affected by this prompt. It does not necessarily mean that ChatGPT performs better without it, as that is beyond the scope of this paper.

4.2 Mathematics

It can be seen in Figure 7 that ChatGPT does not have a pattern in terms of consistency in any of the mathematics subareas. Notably, it was not able to get any questions correct across all machines in the ADDITIONAL TOPICS IN MATHEMATICS subarea. Moreover, it could only answer five questions correctly two times in the HEART OF ALGEBRA subarea. ChatGPT performs the best in PASSPORT TO ADVANCED MATH, not being *consistently incorrect* in any question

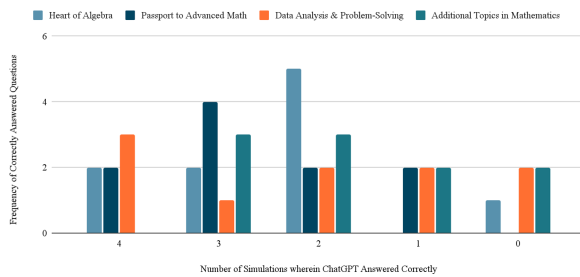


Figure 7: Batch 1 Results

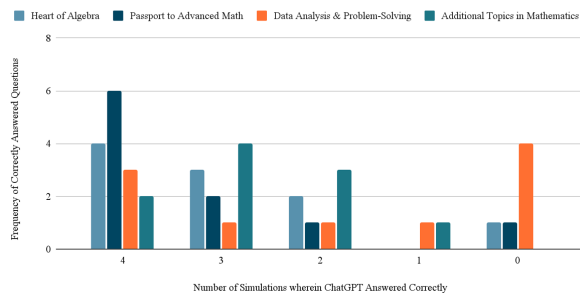


Figure 8: Batch 2 Results

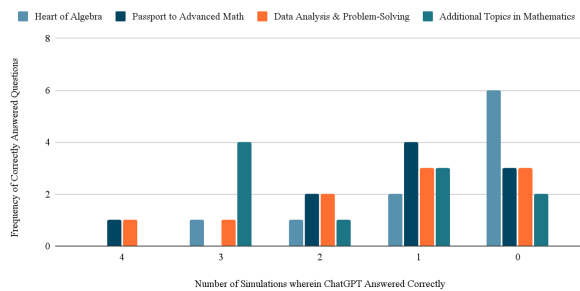


Figure 9: Batch 3a Results (with prompt)

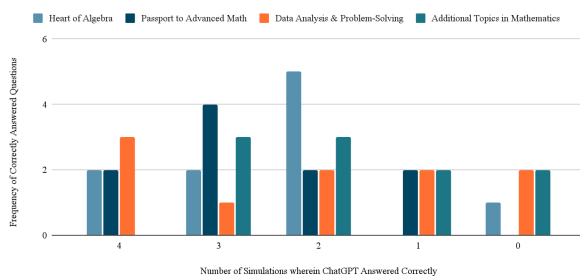


Figure 10: Batch 3b Results (without prompt)

Figure 11: ChatGPT’s Consistency in the SAT Mathematics Subareas

and getting four questions correct three times, albeit only being able to answer two questions correctly across all batches.

Figure 8 shows that ChatGPT exhibits significant improvement in Batch 2 in terms of consistency. It has the largest improvement in DATA ANALYSIS AND PROBLEM SOLVING and ADDITIONAL TOPICS IN MATHEMATICS. This may mean that ChatGPT may have been updated between the Batch 1 and Batch 2 simulations.

In contrast, the results of Batch 3a in Figure 9, which utilizes the “Give answer only” prompt, show a drastic performance decline in all subareas. Notably, ChatGPT answered *consistently incorrectly* six questions in HEART OF ALGEBRA subarea. In PASSPORT TO ADVANCED MATH, it only answered four questions correctly once. Moreover, it got no questions correct in all batches in HEART OF ALGEBRA and ADDITIONAL TOPICS IN MATHEMATICS. This illustrates that using the “Give answer only”, which prevents ChatGPT from generating solutions, increases its tendency to be *consistently incorrect*.

Without the “Give answer only” prompt, ChatGPT’s consistency returns to the similar trend observed in Batches 1 and 2. Figure 10 shows that ChatGPT answered six questions *consistently correctly* in all four machines in the PASSPORT TO ADVANCED MATH subarea. Results in this simulation show a decline in performance compared to Batch 2.

After analyzing the consistency of ChatGPT’s answers for the mathematics simulations, the following can be observed:

- (1) ChatGPT does not exhibit any clear trend regarding consistency and inconsistency in the mathematics subareas. For instance, PASSPORT TO ADVANCED MATH could be the most consistently correct subarea for Batch 1, but it is the least consistently correct by Batch 2. This supports the findings of [6], which posits that ChatGPT struggles to do accurate calculations consistently and falls short compared to chatbots specifically trained to do mathematics.
- (2) Succeeding batches show overall improvement. Batch 2 and 3b exhibit greater consistency in answering more questions correct as compared to Batch 1. This could imply that ChatGPT possibly improved over time due to the inputs it receives.
- (3) ChatGPT performs significantly worse if it is prompted only to give an answer without any solutions using the “Give answer only” prompt. It may be inferred that ChatGPT has a higher chance of arriving at a correct answer if it is given the liberty to generate a thought process or solution. It is worth noting that its overall performance may change if a different prompt is used to ask ChatGPT not to show any solution.
- (4) ChatGPT may not be suitable for mathematical contexts. ChatGPT is a text-oriented LLM chatbot that follows patterns from its online database. The nature of math problems changes depending on the wording and values provided in the question.

4.3 Comparison of ChatGPT’s Consistency between Linguistics and Mathematics

To determine in which domain—linguistics or mathematics, ChatGPT exhibits more consistency, the standard deviation of ChatGPT’s

Table 4: Standard Deviation of ChatGPT’s Achieved Linguistics Mathematics Scores per Batch

	Linguistics	Mathematics
Batch 1	1.25	10.05
Batch 2	2.72	3.95
Batch 3a	5.12	9.60
Batch 3b	5.41	2.80
Mean Standard Deviation	3.63	6.60

Table 5: ChatGPT’s Answer to a Mathematics Question with and without prompt

Question number: 1				
Expected answer: D				
	CS*	TV@	MV#	MTV°
Batch 3a ^a	C	B	A	B
Batch 3b ^b	D	D	D	D

* Control setup
 @ Temporal variation
 # Machine variation
 ° Temporal and Machine variation
^a with prompt
^b no prompt, may show solution

score in each batch is computed. Table 4 lists the standard deviation of all setups for all batches. A lower standard deviation value indicates higher consistency.

Data shows that ChatGPT’s scores in linguistics are closer to each other than in mathematics. Hence, ChatGPT is generally more consistent with linguistics. Based on the results of the simulations, ChatGPT is more consistent in linguistics than mathematics, showing its capability to aid high school students in linguistics, especially in tasks related to interpreting and analyzing passages. However, it scores the lowest in the Relevant Words in Context subarea, implicating that ChatGPT cannot successfully discern the context clues or words in sentences.

Meanwhile, in mathematics, it fails to exhibit any clear trend throughout all simulations. This shows that ChatGPT’s ability to consistently generate accurate answers in mathematics is still not entirely dependable, at least compared to linguistics.

Lastly, the use of the "Give answer only" prompt makes ChatGPT more prone to giving incorrect responses. This reveals that prompts have an impact on influencing ChatGPT’s consistency. This trend is highly exhibited in its responses to mathematics questions in batches 3a and 3b. Table 5 depicts a sample of its responses in each setup for the first question of these two batches. In this question, the correct answer was D. When ChatGPT was prompted to provide its answer only, its choices across all four setups were inconsistent and incorrect. Without the prompt, ChatGPT arrived at the correct answer for all setups. From these results, it can be observed that ChatGPT tends to answer more mathematical questions correctly when it has the liberty to show the thought process it used to arrive at the answer.

Figures 12 to 14 show sample conversations with ChatGPT. ChatGPT is asked to answer a Mathematics question. In Figure 12, the "Give answer only prompt" is included after posting the question to

Figure 12: Machine B Time A Question No. 29 (with prompt)

You
 At a restaurant, each large order of fries has 350 more calories than one large soda. If 2 large orders of fries and 3 large sodas have a total of 1,500 calories, how many calories does one large order of fries have?
 Give the answer only.

ChatGPT
 One large order of fries has 650 calories.

Figure 13: Machine B Time A Question No. 29 (no prompt)

ChatGPT
 Let’s denote the number of calories in one large soda as S (in calories).
 According to the given information, each large order of fries has 350 more calories than one large soda. Therefore, the number of calories in one large order of fries is $S + 350$.
 Now, the total number of calories from 2 large orders of fries and 3 large sodas is given as 1,500 calories:

$$2 \times (S + 350) + 3 \times S = 1,500$$
 Simplify the equation:

$$2S + 700 + 3S = 1,500$$
 Combine like terms:

$$5S + 700 = 1,500$$
 Subtract 700 from both sides:

$$5S = 800$$
 Divide by 5:

$$S = 160$$
 Now that we know the number of calories in one large soda ($S = 160$ calories), we can find the number of calories in one large order of fries:
 Calories in one large order of fries = $S + 350 = 160 + 350 = 510$
 Therefore, one large order of fries has 510 calories.

ChatGPT, answering 650 calories. In Figure 13, the same question is given to ChatGPT, excluding the prompt. ChatGPT answered the problem by showing its solution, obtaining an answer of 510 calories, different from its answer when a prompt was included at the end of the problem.

The conversation shown in Figure 14 is a Temporal and Machine variation setup. For this question, ChatGPT ended up with the same answer and solution as Figure 13, only using different variables. In other cases, ChatGPT may find other ways to approach a problem but still obtain the correct answer.

Tables 6 and 7 provide the number of instances ChatGPT remains consistent with its responses and answered correctly across all setups in both linguistics and mathematics. The following can be observed from the tabulated results:

- (1) ChatGPT has a higher frequency of maintaining correct answers than maintaining incorrect answers between the two batches in both linguistics and mathematics.
- (2) There is a higher frequency of non-matching answers between the two batches in the domain of mathematics.
- (3) In all four setups, ChatGPT gets more mathematics questions correctly in Batch 3b than in Batch 3a. This further suggests that ChatGPT tends to be more accurate when the "Give answer only" prompt is omitted.

Figure 14: Machine C Time C Question No. 29 (no prompt)

You
 At a restaurant, each large order of fries has 350 more calories than one large soda. If 2 large orders of fries and 3 large sodas have a total of 1,500 calories, how many calories does one large order of fries have?

ChatGPT
 Let's denote the number of calories in one large soda as x .

The number of calories in one large order of fries is $x + 350$ (since it has 350 more calories than one large soda).

Now, we are given that 2 large orders of fries and 3 large sodas have a total of 1,500 calories. We can set up the equation:

$$2 \cdot (x + 350) + 3 \cdot x = 1,500$$

Now, let's solve for x :

$$2x + 700 + 3x = 1,500$$

$$5x + 700 = 1,500$$

$$5x = 800$$

$$x = 160$$

Now that we know x , we can find the number of calories in one large order of fries:

$$\text{Calories in one large order of fries} = x + 350 = 160 + 350 = 510$$

Therefore, one large order of fries has 510 calories.

Table 6: Comparison of ChatGPT's Answers in Batches 3a and 3b (Linguistics)

	TQ*	MC@	MIC#	≠3a°	≠3b^	≠Inc'
Machine A, Time A	40	16	12	3	5	4
Machine B, Time A	40	19	9	3	4	5
Machine A, Time B	40	17	16	1	2	4
Machine C, Time C	40	15	6	7	5	7

- * Total Number of Questions
- @ Answers match, both correct
- # Answers match, both incorrect
- ° Answers don't match, correct in Batch 3a
- ^ Answers don't match, correct in Batch 3b
- ' Answers don't match, incorrect in both batches

(4) In instances where ChatGPT is correct in Batch 3a but not in Batch 3b, it is plausible that ChatGPT generated learned the said item. In other words, the question and its answer may have been part of the dataset used by ChatGPT in learning. As for the reason it becomes incorrect in the absence of the prompt, it is likely that ChatGPT follows a structure of problem-solving it has learned and generates a solution using one of these structures.

5 CONCLUSION

Based on the outcome of the simulations, ChatGPT is more consistent in linguistics than in mathematics. It showed its capability in linguistics, specifically in tasks related to interpreting and analyzing passages. However, the results show that ChatGPT cannot

Table 7: Comparison of ChatGPT's Answers in Batches 3a and 3b (Mathematics)

	TQ*	MC@	MIC#	≠3a°	≠3b^	≠Inc'
Machine A, Time A	40	12	1	2	13	12
Machine B, Time A	40	5	2	3	23	7
Machine A, Time B	40	16	3	2	13	6
Machine C, Time C	40	5	0	2	22	11

- * Total Number of Questions
- @ Answers match, both correct
- # Answers match, both incorrect
- ° Answers don't match, correct in Batch 3a
- ^ Answers don't match, correct in Batch 3b
- ' Answers don't match, incorrect in both batches

consistently discern the context clues or words in sentences. Meanwhile, in mathematics, ChatGPT fails to exhibit any clear trend throughout all simulations. This shows that ChatGPT's ability to consistently generate accurate answers in mathematics is still not entirely dependable. Lastly, utilizing the "Give answer only" prompt makes ChatGPT vulnerable to giving incorrect responses. This reveals that prompts play an important role in influencing ChatGPT's performance.

As this study assesses ChatGPT's consistency across different SAT subareas in linguistics and mathematics, future works could determine the performance of ChatGPT on the specific areas e.g. evaluation of simple arithmetic expressions, algebraic expression, geometry-related questions, along with the respective reasons. Moreover, prompts have a significant impact on ChatGPT's performance. Further studies may focus on identifying the proper prompts for ChatGPT to perform with a high level of consistency. A definition of how prompts should be formed for ChatGPT to achieve an ideal performance is also a possible research focus.

REFERENCES

- [1] College Board. 2023. SAT program results for the class of 2023 show continued growth in SAT participation. <https://newsroom.collegeboard.org/sat-program-results-class-2023-show-continued-growth-sat-participation>.
- [2] Ian Bogost. 2023. Is this the singularity for standardized tests? <https://www.theatlantic.com/technology/archive/2023/03/open-ai-gpt4-standardized-tests-sat-ap-exams/673458/>.
- [3] Peter Andr'e Busch and Geir Inge Hausvik. 2023. Too good to be true? An empirical study of ChatGPT capabilities for academic writing and implications for academic misconduct. https://www.researchgate.net/publication/370106469_Too_Good_to_Be_True_An_Empirical_Study_of_ChatGPT_Capabilities_for_Academic_Writing_and_Implications_for_Academic_Misconduct. *Twenty-ninth Americas Conference on Information Systems*.
- [4] Jonathan H Choi, Kristin E Hickman, Amy B Monahan, and Daniel Schwarcz. 2023. ChatGPT goes to law school. <http://dx.doi.org/10.2139/ssrn.4335905>. *Journal of Legal Education* 71 (2023), 387.
- [5] Joost C. F. de Winter. 2023. Can ChatGPT Pass High School Exams on English Language Comprehension? <https://doi.org/10.1007/s40593-023-00372-z>. *International Journal of Artificial Intelligence in Education* (2023).
- [6] Simon Frieder, Luca Pinchetti, Alexis Chevalier, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Christian Petersen, and Julius Berner. 2023. Mathematical capabilities of ChatGPT. <https://arxiv.org/abs/2301.13867>.
- [7] Robert Lew. 2023. ChatGPT as a COBUILD lexicographer. <https://doi.org/10.1057/s41599-023-02119-6>. *Humanities and Social Sciences Communications* 10, 1 (2023), 704.
- [8] Neville J. McKenzie. March 8, 2023. The limitations and biases of ChatGPT: A critical look. <https://www.linkedin.com/pulse/limitations-biases-chatgpt-critical-look-neville-j-mckenzie>.
- [9] Hannah Muniz. 2021. The 4 SAT sections: What they test and how to do well. <https://blog.prepscholar.com/sat-sections>.

- [10] Erin Ohsie-Frauenhofer. 2023. The SAT will become fully digital—and shorter—by 2024. Here's what's changing and what's staying the same. <https://blog.arborbridge.com/sat-will-become-fully-digital-and-shorter-by-2024-whats-changing>.
- [11] Paulo Shakarian, Abhinav Koyyalamudi, Noel Ngu, and Lakshmivihari Mareedu. 2023. An Independent Evaluation of ChatGPT on Mathematical Word Problems (MWP). <https://api.semanticscholar.org/CorpusID:257219363>. *ArXiv abs/2302.13814* (2023).
- [12] USAFacts Team. 2022. Are fewer students taking the SAT? <https://usafacts.org/articles/are-fewer-students-taking-the-sat/>.
- [13] Christian Terwiesch. 2023. *Would Chat GPT3 get a Wharton MBA? A prediction based on its performance in the operations management course*. Technical Report. William and Phyllis Mack Institute for Innovation Management. <https://mackinstitute.wharton.upenn.edu/wp-content/uploads/2023/01/Christian-Terwiesch-Chat-GTP.pdf>.
- [14] UNESCO. 2024. Sustainable development goal 4 (SDG 4).
- [15] Yousef Wardat, Mohammad A. Tashtoush, Rommel AlAli, and Adeeb M. Jarrah. 2023. ChatGPT: A revolutionary tool for teaching and learning mathematics. <https://doi.org/10.29333/ejmste/13272>. *Eurasia Journal of Mathematics, Science and Technology Education* 19, 7 (2023).

Formal Verification of a Custom Scheduling Algorithm and Specification of a Round-Robin Implementation using Coq

Christian Choa
University of the Philippines Diliman
cjchoa@up.edu.ph

Brylle Logroño
University of the Philippines Diliman
bblogrono@up.edu.ph

Raymart Villos
University of the Philippines Diliman
rpvillos@up.edu.ph

Alfonso B. Labao
University of the Philippines Diliman
ablabao@up.edu.ph

Henry N. Adorna
University of the Philippines Diliman
hнадorna@up.edu.ph

ABSTRACT

The Operating System (OS) scheduler is crucial for resource allocation and task execution. However, ensuring the dependability of scheduling algorithms is challenging due to possible human errors in testing. Formal verification provides a systematic approach to prove algorithm correctness and soundness. By mathematically modeling and verifying properties, formal methods ensure scheduler implementations adhere to specifications and function correctly. In this study, we used Coq to formally verify a custom scheduling algorithm, *mysched*, and formulated related correctness and soundness theorems that were then proven using Coq proof tactics. Moreover, we also provided formal specifications of the Round-Robin scheduling algorithm in Coq.

CCS CONCEPTS

• **Theory of computation** → **Logic and verification**;

KEYWORDS

Formal Verification, Coq, Operating System, Round-Robin Scheduling Algorithm

ACM Reference Format:

Christian Choa, Brylle Logroño, Raymart Villos, Alfonso B. Labao, and Henry N. Adorna. 2024. Formal Verification of a Custom Scheduling Algorithm and Specification of a Round-Robin Implementation using Coq. In *Proceedings of Philippine Computing Science Congress (PCSC2024)*. Laguna, Philippines, 6 pages.

1 INTRODUCTION

Operating Systems (OS) are software programs that facilitate the execution of tasks and manage computer resources. As an essential component of most computers, the different modules of the OS should be correctly programmed to ensure the entire system's security, functionality, and efficiency. The scheduler determines which task runs in the CPU and is among the most important parts of the OS [2]. Its purpose is to optimize the utilization and allocation of CPU resources, to improve system responsiveness, and to increase overall system performance.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

PCSC2024, May 2024, Laguna, Philippines

© 2024 Copyright held by the owner/author(s).

The Formal verification process involves modeling the scheduler as a mathematical object and proving that it implements the desired functions correctly. Coq is a formal proof assistant that provides a formal language to write mathematical definitions, executable algorithms, and theorems [9]. Formal proof assistants like Coq are used to mechanize the proofs of the soundness and other related properties of scheduling algorithms. Formal verification of an operating system (OS) scheduler is crucial for ensuring its correctness and reliability.

This paper will show proof of concept through a custom scheduling algorithm called *mysched* as well as show Coq specifications for its algorithm, properties, and proofs of the properties. In addition, Coq specifications for the Round-robin algorithm will be shown afterwards.

The proof assistant of choice for our research was Coq because it is built upon a strongly typed core language called the Calculus of Inductive Constructions (CIC). This ensures a high degree of correctness in proofs by enforcing type consistency throughout the development process. Furthermore, the strong type system helps catch many errors at compile time rather than at runtime, leading to more reliable proofs. Another good thing about Coq is that it can automatically extract executable programs from specifications, generating Objective Caml or Haskell source code. This feature facilitates the practical application of formal methods by bridging the gap between formal specifications and executable implementations. Next, Coq also supports expressive types, including inductive structures, dependent types, and subset types (Σ -types). These types enable precise specifications and proofs, allowing users to capture complex mathematical structures and properties. Following, the syntax of Coq is more similar to a programming language as opposed to the syntax of Isabelle, a different formal proof assistant, which uses common mathematical ASCII symbols in proof which makes it slightly inconvenient when typing on an IDE [10]. Finally, Coq implements a functional programming language supporting its types. This allows users to express functions over inductive types, write proofs using case analysis, and interactively evaluate expressions, enhancing the flexibility and usability of Coq as a proof assistant [9].

1.1 Related Work

A case study done by Bedarkar et.al verified the Response-Time Analysis (RTA) of FIFO scheduling with the use of Coq Proof Assistance. The study aims to motivate more researchers to explore the use of proof assistants to minimize human error in mathematical

reasoning. In addition, the study points out that there have been a lot of intuitive results in the past that, at first, seemed sound but was later to be proven incorrect. Verification was done by defining the system model, encoding the scheduling policy and preemption model, proving abstract work conservation, bounding the maximum busy-window length and delay within a busy window, and defining the search space [3]. This process will be similarly seen in the following listed studies.

In another study, Sun and Lei formally verified a task scheduler that selects the highest priority ready task for Embedded Operating Systems (EOS) [8]. Here, the researchers verified the EOS task scheduler at different abstraction levels under a unified framework. They had three parts which are: defining the API specifications, defining the functional specifications, and lastly, linking together the aforementioned modules once verified.

Lastly, the formal verification of the OS Kernel seL4, which uses a round-robin scheduler, was done through Isabelle by Klein et.al [7]. According to the researchers, seL4 is the first-ever general-purpose OS kernel that is fully formally verified for functional correctness which will allow for the construction of more secure and reliable systems on top. Verification process is done through different specification layers such as abstract specification, executable specification, C implementation, and machine model which only focuses on the cache and TLB. The researchers conclude that performance does not have to be sacrificed for formal verification. In addition, it was also concluded that "future application proofs can rely on the formal kernel specification that seL4 has been proven to implement". Isabelle and Coq have similarities and differences with one another. Both Isabelle and Coq have automatic tactics for proving or simplification of complex statements. They both as well have approximately the same size of proof and both have a large set of libraries. Isabelle is different to Coq in its expressiveness of underlying logic. While Isabelle uses classical higher-order logic, Coq uses intuitionistic logic [10]. In addition, the author Yushkovskiy believes that Coq requires a deeper understanding of underlying logic theory; however, both Isabelle and Coq would require unfamiliar users a lot of required additional learning in order to understand the process of proving [10].

1.2 Significance

Computers with operating systems are used everywhere in the world. The computer has integrated itself to be a necessity in all aspects of the world. From day-to-day devices such as smartphones to complex computer systems that manage transportation systems [1], ensuring that these computers work as intended holds significant importance in maintaining the safety of those that rely on them. One of the roles of a computer's operating system is scheduling. There are different kinds of scheduling algorithms, such as the First In, First Out algorithm, Shortest Job to Completion algorithm, and the Round-robin algorithm, each with its own strengths, weaknesses, and appropriateness of usage. [4]

This study aims to formally verify the round-robin algorithm through Coq proof assistance. In doing so, the researchers will be able to determine if the algorithm is sound and will be able to find any errors, limitations, and bugs that exist in the algorithm.

If the round-robin algorithm is successfully formally verified, it is ensured that operating systems that use the round-robin algorithm use a sound scheduling algorithm, therefore ensuring that computers work as intended in that aspect. Additionally, future researchers will be able to use this verification of the round-robin algorithm to check the soundness of other algorithms that is based on the round-robin algorithm. [5].

Further down the line, the formal verification of a round-robin algorithm will provide a step in the right direction to formally verify real-time systems. The round-robin algorithm, being the simplest preemptive scheduling algorithm, has so many different applications, uses, and algorithms based on it. Below is a study on a round-robin based load balancing in Software Defined Networking (SDN):

Results show that round-robin strategy is better than random strategy because round-robin distributes the load uniformly while random does not. [6]

The verification of the round-robin algorithm will allow for the transition from the random strategy to the round-robin strategy to be simpler. It would remove the need to ensure that the round-robin algorithm is sound since it has already been verified through Coq proof assistance.

Formal verification of a widely used algorithm, such as the round-robin algorithm, ensures that programs and real-world systems that use the algorithm will work and behave as intended which will allow for assurance that computers used all around the world stay safe and work as intended.

2 SCHEDULERS

In developing a proof for the algorithm, an accurate translation to Coq is one of the primary priorities. There are many ways to achieve an implementation that satisfies the definition of the scheduling algorithm.

However, careful considerations should be kept in mind because of how it will affect the complexity of proving the theorems. Representations must also be chosen such that they satisfy the structural requirements of Coq. Coq follows a functional paradigm, hence, mutable states and variables are not used. Recursive functions must also be well-founded. Well-founded recursion generalizes both strong induction and recursion.

In this paper, the behavior of an actual operating system scheduler is simplified to achieve a bounded complexity in the theorems we aim to prove. Nevertheless, these simplified models do not have behaviors that we do not expect from a process task being scheduled by an operating system. The tasks or processes all have unique identifiers, some amount of burst time, and in the case of the Round-robin Scheduler, a total burst time that is divisible by the time slice. Further delimitations were applied to the scheduler. All processes to be scheduled arrive at the same time at $t = 0$, the burst times of all the processes are equal, and the algorithm is offline. Offline algorithms are given the whole problem data from the beginning. Finally, the overhead for context-switching between two processes is disregarded.

2.1 Custom Scheduling

The custom scheduling algorithm, *mysched*, processes a list of task priority numbers. It iterates through the priorities from left to right, executing a task immediately if its priority is higher than the sum of all remaining priorities to its right. Otherwise, it stores the task in a FIFO queue for later execution. After examining all tasks, it executes them in the order they were stored in the FIFO queue.

Algorithm 1 *mysched* Algorithm

```

1: procedure MYSCHEd(tasks)
2:    $n \leftarrow$  length of tasks
3:   queue  $\leftarrow$  empty queue
4:   for  $i \leftarrow 1$  to  $n$  do
5:     if  $\text{tasks}[i] > \sum_{j=i+1}^n \text{tasks}[j]$  then
6:       execute task  $\text{tasks}[i]$ 
7:     else
8:       push  $\text{tasks}[i]$  onto queue
9:     end if
10:  end for
11:  while queue is not empty do
12:    execute task  $\text{pop}(\text{queue})$ 
13:  end while
14: end procedure
    
```

2.1.1 Algorithm.

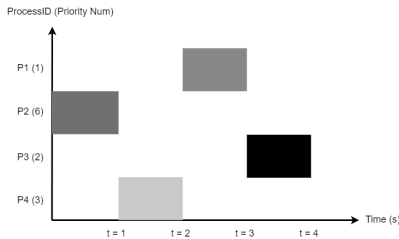


Figure 1: Custom Scheduler Sample Execution

Process ID	Priority Number	Arrival Time	Burst Time
1	1	0	1
2	6	0	1
3	2	0	1
4	3	0	1

Table 1: Example Process Table

The *mysched* algorithm takes in as input a list of task priority numbers seen in Table 1. The processes arrive at the same time. However, the scheduler still has to determine which among the processes to process first. Hence, we break ties by ordering them on increasing process IDs. The scheduler then examines the list from left to right. If it finds that the priority number of a task is strictly greater than the sum of all the other priority numbers to the right,

the algorithm will immediately execute the task. Consider process P2 in Table 1. It executed immediately since its priority number $6 > 2 + 3$. It is the first process to execute since, for the other case, the algorithm places the task in a First-in-First-Out(FIFO) queue first. It then goes on to examine the next number in the list. Once the list has been examined, the algorithm executes the FIFO queue.

2.1.2 Coq Translation.

```

Fixpoint mysched (l : list nat) : list nat :=
  match l with
  | [] => []
  | h::t => if ((sum t) <? h) then
     $\hookrightarrow$  h::(mysched t)
    else (mysched t) ++ [h]
  end.
    
```

The *mysched* function operates on a list of natural numbers (nat). When provided an empty list, it returns an empty list as well. However, for a non-empty list l , which has a head element h and a tail list t , if the sum of the elements in t is less than h , h is placed at the beginning of the result obtained by recursively applying *mysched* to t . On the other hand, if the sum of elements in t is greater than or equal to h , h is appended at the end of the result obtained by recursively applying *mysched* to t .

2.1.3 Soundness Property. We define a theorem on the custom scheduler soundness.

THEOREM 2.1. For all input list L , the output of the algorithm are permutations of the list L .

```

Theorem mysched_is_sound :
  forall l, Permutation l (mysched l).
Proof.
  intros. induction l.
  - simpl. apply Permutation_refl.
  - simpl. bdestruct (sum l <? a).
    + change (a::l) with ([a]++l).
      change (a::mysched l) with
         $\hookrightarrow$  ([a]++(mysched l)).
      apply Permutation_app_head. apply
         $\hookrightarrow$  IHl.
    + change (a::l) with ([a]++l).
      apply perm_trans with (l ++ [a]).
      -- apply Permutation_app_comm.
      -- apply Permutation_app_tail.
      -- apply IHl.
Qed.
    
```

This theorem states that running *mysched* with a given input list, its output list will be a rearrangement of the input list. The algorithm does not add or remove any items on the list, for it only changes the order of the input list. This theorem indicates that the algorithm works correctly by ensuring that no items in the list are added or removed since the output list is only a reordering of the input list.

The Coq proof shows that the *mysched* function is sound by showing that the function outputs the same list but rearranged. It

proves this by looking at two cases: when the list l is empty and when it is not.

In the case that the list is empty, there is nothing to rearrange; therefore, the rearrangement of the list is the same as the input.

When the list has elements, it checks if the sum of the remaining elements is less than the first element. If it is, it sets the first element as the first item in the output list, otherwise, it is placed at the end of the output list.

The proof proceeds by breaking down the list into its constituents and applying appropriate permutations to maintain the desired ordering. This is achieved through various applications of permutation properties, such as reflexivity, transitivity, and properties of list concatenation, ultimately proving that `mysched` produces a permutation of the input list.

2.1.4 Correctness Property. To aid with the proof for the correctness of the custom scheduler, we have a helper function that gets the index of a task.

```

Fixpoint getindex (l : list nat) (v : nat) :
  ↪ nat :=

  match l with
  | [] => 0
  | h::t => if (v =? h) then 0 else
            1 + getindex t v
  end.

```

The following is the Coq code for the proof of correctness to demonstrate that the intended behavior of the algorithm is assured.

```

Theorem mysched_propone : forall a b l,
  (b =? a = false) ->
  (0 <=? getindex (mysched l) a = true) ->
  (1 <=? getindex (mysched l) b = true) ->
  (0 <=? getindex (mysched l++[b]) a = true)
  ↪ ->
  (1 <=? getindex (mysched l++[b]) b = true)
  ↪ ->
  (sum (b::l) <? a = true) ->
  ((getindex (mysched (a::b::l)) a <?
  ↪ getindex (mysched (a::b::l)) b) =
  ↪ true).

```

Proof.

```

intros. simpl. bdestruct (b + sum l <? a).
+ simpl. bdestruct (b =? a).
  -- discriminate H.
  -- bdestruct (sum l <? b).
  ++ simpl. bdestruct (b =? b).
  --- bdestruct (a =? a).
  +++ apply Nat.ltb_lt. lia.
  +++ bdestruct (a =? b).
  ---- apply helper_one in
    ↪ H9. destruct H9.
      reflexivity.
  ---- apply helper_one in
    ↪ H9. destruct H9.

```

```

      reflexivity.
  --- bdestruct (a =? a).
  +++ apply Nat.ltb_lt. lia.
  +++ bdestruct (a =? b).
  ---- apply Nat.ltb_lt.
    ↪ lia.
  ---- apply Nat.ltb_lt.
    apply Nat.leb_le
    ↪ in H0.
    apply Nat.leb_le
    ↪ in H1.
    lia.
  ++ simpl. bdestruct (a =? a).
  --- apply Nat.ltb_lt. lia.
  --- apply Nat.leb_le in H2.
    apply Nat.leb_le in H3.
    lia.
+ simpl. bdestruct (sum l <? b).
  -- apply Nat.leb_le in H4. simpl in H4.
    apply helper_two in H4. destruct
    ↪ H4. apply H5.
  -- apply Nat.leb_le in H4. simpl in H4.
    apply helper_two in H4. destruct
    ↪ H4. apply H5.

```

Qed.

This theorem guarantees that when a new item task b is added to an existing list of tasks, the `mysched` algorithm will make sure that the priority ordering of task a and the newly added task b will be maintained. If a has a higher priority than the new task b , it will maintain that status in the list.

This theorem checks different scenarios to confirm this property. It considers where tasks a and b are positioned in the output list of `mysched`, their priorities relative to each other, and how they compare to other tasks in the list.

The theorem `mysched_propone` is focused on proving that the `mysched` algorithm maintains the correct order between tasks a and b in the output list, even after adding task b to an existing list of tasks. The proof examines various scenarios to ensure this correctness:

- (1) If the combined priority of task b and the tasks in the list is lower than the priority of task a , then the proof makes sure that a retains its priority over task b in the resulting list.
- (2) If the combined priority of task b and the tasks in the list is greater than the priority of task a , then the proof makes sure that b gets priority over task a in the resulting list.
- (3) Additional conditions are considered to handle cases where tasks a and b have equal priority or are not present in the list, ensuring the correctness of the algorithm under various circumstances.

The proof relies on breaking down scenarios using boolean decomposition and employing helper lemmas to address each case. By doing so, it demonstrates that the `mysched` algorithm successfully preserves the relative ordering of tasks a and b based on their priorities, therefore confirming the algorithm's correctness even after task b is added to the list.

2.2 Round-Robin Scheduling

2.2.1 Algorithm.

Algorithm 2 Round-Robin

```

1: // Initialize variables
2: ready_queue ← Queue for processes ready to execute
3: curr_proc ← None
4: quantum ← 5
5: while ready_queue is not empty do
6:   curr_proc ← ready_queue.dequeue()
7:   remaining_time ← min(quantum, curr_proc.remaining_burst)
8:   for t ← 1 to remaining_time do
9:     curr_proc.remaining_burst ←
curr_proc.remaining_burst - 1
11:    if curr_proc.remaining_burst = 0 then
12:      // Process finished, no longer in the ready queue
13:      break
14:    end if
15:  end for
16:  if curr_proc.remaining_burst > 0 then
17:    // Process not finished, enqueue back to the ready queue
18:    ready_queue.enqueue(curr_proc)
19:  end if
20: // All processes finished, algorithm ends
    
```

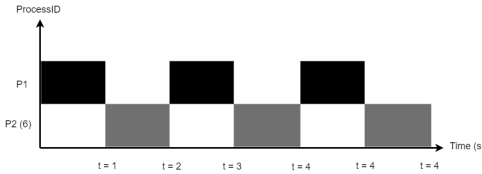


Figure 2: Round-robin Algorithm Execution Example

Process ID	Arrival Time	Burst Time
1	0	3
2	0	3

Table 2: Example Process Table

The Round-Robin scheduling algorithm assigns a fixed time unit, known as quantum, to each process during which it can execute. If a process does not complete within its quantum, it is temporarily suspended, and the next process in the queue is given a chance to execute. The suspended process is then placed at the end of the ready queue to await its next turn. This process continues until all processes have completed execution. The pseudo code shown in Algorithm 2 shows a possible implementation of the Round-Robin algorithm and will not necessarily be the exact specification used for the Coq implementation.

We can see from Table 2 that the two processes— P_1 and P_2 —both have the same burst times of 3 ms and arrive at 0 ms. It is also evident that both processes alternately run for 1 ms each in the order P_1 and P_2 , respectively until their burst times are satisfied. Since they both have a burst time of 3 ms, it is expected that the last process, P_2 , would end after 6 ms, which is the case.

2.2.2 Coq Translation.

```

Fixpoint rr (l1 : list nat) (l2 : list nat)
  ↪ (n : nat) : list nat :=
match n with
| 0 => []
| S n' => match l1, l2 with
| [], [] => []
| [], h::t => h::(rr [] t n')
| h::t, [] => h::(rr t [] n')
| a::b, c::d => a::c::(rr b d n')
end
end.
    
```

The fixpoint `rr` recursively implements an implementation of a Round-robin algorithm. The fixpoint `rr` takes in two lists of the `nat` type, takes the first element of each list, and concatenates them into an output list of type `nat`.

2.2.3 Soundness Property. For the theorem on the algorithms soundness, we will define two axioms that will assist in formulating the proof.

AXIOM 1. *If it is true that all id's in list l1 is ID i, then the length of adding another job with id i to l1 is 1 + the length of previous joblist.*

AXIOM 2. *If it is true that all id's in list l1 is id i, then it is true as well for the tail of the list.*

We will need another axiom for the second argument of the algorithm.

AXIOM 3. *If it is true that all id's in list l2 is ID i, then the length of adding another job with id i to l2 is 1 + the length of previous job list.*

For the theorem on soundness, we have the following definition.

THEOREM 2.2. *For all lists l1, l2, and an element x: If the length of the result of a function rr applied to an empty list [], l2, and x is 0, and If the uniform function applied to l1 is true, Then the length of l1 is equal to the length of the result of the rr function applied to l1, l2, and x.*

```

Axiom firstaxiom: forall l1 l2 x a, uniform
  ↪ l1 1 = true ->
  getlength (rr (a :: l1) l2 x) 1 = S
  ↪ (getlength (rr (l1) l2 x) 1).

Axiom secondaxiom: forall l a, uniform (a ::
  ↪ l) 1 = true -> uniform (l) 1 = true.

Theorem soundnessone : forall l1 l2 x,
  0 = getlength (rr [] l2 x) 1
  -> uniform l1 1 = true
    
```



```

-> (length l1) = (getlength (rr l1 l2
  ↪ x) 1).
Proof.
  intros. induction l1.
  - simpl. apply H.
  - simpl. rewrite firstaxiom. rewrite IH1.
    ↪ reflexivity.
    apply secondaxiom in H0. apply H0.
    apply secondaxiom in H0. apply H0.
Qed.

Axiom thirdaxiom: forall l1 l2 x a, uniform
  ↪ l2 2 = true ->
  getlength (rr l1 (a::l2) x) 2 = S
  ↪ (getlength (rr (l1) l2 x) 2).

Axiom fourthaxiom: forall l a, uniform (a ::
  ↪ l) 2 = true -> uniform (l) 2 = true.

Theorem soundnesstwo : forall l1 l2 x,
  0 = getlength (rr l1 [] x) 2
  -> uniform l2 2 = true
  -> (length l2) = (getlength (rr l1 l2
  ↪ x) 2).
Proof.
  intros. induction l2.
  - simpl. apply H.
  - simpl. rewrite thirdaxiom. rewrite IH2.
    ↪ reflexivity.
    apply fourthaxiom in H0. apply H0.
    apply fourthaxiom in H0. apply H0.
Qed.

```

In the Coq code, we can see the use of Axioms within Coq's proof environment. Axioms are crucial since they allow users to express mathematical ideas and theories that may not be fully captured by the system's built-in rules and definitions. However, it must be noted that careful selection and reasoning about axioms are essential to ensure that the formalized theories are both complete and consistent.

The first axiom states that if all id's in the list are 1, then adding another job with id 1 in the list will result to its length being one more than the length of the previous job list

The first Soundness theorem states that if all id's in the list are 1, then applying the round-robin algorithm to the list will result in a list where the number of jobs with id 1 is equal to the length of the list.

We can observe that the third and fourth axioms and the second Soundness theorem are counterparts for the second list input.

3 CONCLUSIONS

We have demonstrated the application of formal verification techniques using Coq to ensure the correctness and soundness of a custom scheduling algorithm, referred to as mysched, and of a Round-Robin scheduling algorithm in a simpler case. By mathematically modeling these algorithms and proving their properties using

Coq, we have shown that formal methods can be effectively used to ensure that scheduler implementations adhere to specifications and function correctly based on how they were defined.

We have selected a subset of properties to be proven through the formal verification process over a simplified model of schedulers. We, nonetheless, have highlighted the importance of scheduling algorithms in operating systems and the need for rigorous verification methods to ensure their dependability. The formal verification of these algorithms is crucial for ensuring the correctness and reliability of operating systems, which are integral to various computer systems and applications worldwide.

Moving forward, the study can be expanded to more general cases. Some cases future researchers could consider would be varying the arrival time and total burst times of the processes. This, however, entails increased complexity in creating the proofs. The theorems demonstrated could also serve as a starting point for proofs of the general case. The general case is desired since it models actual OS behavior.

The use of formal verification techniques, as demonstrated in this study, will continue to play a vital role in ensuring the safety and functionality of computer systems in various applications and industries.

REFERENCES

- [1] Md. Zahangir Alam, Mahfuzulhoq Chowdhury, and Parijat Prashun Purohit. 2014. Development of an Intelligent Traffic Management System Based on Modified Round-Robin Algorithm. *International Journal of Control and Automation* 7, 12 (Dec. 2014), 121–132. <https://doi.org/10.14257/ijca.2014.7.12.12>
- [2] Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau. 2018. *Operating Systems: Three Easy Pieces*. CreateSpace Independent Publishing Platform, North Charleston, SC, USA. <https://doi.org/10.5555/3299537>
- [3] Kimaya Bedarkar, Mariam Vardishvili, Sergey Bozhko, Marco Maida, and Bjorn B. Brandenburg. 2022. From Intuition to Coq: A Case Study in Verified Response-Time Analysis 1 of FIFO Scheduling. In *2022 IEEE Real-Time Systems Symposium (RTSS)*. IEEE. <https://doi.org/10.1109/rtss55097.2022.00026>
- [4] S Bharathi, MP Chethan, and SN Darshan. 2022. Comprehensive Analysis of CPU Scheduling Algorithms. *International Research Journal of Modernization in Engineering Technology and Science* (2022).
- [5] Yosef Hasan Jbara. 2019. A new Improved Round Robin-Based Scheduling Algorithm-A comparative Analysis. In *2019 International Conference on Computer and Information Sciences (ICCIS)*. IEEE. <https://doi.org/10.1109/iccis.2019.8716476>
- [6] Sukhveer Kaur, Krishan Kumar, Japinder Singh, and Navtej Singh Ghuman. 2015. Round-robin based load balancing in Software Defined Networking. In *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, 2136–2139.
- [7] Gerwin Klein, Kevin Elphinstone, Gernot Heiser, June Andronick, David Cock, Philip Derrin, Dhammika Elkaduwe, Kai Engelhardt, Rafal Kolanski, Michael Norrish, Thomas Sewell, Harvey Tuch, and Simon Winwood. 2009. seL4: formal verification of an OS kernel. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles (SOSP09)*. ACM. <https://doi.org/10.1145/1629575.1629596>
- [8] Haiyong Sun and Hang Lei. 2020. Formal verification of a task scheduler for embedded operating systems. *Journal of Intelligent amp; Fuzzy Systems* 38, 2 (Feb. 2020), 1391–1399. <https://doi.org/10.3233/jifs-179502>
- [9] The Coq Development Team. [n. d.]. *The Coq Reference Manual*.
- [10] Artem Yushkovskiy. 2018. Comparison of Two Theorem Provers: Isabelle/HOL and Coq. In *proceedings of the Seminar in Computer Science (CS-E4000)*, Aalto Univeristy, Autumn 2017. (2018). arXiv:arXiv:1808.09701

Modeling Perceptual Organization in Abstract Art Using Eye Tracking Data

Carlo Ram M. Ferrer
Ateneo de Manila University
Quezon City, Metro Manila, Philippines
carlo.ferrer@obf.ateneo.edu

Maria Mercedes T. Rodrigo
Ateneo de Manila University
Quezon City, Metro Manila, Philippines
mrodrigo@ateneo.edu

ABSTRACT

The review of quantitative studies of Gestalt principle of proximity modelling perceptual organization showed insufficient evidence of its consistency in different contexts. Most of the methods mentioned the use of eye tracking as a viable alternative approach to quantify perceptual organization through clustering of gaze movements, specifically fixations of participants. The domain in which these studies are conducted includes the perception of artistic paintings. Subjectivity of visual perception may be explained by the differences in participants' gaze behaviors and art expertise. Based on their limitations, further research was seen to determine whether Gestalt grouping principles can neutralize subjectivity between experts and novices while viewing art. This study reveals novice and advance participant gaze clustering are likely equal, although having low adherence to the Gestalt grouping principle of proximity.

KEYWORDS

Gestalt theory, visual perception, eye tracking, abstract art

1 INTRODUCTION

Human beings have the innate capacity to generate groupings out of relevant structures from complex visual images without previous knowledge of its components [6]. This ability is known as *perceptual organization*. Perceptual organization occurs when disjoint objects are perceived as a single, unified whole [20]. Gestalt theory proposes that the perceptual process is *wholistic* and integrative, that human beings perceive wholes rather than the parts [3, 7, 14]. Max Wertheimer, one of Gestalt's founders, discovered several Gestalt laws of perceptual grouping that come into play in the perceptual process including the principle of proximity [6, 14, 21]. It was stated as the tendency of human beings to perceive closely grouped objects to be more related than distant objects.

Because of the subjective nature of perception, there were criticisms to the consistency of Gestalt grouping principles in various contexts. First, demonstrations of Gestalt theorists used stimuli with low naturalness or *ecological validity* to represent real-life objects [23]. Second, Gestaltists have not specified how perceptual organization can vary depending on differences, such as age, experience [11] and circumstance, among individuals. In the context of art appreciation, varying levels of art expertise between

novice and expert can have an impact on the way the work is perceived [17, 19]. Experts tend to have a greater capability to detect patterns than novices [12]. For them, experience had strong significance in shaping human perceptual development concerning a cognitive aspect in perception [15, 18, 23]. Yet, Gestaltists denied the existence of the influence of past experiences and cognition in all perceptual experiences [9, 11].

Understanding how humans make use of vision to acquire specific information from an environment with visual noise is one of the difficulties cognitive sciences tries to solve [5, 8, 22]. In fact, questions regarding how the brain processes visual information [10] and how these are organized to make representations of faces, objects, and art [4] remains unanswered [23]. Although several theories of perception such as Gestalt theory [16] have been used to explain perceptual organization, these still lack quantitative models that are needed in various fields such as computer vision [1, 13]. Unless there are quantitative approaches to perceptual grouping, the challenge for computer vision research to efficiently understand features in the real world remains without finding how important visual information is organized.

One possible means of building these quantitative models is using eye tracking data. Pedreira and Quiroga's eye tracking study, *How Do We See Art*, attempts to bridge science and arts by quantifying how subjects view paintings [4]. This paper follows up on Pedreira and Quiroga's findings by analyzing subjects' gaze movement variability to determine the effect of their level of expertise in art on their gaze behaviors when viewing abstract art. Specifically, the paper uses Gestalt grouping principles to find whether these principles promote uniformity in visual perception and perceptual organization based on the compliance of subjects' gaze behaviors. In section 2, related work is described. The paper provides conclusion and future work in section 6.

2 RELATED WORK

2.1 Eye Tracking and Gestalt Principle of Proximity

An alternative method to quantify visual perception is through eye-tracking. In a bottom-up approach, eye gaze data can be used to validate the effects of the principle of proximity. Gaze movements identify informative details of a stimuli [27, 26] that report cognitive factors such as areas of interest relating to their

attention. Thus, gaze movements can provide a general analysis of hypotheses and can evaluate the variability of individual gaze behaviors [28].

Alfred Yarbus states how human thought processes can be observed through their gaze behaviors from their perception of complex objects [28]. According to Yarbus, gaze movements accompany attention fixation (on stationary object's elements) [28]. These gaze movements may be in the form of voluntary or involuntary saccades measuring from a duration of at least 0.01 to 0.07 seconds depending on its amplitude of approximately 20 rotational degrees [29, 28]. These series of saccades change into different areas of fixation of the perceptual field to a feature element of perception [28]. Thus, fixation points derived from saccades can possibly present feature elements perceived as clusters or regions of interest as proposed by Gestaltists.

More recent studies specified a definite real-life context by viewing a more natural stimulus such as abstract and representational paintings. These studies used eye tracking as inputs to computational models clustering eye gaze data into regions of high interest or attention which may indicate a subject's compliance to basic image concepts and Gestalt grouping principles. Khushnood Naqshbandi et. al. studied application of automatic clustering models for eye gaze data [30]. These clustering algorithms operate similar, if not equal to, the principle of proximity. They used k-means and a density-based clustering algorithm, OPTICS, to cluster gaze points from a static stimulus before their classification. Although both performed well, their application of OPTICS gave higher success rates compared to k-means with proper hyperparameters [30]. The study used Seeing Machines FaceLAB 5 infrared cameras as gaze sensors, performed calibration and used three paintings as their stimuli [30].

2.2 Applications of Eye Tracking and Gestalt Theory on the Principle of Abstract Art

A domain in which methods quantifying perceptual organization can be tested is by studying how humans perceive art. Art history has had research on gaze movements since the sixth century [29]. In fact, psychological research on visual perception used gaze data based on artistic paintings as stimuli. The first eye tracking experiment was done by Guy Buswell in 1935 where he recorded eye tracking data from 200 individuals who viewed different paintings, drawings, and other works of art [29]. Even so, his results were not able to derive conclusions solely related to aesthetic sciences and psychology [29].

Raphael Rosenberg and Christoph Klein expressed similar findings from Buswell. Their conclusions emphasized a high regard for variation between individuals, especially between experts and novices [15]. Another eye tracking study by Carlos Pedreira and Rodrigo Quirán Quiroga attempts a more in-depth analysis of variations of participants' gaze movements by observing their compliance to art principles while viewing art [4]. Their findings show that participants' gaze behaviors behaved similarly while viewing the original version of *Composition II* in red, blue, and yellow and had varying gaze behaviors when viewing its modified version [4]. Pedreira and Quiroga's discussion concludes basic

uniformity in their participants' gaze behaviors [4]. However, participants' prior knowledge of the artwork generated large variability in their results [4].

The high level of subjectivity observed in the method applied by the preceding studies may be explained by studying the function of art as discussed by Patrick Cavanaugh. According to Cavanaugh, the creation of art by artists makes the artist a practitioner of physics by representing real world elements into his or her artwork [31]. Like the function of the visual brain, the function of art allows us to search for "essential and enduring features of objects, surfaces, faces, situations, and so on, which allows us to acquire knowledge" [8]. To facilitate this acquisition of knowledge, artists allow the beholder of their art to view their creation from their own perspective through different techniques like abstraction.

Rudolf Arnheim discusses the concept of perceptual abstraction in art in his works [8]. According to Arnheim, children manifest abstraction in perception as a basic operation [33, 32]. To illustrate, he takes note of how children represent human faces as circles [32]. The use of this type of representation (of children) denotes an abstraction of the face as a round object.

Definitions of perceptual abstraction and current studies on abstract art generate questions of opposition. Do gaze movements (in a specific context such as abstract art) present uniformity based on principles at a minimum as demonstrated by Pedreira and Quiroga? Or do these present subjectivities between differences of individuals, especially experts and novices? Current work on Gestalt perceptual organization and eye tracking does not account for all contexts. Specifically, further research is needed to determine whether Gestalt accounts for all levels of expertise.

3 METHODS

3.1 Structure of the Study

In this study, the research design uses a triangulation research design model to analyze qualitative and quantitative data: pre-test questionnaire, post-test interview and eye-tracking data. The pre-test questionnaire is used to evaluate participants' art expertise and background, and to classify participants to novices and experts. An eye tracking experiment is conducted for both novice and expert groups to gather quantitative data on their gaze behavior while viewing digital abstract art. The post-test interview serves to validate quantitative comparison of novice and expert gaze behaviors including their fixation data and scanpath patterns, and to enhance comprehensive evaluation of the effects of proximity in their gaze behaviors. With this method, the chosen model shall answer the research questions: "how can Gestalt grouping principle of proximity be quantified in the context of abstract art?", and "to what extent does the level of expertise in art affect the Gestalt grouping principle of proximity when viewing abstract art?".

Before the actual testing, a pre-test is conducted to determine modifications in the materials used in the study. During the pre-test, the research protocol is performed as designed. Data gathered in the pre-test is used to test instruments and procedures and is not included in the final dataset. Finally, the research protocol is

executed for actual testing based on modifications in the instruments and procedures based on the test data.

3.2 Selection of Stimuli

To select the appropriate set of non-representational abstract art, two classically or formally trained art studies experts proficient in art criticism are consulted. First, one expert is asked to aid in the selection of the stimuli based on two criteria. The abstract art print must exhibit both Gestalt proximity effects and art composition relationships which can also be analyzed by experts using any other formal criteria they may have while viewing art. To reduce bias in artistic styles, prints are chosen from varying artists but not limited to a specific period. Lastly, the second consultant shall verify and review the validity of the selected prints based on the criteria mentioned. Only six prints from the master list must be selected as part of the shortlist of the stimuli for the experiment. To guide, a formal document is provided to each consultant including the objectives of the study, the instructions for the selection of prints, and an informed consent form.



Figure 1: Sample art print “Untitled/5” by Antonio Lorenzo

To be able to determine areas of interest (AOI), consultants are asked to mark their expected clustered regions on each stimulus based on their evaluation of the abstract art prints. These shall be the bases to computationally determine groupings based on the principle of proximity. To do this, digital versions of the stimuli are resized to fit the dimensions of the screen. Each stimulus is assessed using a K-Means image clustering technique. Clusters formed from the algorithm are adjusted based on the validation of the consultants against their expected clustered regions.

3.3 Eye Tracking Experiment

The eye tracking study is performed remotely through Zoom Conferencing Tool as compromise to the situation caused by the 2019 N-Coronavirus. The room and environment of the remote participants must be well-lit, enclosed, and silent for the duration of the experiment. For this study, different integrated and standalone non-infrared webcams with varying resolution qualities are used under RealEye Eye Tracking Software to provide a higher accuracy rate regarding webcam differences. Each webcam must have at least 640 x 480 resolution (low-image granularity) at 720p and 30 FPS to meet the minimum requirements of RealEye. The eye tracking

solution also requires running in Google Chrome web browser. This solution gathers data online and can be retrieved or exported by the researcher. Exported data includes raw gaze data, fixation duration, coordinates, and summaries of fixation data per stimulus.

Table 1: Participant Count per Educational Degree

Participant Degree	Count
Information Design, Art Management	8
Information Systems, Computer Science	4
Management, Economics	3
Legal, Political Science	4
Illustrator	1

20 participants composed of university graduates and college undergraduate students are asked to participate as volunteers for the study. Participants of the study were selected through convenience sampling and are male and female and are from 18 to 32 years old, with excellent visual acuity and good contrast sensitivity. Participants are asked to read and sign the informed consent form containing the purpose of the study, a description of the experiment and the rights of the participants to serve as a formal document of the communication between the researchers and the participants.

Participants who agree to carry on with the experiment are asked to answer a pre-test questionnaire which includes:

- Participant Demographic: Age, Sex, and Course
- Visual Acuity
- Background in Art Principles or Formal Education
- Background in Contemporary and Abstract Art Works

To protect their privacy, each participant was identified only based on their control number in the questionnaire. The questionnaire is adjusted specific to the academic expectations of a consultant art appreciation instructor in the university to evaluate art expertise of the participant. Aside from formal art background, participants’ knowledge in artistic principles and elements are further evaluated based on 12 multiple choice questions which consists of basic principles, and elements of art and design: Principles of Design (2), Principles of Art (5), Elements of Art (5). Additionally, participants are asked to provide a self-assessment of their perceived expertise and appreciation in art based on a Likert scale with weighted scores of 1 (Strongly disagree) to 5 (Strongly agree). Participants who correctly answered at least 10 out of 12 multiple choice questions and have scored at least a mean score of 4.0 in their self-assessment are considered as “experts”. Both criteria must be met to be considered as an expert. Participants who scored otherwise are considered as “novices”.

Each participant is informed of a free viewing of six abstract art in succession and has exactly 30 seconds viewing time per stimulus. For every participant, the viewing sequence of stimuli is randomized and is asynchronous. Once all art prints have been shown, participants are interviewed. The interview includes an evaluation of the participant’s familiarity of each abstract art shown, perceived aesthetic qualities of the art, perceived prominent areas of

the art and their viewing criteria for each print. Participants are provided digital copies of the abstract art shown for their annotations during the demonstration. Furthermore, participants are asked regarding observance of the presence of Gestalt principle of proximity. Specifically, they are asked whether they were able to distinguish grouped components in each art print.

3.4 Scanpath Pairwise Distance Analyses

Participants' scanpath were compared against participants within each sample group and against each sample group to measure effects of art expertise using Eyanalysis. First, the denoised raw gazes eye tracking data from RealEye was transformed to retrieve the scanpath for each participant per art print viewed. The transformed data was a dataset consisting of a dictionary where each dictionary entry is a single named scanpath. A scanpath consists of a list of points. A point consists of a list of coordinates. This scanpath data was grouped per art print for analysis, and this was categorized based on participant sample group.

The transformed data was derived to determine the pairwise distance between the raw gazes of all participants against other participants. Specifically, the pairwise distances were calculated using the cross-compare function of the Eyanalysis Python Library to validate the statistical significance of intra and inter sample group scanpath similarities with respect to measuring the effects of art expertise. Eyanalysis is a modified Mannan linear distance method which compares fixation positions based on spatiotemporal similarity [24]. Since the original Mannan linear distance only maps each coordinate of a scanpath set to its nearest neighbor from another scanpath set without considering fixation order or sequence [24], Eyanalysis uses timestamp of each coordinate to measure similarity between sequences of fixation positions without considering the pre-defined areas of interest [24]. Similarity of scanpath is measured based on their distance measure. A scanpath with a distance measure D of 100 pixels are considered similar as proposed by the author of the algorithm, Sebastiaan Mathot [25]. The distance measure is given by the equation function [26]:

$$D(S, T) = \frac{\sum_{i=1}^n d_S^i + \sum_{j=1}^n d_T^j}{\max(n_S, n_T)}$$

where d is the distance calculated for each point in scanpath S and T . To determine variance within participant categories, participants' scanpath of each group are compared. Each participant's scanpath is compared against the scanpath of all participants within their group using Eyanalysis.

For both participant groups, standard deviation SD is computed using ANOVA for all stimuli to determine the spread of the distribution between scanpath within groups.

The pairwise distances of the sample groups were compared against each other using Welch's t-Test per art print to confirm there is no statistical significant difference between the gaze behavior of non-art participants and participants with formal background in art. Specifically, the test was used to determine the mean statistical difference of the gaze behavior of the sample groups in relation to

the standard error of their mean using their gaze scan path pairwise distances, wherein the standard error of the mean demonstrates how far the sample mean of the data was likely to be within the true population mean. Welch's t-Test was determined to be the statistical test used due to unequal variances of the samples. The Brown-Forsythe test was used for validation for each art print t-Test result.

3.5 Implementation of Intersection Over Union of Convex Polygons of Gaze Clustering

Participant gaze clustering results were analyzed to measure the effect of the Gestalt grouping principle of proximity and to validate results of the scan path distance analysis. Density-based clustering algorithms, DBSCAN and OPTICS, were used to identify high-density regions of arbitrary shapes. Each participant gaze clustering was compared. Common clustering comparison methods, such as Adjusted Rand Indexing and Adjusted Mutual Information, Silhouette Coefficient, were identified as not suitable due to varying sample sizes of gaze points from each participant. Hence, the intersection over the union (IoU) of the convex polygons of the gaze cluster results were calculated to compare clustering results of each participant against each other. The results confirmed observations from the scan path pairwise distance analysis.

DBSCAN and OPTICS clustering models were used to analyze the gaze clustering of participant gazes to measure the effects of proximity using Scikit Learn machine learning libraries. The OPTICS model used the Minkowski distance metric wherein its parameters included minimum of 6 neighbor gaze points to define a core point and with minimum size of 0.5 gaze points to define a cluster. This was also tested using the DBSCAN cluster method, and compared against the initial Xi cluster method, with maximum distance between gaze points of 1.0 and 2.0 epsilon cuts performed separately. The epsilon cuts of the DBSCAN cluster method were defined through the K Nearest Neighbors (KNN) elbow method. OPTICS was finally used based on its performance and as it was relatively insensitive to parameter settings and can automatically adjust epsilon cuts with $\text{eps} = \text{infinity}$.

4.4.1 Implementation of Intersection over Union of Convex Polygons of Gaze Clustering. IoU scores were calculated for all possible comparisons of each participant within and outside their sample groups for each art print. Where IoU is calculated by the formula:

$$\begin{aligned} \text{Area}(Q1 \cup Q2) &= \text{Area}(Q1) \cup \text{Area}(Q2) - \text{Area}(Q1 \cap Q2) \\ \text{IoU} &= \text{Area}(Q1 \cap Q2) / \text{Area}(Q1 \cup Q2) \end{aligned}$$

where $Q1$ is the gaze clustering set of the first participant and $Q2$ is the gaze clustering set of the second participants. IoU scores of 1 predicts that convex polygons exactly match, and scores of 0 predicts that convex polygons do not. IoU scores are interpreted as:

$$\begin{aligned} \text{IoU} > 0.5 &\text{ is "decent"} \\ \text{IoU} > 0.7 &\text{ is "good"} \\ \text{IoU} > 0.9 &\text{ is "almost perfect"} \\ \text{IoU} = 1 &\text{ is "equal"} \end{aligned}$$

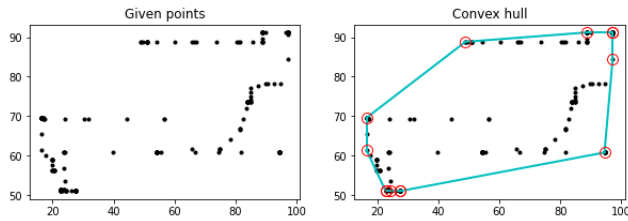


Figure 2: Formation of Convex Hull from Gaze Clusters

To obtain the IoU scores, convex polygons or convex hulls of the OPTICS gaze clusters were calculated using the SciPy Spatial Convex Hull Python Library. The library provides the vertices and the simplices in or to define the bounding box of the convex hull. This also provided the areas and volume of the convex polygons obtained for each cluster.

The IoU was calculated for each convex hull in a participant scan path against all convex hull of the second participant scan path to find its most similar cluster, since cluster count for each scan path varied. The IoU of the two most similar clusters from the first and second scan path was achieved by calculating the largest IoU score iteratively for each defined hull of the participants.

The average IoU scores of each participant were measured to achieve the average IoU score of the intra-group and inter-group comparisons to compare intra and inter group scores as the sample sizes of each group are unequal.

4 RESULTS

4.2 Pre-test Results

The revised research protocol was executed for the final study. The assessment test results are outlined below.

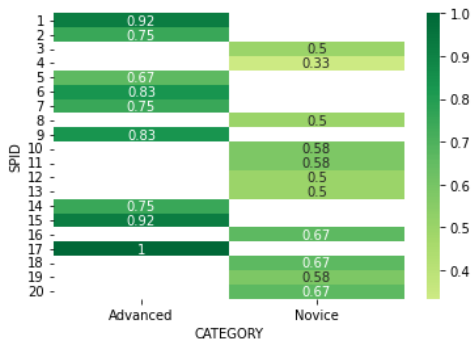


Figure 3: Participant Artistic Knowledge Assessment Average Scores

4.2.1 Assessment Test Section 1. Participants with formal educational background in arts or are currently art professionals had test scores greater than or equal to 67%. Overall, the art participant sample group attained an average of 82% test score.

Participants who had no formal background in arts obtained an average of 55% with the lowest test score of 33%. Only three out

of eleven participants achieved test scores of 67% or above, however these participants failed to pass Section 2. To further validate, Section 2 results were analyzed to verify results of the first assessment test and to confirm the relationship between the sample groups.

4.2.2 Assessment Test Section 2. To demonstrate, the diverging stacked bar charts below illustrate the level of agreement of participants per their sampling categories.

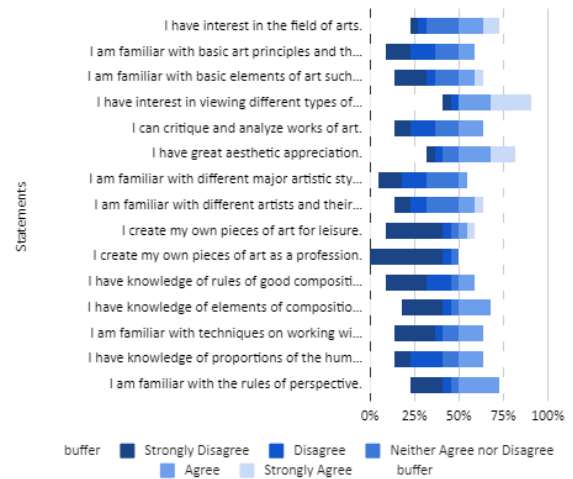


Figure 4: Level of Agreement of Novice Art Participants

Figure 4 shows lower perceived artistic familiarity, aesthetic appreciation, and fundamental knowledge on visual arts of participants with no formal background in arts. Although, nine out of eleven participants in the group agreed to have interest in viewing pieces of art. Similarly, the median for the group had a good perceived aesthetic appreciation.



Figure 5: Level of Agreement of Advanced Art Participants

Figure 5 shows higher perceived fundamental knowledge, aesthetic appreciation, and interest in arts of participants with formal education and profession in visual arts. However, 25-50% of these participants had neutral agreement on statements regarding their knowledge of basic art principles and elements of composition.

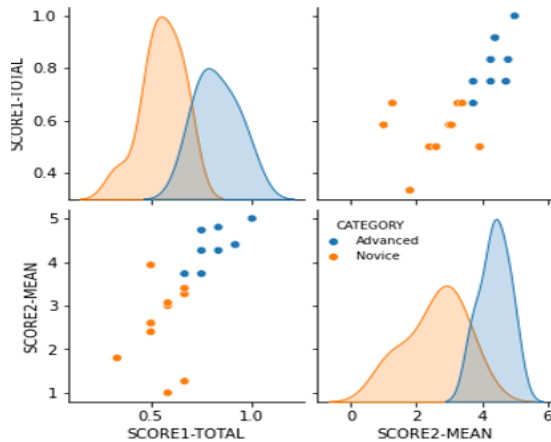


Figure 6: Participant Knowledge Assessment Pair Plot

Correlation analysis between participant category or sample group and their assessment tests showed novice participants with lower perceived artistic knowledge with respect to their section 1 assessment tests scores compared to participants with formal background in arts. Figure 6 shows symmetric unimodal distribution of participants with no background in arts and have relatively lower scores in section 1 of the assessment test compared to participants with formal background in arts.

To further analyze the difference between the groups’ perceived interest and knowledge in visual arts, a Mann-Whitney U Test (MWU) was executed using the mean scores of each participant per group. This is to determine the difference between the participant groups regardless of direction with the null hypothesis stating no significant difference between the groups. Samples from both groups were combined with respect to their art background. The derived U Value from the test was 2 with a critical value of U 23 at the significance level of $p < 0.05$. The calculated z-score was -3.57076 at $p = 0.00036$. Given these, the result showed significant difference between advanced and novice groups at two-tailed significance level of $p < 0.05$. Hence, the sampling grouping of the participants was maintained after the assessment test as categorization for the eye tracking experiment results analysis with respect to Section 1 results. This was the final participant grouping used for the analysis of aggregated eye gaze data in the succeeding sections.

4.2.3 Eye Tracking Experiment Results. Several eye-tracking test data were exported from RealEye Dashboard – Gazes, Fixations, AOI-based exports, Facial Coding and Survey Results. Among three (3) gaze data exports, Denoised Raw Gazes export was used to ensure no additional interpolation of gazes within 50ms to 60Hz were applied to the dataset. The raw gaze data established recording of raw gazes every 30ms.

Exploratory data analysis was performed to validate the integrity of the exported data. 20 tests were completed overall for six stimuli, with 30 seconds viewing time (+/- 0.1 seconds) each. To validate the quality of each test, the test quality grades were studied for each tester and stimulus viewed. Stimulus results with less than a grade of 3 or “Good” were not included in the succeeding sections of the study to ensure quality of the eye tracking data analysis.

4.3 Scanpath Pairwise Analyses

The calculated pairwise distances for each sample group were analyzed per art print viewed. There are 45 pairwise distances for participants with no formal background in art with an average distance score of 0.47 pixels with standard deviation of 0.26, and only 28 pairwise distances for participants with formal background in art with an average distance score of 0.42 pixels and standard deviation of 0.11. The intra group scanpath distance analysis using Eyeanalysis presents similarity of scanpath through relatively shorter pairwise distances for participants with formal background in art.

Table 2: Welch's' t-Test and Brown-Forsythe Test Results Matrix

item_id	t	p tTest	W	p BF	DOF
1	3.08	0.0031	7.78	0.0068	65.11
2	2.80	0.0084	12.37	0.0008	33.71
3	2.14	0.0359	3.40	0.0692	66.42
4	0.27	0.7903	0.002	0.9620	63.69
5	0.49	0.6228	3.72	0.5774	70.99
6	1.26	0.2115	6.85	0.0108	64.63

The overall analysis showed there was a statistically significant difference between the pairwise distances of non-art participants and participants with formal background in art ($N=153$), for art print 1 and 2 only with at least $t = 2.54, p \leq .05$. This conformed with the initial IoU scores of the sample groups.

4.4 Measuring Effects of the Principle of Proximity

The researchers studied the average IoU scores for each art print to validate the sensitivity of the effects of the proximity principle with respect to the participants’ eye tracking behavior segments which exhibits higher visual perceptual activity and lower cognitive processes. The effects of proximity were measured against different scanpath lengths from its maximum duration of 30 seconds.

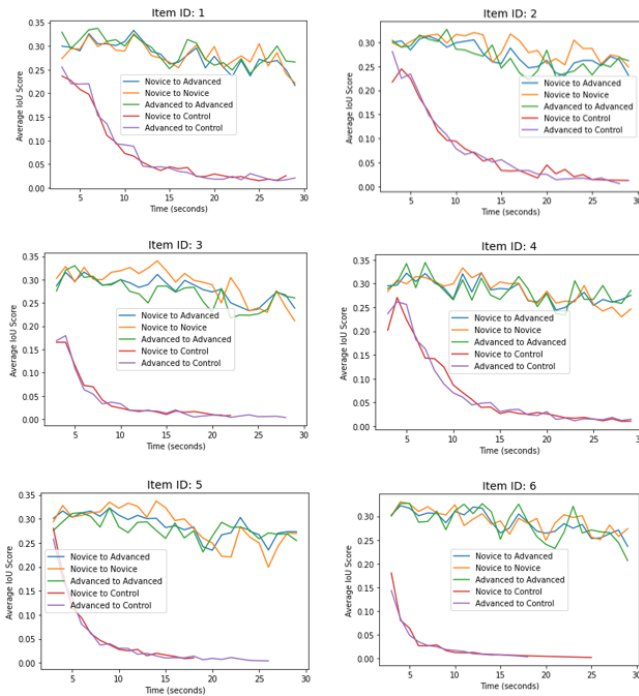


Figure 7: IoU Average Score Matrix over Time

The maximum average IoU scores of the participant groups were no more than 0.35 and did not meet a “decent” similarity score. Despite the low similarity score, the average IoU scores for each art print were higher in earlier time marks and showed a downward trend within the duration of 30 seconds. Although, the average IoU scores oscillated and did not show a consistent decline. There was also no significant difference between the average IoU scores when compared against the same and opposite sample groups. The average IoU score of participants with formal background in art were neither consistently better nor worse than participants with no formal background in art.

The average IoU scores of the controlled dataset based on the expected gaze clustering, defined with the consultant art expert, were also computed against the participant groups gaze clustering. The average IoU scores of “Novice to Control” and “Advanced to Control” were both lower than the average IoU scores of the sample groups and did not meet a “decent” similarity score. Like the comparison of average IoU scores of sample groups, the average IoU scores, when compared against the controlled dataset, were higher in the earlier time marks and showed a downward trend within the duration of 30 seconds. There was least oscillation observed when comparing against the average IoU scores with the controlled data set.

To quantify whether there exists a statistically significant difference between the mean values of the participant IoU scores per art print, a one-way ANOVA test was conducted. All art print items had an ANOVA test with p value < 0.05 except for art print item 5. This confirmed there was no significant difference of IoU scores of the sample groups. This presented both sample groups

performed comparably when measured against the control gaze clustering with low similarity scores. This confirmed non-adherence of the sample groups to the Gestalt grouping principle of proximity.

The tests also presented significant differences when comparing the IoU scores of participants in the within their sample groups and IoU scores of participants in different sample groups with their performance against the control. This suggests clustering performance measured between the same and opposite sample groups are better than when measuring the sample groups’ performances against the control.

5 CONCLUSION AND FUTURE WORKS

As discussed in Gestalt principles of visual perception, all humans can group similar elements through visual cues such as similarity, closure, and proximity. Criticisms of the Gestalt theory commonly discuss its potential lack of empirical research and focus on phenomenological methods [1, 13]. The result of this research helps in finding the differences of the effects of Gestalt Theory in specific cases and demographics especially in the field of arts. Given the level of subjectivity of visual perception in abstract art, this study can aid to confirm and provide a new direction to what is currently discussed in Gestalt research. This study has provided empirical evidence to show the presence of the Gestalt principle of proximity when viewing specific abstract arts, especially for non-art participants compared to participants with formal background in art.

The study leveraged gaze clustering analysis to measure the effects of the Gestalt principle of proximity between participants with advance or formal art, and novice or non-art backgrounds. Specifically, the gaze clustering algorithm explored was OPTICS to analyze the effects of proximity to participant gazes. Due to the variance in gaze point counts to measure gaze clustering similarity, convex hulls were calculated from the OPTICS clusters. The Intersection over Union (IoU) scores of the participants’ gaze convex hulls and the expected clusters’ convex hulls were studied to measure adherence to the Gestalt principle of proximity. To further confirm, scanpath pairwise distance analysis was performed to measure the similarity of scanpaths and validate the effects of participant art expertise.

Both novice and advance participant gaze clustering had low similarity scores against the expected gaze clusters. However, both novice and advanced similarity scores present likely equal performance to their adherence to the effects of the Gestalt Principle of Proximity. With respect to the effects of expertise, within group variances show likely equal similarity scores. The post-test interview indicated comparable perceived objects and observed clustering of the participants, especially for art print item 3, which yielded potential correlation to the similarity of the gaze clustering performance of the sample groups. Overall, the results presented limited effects of expertise when viewing abstract art but had low adherence to the Gestalt principle of proximity.

Through the proposed gaze clustering analysis methodology, the study invites further quantitative research on the effects of Gestalt principle of proximity in visual perception in arts and across various fields. Further investigation is needed to validate two potential directions of research: (1) considering use of non-abstract or non-art stimuli to measure the effects of the Gestalt principle of proximity and level of expertise in a domain relative to the chosen stimuli, and (2) evaluating alternative clustering algorithms to measure the effects of Gestalt principle of proximity. The findings in this study were specific in the domain of abstract art and limited to remote facilitation of eye tracking experiment brought by the effects of the COVID 19 pandemic. More accurate eye tracking instruments and validation through use of other abstract art as stimuli would be recommended to refine the findings of the study.

ACKNOWLEDGMENTS

The authors would like to thank the Engineering Research and Development for Technology (ERDT) for the scholarship grant to aid the study. Thank you to Ateneo de Manila University Fine Arts consultants who helped in facilitating data collection for the art prints. Lastly, thank you to the Ateneo de Manila Department of Information Systems and Computer Science for their support for student activities.

REFERENCES

- [1] James J. DiCarlo, Davide Zoccolan, and Nicole C. Rust. 2012. How does the brain solve visual object recognition? *Neuron* 73, 3 (February 2012), 415–434. DOI:https://doi.org/10.1016/j.neuron.2012.01.010
- [2] Richard L. Gregory. 2015. *Eye and Brain: The Psychology of Seeing*. Princeton University Press. Retrieved from <https://play.google.com/store/books/details?id=MYgVBgAAQBAJ>
- [3] Frank Jäkel, Manish Singh, Felix A. Wichmann, and Michael H. Herzog. 2016. An overview of quantitative approaches in Gestalt perception. *Vision Res.* 126, (September 2016), 3–8. DOI:https://doi.org/10.1016/j.visres.2016.06.004
- [4] Rodrigo Quiroga and Carlos Pedreira. 2011. How do we see art: an eye-tracker study. *Front. Hum. Neurosci.* 5, (September 2011), 98. DOI:https://doi.org/10.3389/fnhum.2011.00098
- [5] S. Ramachandran. The science of art: a neurological theory of aesthetic experience. Retrieved from https://books.google.com/books/about/The_science_of_art.html?hl=&id=eFpbswEACAAJ
- [6] Johan Wagemans, James H. Elder, Michael Kubovy, Stephen E. Palmer, Mary A. Peterson, Manish Singh, and Rüdiger von der Heydt. 2012. A century of Gestalt psychology in visual perception: I. Perceptual grouping and figure-ground organization. *Psychol. Bull.* 138, 6 (November 2012), 1172–1217. DOI:https://doi.org/10.1037/a0029333
- [7] Johan Wagemans, Jacob Feldman, Sergei Gepshtein, Ruth Kimchi, James R. Pomerantz, Peter A. van der Helm, and Cees van Leeuwen. 2012. A century of Gestalt psychology in visual perception: II. Conceptual and theoretical foundations. *Psychol. Bull.* 138, 6 (November 2012), 1218–1252. DOI:https://doi.org/10.1037/a0029334
- [8] Semir Zeki. 2017. Art and the Brain. *The Brain*, 71–103. DOI:https://doi.org/10.4324/9781351305204-3
- [9] Abraham S. Luchins. 1951. An evaluation of some current criticisms of Gestalt psychological work on perception. *Psychological Review* 58, 2 (1951), 69–95. DOI:https://doi.org/10.1037/h0053552
- [10] Zena O'Connor. 2015. Colour, contrast and gestalt theories of perception: The impact in contemporary visual communications design. *Color Research & Application* 40, 85–92. DOI:https://doi.org/10.1002/col.21858
- [11] Alan Gilchrist. 2012. Objective and Subjective Sides of Perception. *Visual Experience* (December 2012), 105–121. DOI:https://doi.org/10.1093/acprof:oso/9780199597277.003.0006
- [12] Aaron Kozbelt. 2001. Artists as experts in visual cognition. *Visual Cognition* 8, 6 (2001), 705–723. DOI:https://doi.org/10.1080/13506280042000090
- [13] Markus Vincze, Sven Wachsmuth, and Gerhard Sagerer. 2014. Perception and computer vision. In K. Frankish & W. Ramsey (Eds.), *The Cambridge Handbook of Artificial Intelligence* (pp. 168–190). Cambridge: Cambridge University Press. doi:10.1017/CBO9781139046855.012
- [14] D.W. Hamlyn. 2017. *The Psychology of Perception*. (2017). DOI:https://dx.doi.org/10.4324/9781315473291
- [15] Michela C. Tacca and Arnon Cahen. 2013. Linking Perception and Cognition. *Frontiers Research Topics* (2013). DOI:https://dx.doi.org/10.3389/978-2-88919-152-9
- [16] Desolneux Agnès, Lionel Moisan, and Jean-Michel Morel. 2011. *From Gestalt theory to image analysis: a probabilistic approach*. New York: Springer.
- [17] James R. Antes and Arlinda F. Kristjanson. 1991. Discriminating Artists from Nonartists by Their Eye-Fixation Patterns. *Perceptual and Motor Skills* 73, 3 (1991), 893–894. DOI:https://dx.doi.org/10.2466/pms.1991.73.3.893
- [18] Michael Kubovy, William Epstein and Sergei Gepshtein. 2013. *Visual Perception: Theoretical and Methodological Foundations*.
- [19] Naoko Koide, Takatomi Kubo, Satoshi Nishida, Tomohiro Shibata, and Kazushi Ikeda. 2015. Art Expertise Reduces Influence of Visual Saliency on Fixation in Viewing Abstract-Paintings. *Plos One* 10, 2 (June 2015). DOI:https://dx.doi.org/10.1371/journal.pone.0117696
- [20] Michael Kubovy. 1981. *Perceptual Organization*. Hillsdale: Lawrence Erlbaum Assoc.
- [21] David G. Lowe. 1986. *Perceptual organization and visual recognition*. Boston u.a.: Kluwer.
- [22] Oliver Braddick. 2015. Visual Perception, Neural Basis of. *International Encyclopedia of the Social & Behavioral Sciences* (2015), 184–190. DOI:https://dx.doi.org/10.1016/b978-0-08-097086-8.56018-8
- [23] Ian E. Gordon. 2004. *Theories of visual perception*. Hove: Psychology Press.
- [24] Nicola C. Anderson, Fraser Anderson, Alan Kingstone, and Walter F. Bischof. 2014. A comparison of scanpath comparison methods. *Behavior Research Methods* 47, 4 (2014), 1377–1392. DOI:https://dx.doi.org/10.3758/s13428-014-0550-3
- [25] Sebastiaan Mathôt, Filipe Cristino, Iain Gilchrist, and Jan Theeuwes. 2012. A simple way to estimate similarity between pairs of eye movement sequences. *Journal of Eye Movement Research* 5, 1 (January 2012), 1–15.
- [26] Quintana-Nevárez, Alberto & López-Orozco, Francisco & Florencia, Rogelio. 2017. Biometric authentication based on eye movements by using scan-path comparison algorithms. (2017)
- [27] Norman H. Mackworth and Anthony J. Morandi. 1967. The gaze selects informative details within pictures. *Perception & Psychophysics* 2, 547–552. DOI:https://doi.org/10.3758/bf03210264
- [28] Alfred L. Yarbus. 1967. *Eye movements and vision*. New York: Plenum.
- [29] Jasmina Stevanov. 2016. Huston, J. P., Nadal, M., Mora, F., Agnati, L. F., & Cela-Conde, C. J. (Eds.). *Art, Aesthetics and the Brain*. *Perception* 45, 10 (October 2016), 1203–1208. DOI:https://doi.org/10.1177/0301006616658823
- [30] Khushnood Naqshbandi, Tom Gedeon, and Umran Azziz Abdulla. 2016. Automatic clustering of eye gaze data for machine learning. 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC). DOI:https://doi.org/10.1109/smc.2016.7844411
- [31] Patrick Cavanagh. 2005. The artist as neuroscientist. *Nature* 434, 7031 (2005), 301–307. DOI:https://doi.org/10.1038/434301a
- [32] Rudolf Arnheim. 1947. Perceptual abstraction and art. *Psychological Review* 54, 2 (1947), 66–82. DOI:https://doi.org/10.1037/h0056797
- [33] Robert Zimmer. 2003. Abstraction in art with implications for perception. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 358(1435), 1285–1291. doi:https://doi.org/10.1098/rstb.2003.1307



Author Index

- | | | |
|------------------------|------------------------|----------------------|
| Abejuela, N., 77 | Gonzales, M.E.M., 46 | Olivarez, S., 115 |
| Adorna, H.N., 139, 153 | Herradura, T.R., 25 | Omalin, J.G., 115 |
| Alva, G.P.N., 115 | Ibrahim, H.O.A., 46 | Ong, C., 123 |
| Asuero, M.L.E., 11 | Kao, H.Y., 99 | Ong, E., 85 |
| Aya-ay, R.A., 93 | Labao, A.B., 139, 153 | Ong, E.B.H., 46 |
| Badiang, R., 107 | Laguer, J.J.R., 3 | Opulencia, L.A., 123 |
| Barreiro, J.A., 54 | Lanting, J.T., 123 | Palacios, N.I., 107 |
| Bernardino, M.G., 54 | Lim, R.T., 93 | Parcasio, J.B., 107 |
| Cabatuan, M., 77 | Lim, S.V.D., 145 | Pornillos, J.A., 107 |
| Casano, J.A., 3 | Limpag, H.M.L., 70 | Pugoy, R.A., 99 |
| Canillo, A.C., 93 | Logroño, B., 153 | Ramos, K.A.L., 145 |
| Choa, C., 153 | Lopez, E.G., 54 | Rodrigo, M.M.T., 159 |
| Chu, S.B., 77, 145 | Luteria, J.L., 139 | Roque, M., 77 |
| Cruz, J.Z., 131 | Makiling, J.C., 11 | Shrestha, A.M.S., 62 |
| Datario, Y.A.T., 123 | Malitao, B.R., 40 | Tadeo, N., 131 |
| Dela Cruz, J.B., 123 | Manalastas, P.R., 19 | Tamano, J.R., 131 |
| Dizon, F.S.V., 62 | Manseras, A.C., 107 | Tan, E.J.L., 145 |
| Encabo, J.N., 25 | Nazario, M.E.K.B., 145 | Tiam-Lee, T.J., 115 |
| Fernandez, R.A., 46 | Nemeño, I.J., 115 | Tiangco, A., 139 |
| Ferrer, C.R.M., 159 | Nogra, E.W., 139 | Ureta, J.C., 62 |
| Fong, H., 85 | Ng, M., 77 | Valencia, J., 77 |
| Gamot, R.M.T., 70 | Nolasco, H., 77 | Vergara, J.P., 54 |
| Garcia, A.F.A., 11 | Obispo, J.R.C., 11 | Villegas, M., 32 |
| Gayao, Y.C., 40 | Ocampo, C.M.R., 40 | Villos, R., 153 |
| Geronimo, J.C., 123 | Ogawa, J., 115 | Vitug, M.C.B., 25 |



Institution Index

Ateneo de Manila University, 3, 19, 32, 54, 131, 159
De La Salle University – Dasmariñas, 25
De La Salle University Integrated School, 145
De La Salle University, Manila, 46, 62, 77, 85, 115, 123, 145
National Cheng Kung University, 99
University of the Cordilleras, 40
University of the Immaculate Conception, 107
University of San Carlos, 93
University of the Philippines Diliman, 139, 153
University of the Philippines Mindanao, 70
University of the Philippines Open University, 99
Xavier University – Ateneo de Cagayan, 11