Towards a Memory-Efficient Filipino Sign Language Recognition Model for Low-Resource Devices

Shuan Noel Co
De La Salle University
Manila, Metro Manila
shuan_co@dlsu.edu.ph

Darius Ardales
De La Salle University
Manila, Metro Manila
darius_ardales@dlsu.edu.ph

Miguel Gonzales
De La Salle University
Manila, Metro Manila
miguel_gonzales@dlsu.edu.ph

Stephanie Joy Suzada
De La Salle University
Manila, Metro Manila
stephanie_susada@dlsu.edu.ph

Waynes Weyner Wu
De La Salle University
Manila, Metro Manila
waynes_wu@dlsu.edu.ph

Thomas James Tiam-Lee
De La Salle University
Manila, Metro Manila
thomas.tiam-lee@dlsu.edu.ph

Ann Franchesca Laguna
De La Salle University
Manila, Metro Manila
ann.laguna@dlsu.edu.ph

ABSTRACT

In this paper, we present a preliminary LSTM-based model for recognizing Filipino sign language words in videos using hand landmarks extracted from MediaPipe. Furthermore, we show that post-quantization can significantly reduce its size without sacrificing its performance, showing the potential for practical use. Despite being trained on only a small number of instances per class, results show that the model was able to achieve an accuracy of 93.29%, while a 90% reduction in model size.

KEYWORDS

Filipino sign language, sign language recognition, tinyML

1 INTRODUCTION

Nowadays, the world is becoming more interconnected and inclusive. Bridging the communication gap for all groups of people has emerged as a pressing goal for researchers and practitioners alike. In the Philippines, the deaf community represents a significant part of society that faces challenges in communication, often leading to discrimination and marginalization [7, 18, 19]. The Philippine Statistics Authority (PSA) estimates that there are 1,784,690 individuals with hearing difficulty in 2020, comprising around 1.6% of the population [14]. In 2018, the Philippine government signed into law Republic Act No. 11106, also known as the "Filipino Sign Language Act", which designates Filipino Sign Language as the national sign language of the Filipino deaf, mandating its use in schools, workplaces, and broadcast media [2].

First, we develop a preliminary LSTM-based deep neural network for recognizing a small subset of words that are specifically unique to FSL. We show that training a model for this small subset is possible with only a small size of training data. Second, we show that quantization can be used to substantially reduce the size of the model without sacrificing its performance.

This paper is structured as follows. Section 2 discusses the related literature. Section 3 discusses the methodology we used for developing the sign language recognition model. Section 4 discusses the evaluation of the model and the results of the model. Finally, Section 5 provides conclusions and directions for future work.

2 RELATED WORKS

This section discusses the related literature of this study and situates the position of this work in the existing body of knowledge.

2.1 Sign Language Recognition

There has been a wealth of studies done on sign language recognition. For a period, hidden Markov models (HMM) and recurrent neural networks (RNN) were the most common approaches to classify sign language [15]. However, with the recent developments in machine learning, most deep learning approaches such as CNNs and LSTMs have shown superior performance in this domain [16]. Despite its seemingly straightforward presentation, the problem of sign language recognition is a complex task with many considerations and challenges.

While most studies focus on the hand gestures only, there are studies that focus on recognizing the body gestures [9] and facial expressions [5, 17] as well. Another challenge in sign language recognition is that some of these features may be occluded at certain points in time [16]. One way to alleviate this is to perform feature fusion, considering all the features in the prediction to make the model more robust to such cases [8].

2.2 Filipino Sign Language Recognition

One major challenge is the small number of datasets available for FSL. The largest dataset to our knowledge for FSL is the FSL-105 dataset, which contains around 20 labelled instances for 105 introductory words and phrases [21]. While helpful, it does not compare to the amount of data available for other sign languages.

In recent years, a few researchers have made attempts to incorporate various sign language recognition approaches to FSL. In the works of [3, 4], a model for recognizing letters of the alphabet was developed. In the former, it was successfully deployed in a Raspberry Pi, achieving an accuracy of 93.29%. However, these works

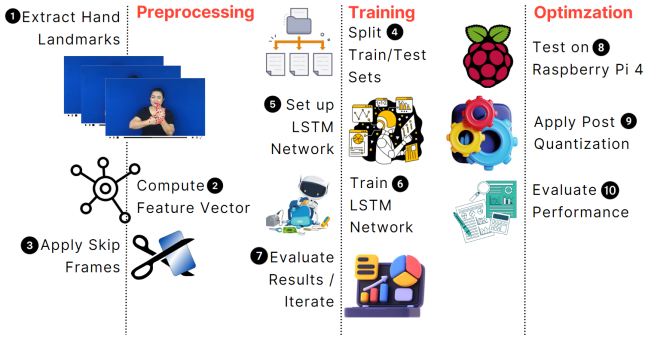


Figure 1: ML Pipeline

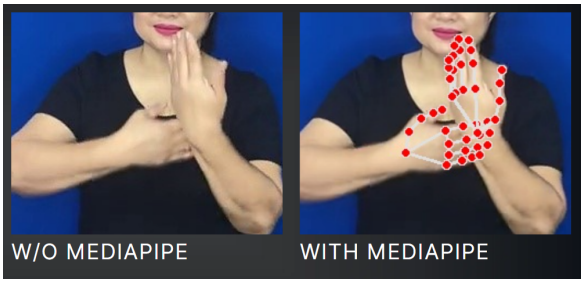
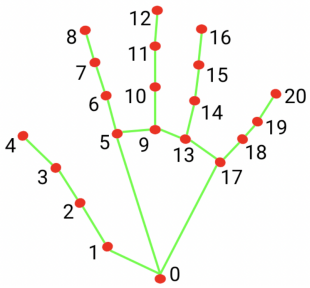


Figure 2: Before and After MediaPipe Processing

Table 1: List of Features Considered for the Model



are limited to alphabet signs only, which are all static gestures that do not incorporate movement. Similarly, the work of [12] trained a CNN model for static images showing numbers. Other works have applied various deep learning approaches to the task of FSL recognition, such as LSTM [6, 13], ResNet [13], and gated recurrent units [20], with good results on low label count manually collected datasets.

3 METHODOLOGY

This section discusses the methodology we used in training the sign language recognition model for FSL.

3.1 Dataset

In this study, we used data from the FSL-105 dataset. The FSL-105 dataset is a labelled dataset comprising 105 introductory words and phrases in FSL [21]. Each instance in the dataset contains the word or phrase, and a video of a person showing the sign language movement corresponding to that word. In this study, we only considered four words: “bread”, “egg”, “chicken”, and “crab”. These words were chosen because they have unique signs in FSL compared to other sign languages, and they are commonly used words. Each word has a total of 20 instances.

3.2 Training Pipeline

Figure 1 shows the pipeline for the training process. The process can be divided into three main phases. In the preprocessing phase, the videos are converted as a sequence of frames (images), which are then pre-processed using computer vision tools to extract key features in preparation for training. In the training phase, an LSTM deep neural network is trained from the input features. Finally, in the optimization phase, a post-quantization method is applied to the resulting model to reduce its size while maintaining its performance.

3.3 Preprocessing

In this phase, we perform preprocessing steps on the data in preparation for training. First, we extracted the instances belonging to the four target classes from the FSL-105 dataset. Next, we preprocess each video to extract the desired features for training.

Each video can be represented as a sequence of frames or images. For each frame, we extract key landmarks showing the inferred position and orientation of the hand on the frame. First, OpenCV was used to extract the image data from the individual frames of the video, then the image data was fed to MediaPipe to extract the landmarks. This preprocessing allows us to isolate the hands from the other parts of the video to eliminate background noise. This helps the model focus on the relevant information. For example, factors such as the skin color of the person doing the gesture can be ignored as they are not relevant to the task.

Our preprocessing and training process closely follows the framework outlined by [10]. Figure 2 shows some examples of video frames and the corresponding landmarks extracted by MediaPipe [11]. MediaPipe extracts 21 landmarks representing key joints in a person’s hand. Each landmark is represented as a normalized point in 3-D space with three numerical values corresponding to the x, y, and z components. From these raw landmarks, we compute a feature vector by engineering a set of features that are more meaningful for gesture representation. Specifically, we compute the distances between a set of landmark pairs. Table 1 shows the distances that were computed for each hand.

Each instance is represented as a sequence of frames, where each frame is embedded as the feature vectors, we only considered the frames where the hand has been detected by MediaPipe. We also applied a skip frame operation to standardize all sequences to 10 frames.

3.4 Training

We posed the sign language recognition problem as a problem of sequential gesture recognition. While there are certain words and phrases that don’t require much movement or changes in the gesture, there are also words and phrases that rely on the movement

of the hand gesture over time. To accommodate this, we chose LSTM as the model architecture for training the sign language recognition model. The "sequence" pertains to the sequence of embeddings per frame of a single sign language gesture instance.

LSTM (Long Short-Term Memory) neural networks excel in capturing long-range dependencies within sequential data. Unlike simple neural networks that struggle with learning relationships over extended sequences due to vanishing or exploding gradient problems, LSTMs are specifically designed to address this issue. The key innovation lies in their gated architecture, allowing the network to selectively remember or forget information over time. Each LSTM unit possesses a memory cell that serves as a persistent storage, and three gates (input, forget, and output) regulate the flow of information. The input gate controls which information to update, the forget gate decides what to discard from the cell's memory, and the output gate determines the information to be passed to the next layer. This intricate mechanism enables LSTMs to capture nuanced temporal dependencies.

We split our dataset into a training and test set with an 80-20 split. This resulted to 64 instances for training and 16 instances for testing. We then define the architecture of our model based on [10], defined as follows. In order: (1) an LSTM layer with 256 neurons, (2) a dropout layer, (3) another LSTM layer with 256 neurons, (4) another dropout layer, (5) an LSTM layer with 128 neurons, (6) a dense layer, (7) a batch normalization layer, (8) a RelU activation function, and finally (9) an output layer with 4 output neurons and softmax activation function. We used categorical cross entropy as the loss function, a decaying learning rate starting from 0.001, and ADAM as the optimizing algorithm. We trained the model for 300 epochs. Finally, we test the performance of the model by evaluating its predictions of the test set. The model training was implemented through TensorFlow. [1].

3.5 Optimization

After training, we used optimization techniques to reduce the memory requirements of the resulting model. The main technique we used to optimize the model was post-quantization. Post-quantization is a technique that reduces the precision of numerical representations such as the weights and activations of the neurons from a floating point to a lower-bit fixed-point of numbers, thereby compressing the model and reducing its memory storage requirement and computational complexity. We also converted the model from TensorFlow to TensorFlow Lite, which streamlined the deployment process for lower-end devices like mobile devices and resource constrained environments.

4 RESULTS AND FINDINGS

This section discusses the performance of the resulting sign language recognition model and compares it against alternative approaches.

4.1 Performance of the Model

Even prior to quantization, the LSTM model trained on the hand landmarks achieved a 100% precision, recall, and accuracy on the validation data. Figure 2 shows the confusion matrix of the predictions, while Figure 3 shows the training and validation set accuracy throughout the training process. The optimal accuracy was already achieved in less than 50 epochs of training.

Table 2: Confusion Matrix for Model Performance

Actual\Pred	Egg	Chicken	Crab	Bread
Egg	3	0	0	0
Chicken	0	5	0	0
Crab	0	0	2	0
Bread	0	0	0	6

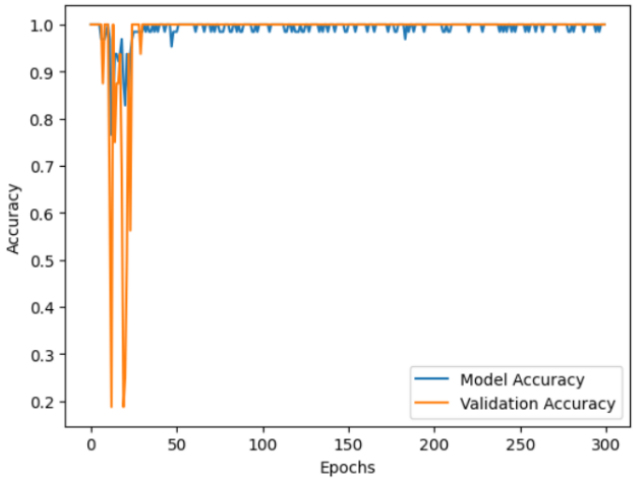


Figure 3: Train and Validation Accuracy in Model Training Process

4.2 Effect of Post-Quantization

After performing post-quantization, we were able to reduce the model size from 11.68MB to 1.01MB, corresponding to a 91.35% reduction in size. Despite this substantial reduction, the model maintained a high accuracy of 93.75%, with only one misclassification in the validation set. Table 3 shows the confusion matrix of the model after post-quantization. These results show the immense potential of post-quantization in the context of FSL recognition.

Table 3: Confusion Matrix for Model Performance After Post-Quantization

Actual\Pred	Egg	Chicken	Crab	Bread
Egg	1	0	0	0
Chicken	0	4	0	1
Crab	0	0	8	0
Bread	0	0	0	2

When testing the model, significant improvements in usability can be observed. For instance, prior to quantization the model would take a while to load when run on lower end systems. It would also

suffer from delays when deployed as a real-time application. However, post-quantization significantly boosted performance speeds, decreased loading times, and removed delays.

Nonetheless, there are still some problems when deploying the model on microcontrollers like Raspberry Pi due to the bottleneck of the MediaPipe processing. This causes performance issues when used in a real-time setup. Nonetheless, the model can run efficiently when used in an offline setting, or if the hand landmarks can be pre-processed. These results show that there are still issues to be resolved before the technology can be deployed in a real-world setting.

4.3 Comparison with Alternative Approaches

To highlight the advantages of the approach discussed in this paper, we compared the performance of the model against more straightforward approaches.

4.3.1 CNN without MediaPipe. The poor performance can of course be attributed to the fact that motion information was not being considered in this condition. CNN does not consider past movements or gestures, it may have difficulties in differentiating the classes, especially given the lack of data.

4.3.2 LSTM Without MediaPipe. We also attempted to train an LSTM model but without using MediaPipe by feeding in the individual frames of the RGB video sequence. For this model, the accuracy improved to 31.25%. Upon closer inspection of the confusion matrix in Table 4, it becomes clear why this is the case – all the validation instances were being predicted under "crab".

Table 4: Confusion Matrix for LSTM Performance Without MediaPipe

Actual\Pred	Egg	Chicken	Crab	Bread
Egg	0	0	5	0
Chicken	0	0	5	0
Crab	0	0	5	0
Bread	0	0	1	0

These results show that while LSTM can theoretically handle image sequence data, the use of raw RGB frames as LSTM input is not enough to successfully train an effective model with such a small dataset. Furthermore, the resulting size of the LSTM model is large at 1.73GB. These results highlight the advantage of adding a pre-processsing step to detect hand landmarks, as it significantly reduces the input size of the model, allowing for faster learning and smaller model size.

4.4 Summary of Results

Table 5 shows the summary of the results. From here, we can see that the introduced model for sign language recognition achieved good results on FSL by capturing hand movements as well as limiting the feature space to only the hand landmarks. Furthermore, post-quantization was able to significantly reduce the model size, showing potential for it to be deployed on low-resource devices.

Table 5: Summary of Results

Model	Input	Quantization	Accuracy	Model Size
CNN	Raw single video frame	no	25.49%	42MB
LSTM	Raw video sequence of frames	no	31.25%	1.73GB
LSTM	**MediaPipe hand landmarks on sequence of frames**	**no**	**100%**	**11.68MB**
LSTM	**MediaPipe hand landmarks on sequence of frames**	**yes**	**93.75%**	**1.01MB**

5 CONCLUSION AND FUTURE WORK

There is still a lot of future work in the field of FSL recognition. First, the models can be scaled up to handle a wider set of vocabulary. In this aspect, it would be interesting to explore whether current models would struggle with a larger set of classes, some of which may have similarities with one another. In this regard, one consideration is the development of techniques that do not require large amounts of data, as FSL datasets are currently limited. Second, the facial expressions and body gestures can be incorporated into the models, handling challenges such as occlusions to improve performance. Third, we can explore optimization techniques such as post-quantization for the development of real-time FSL recognition systems that can work on smartphones or similar devices so that they could be democratized to the Philippine deaf community. We believe these results can serve as a foundation for more FSL research and pave the way for the development of larger-scale recognition systems for the language.

REFERENCES

[1] [n. d.]. TensorFlow. https://www.tensorflow.org

[2] 2018. Republic Act No. 11106. https://www.officialgazette.gov.ph/2018/10/30/republic-act-no-11106/

[3] Mark Christian Ang, Karl Richmond C Taguibao, and Cyrel O Manlises. 2022. Hand Gesture Recognition for Filipino Sign Language Under Different Backgrounds. In *2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*. IEEE, 1–6.

[4] Mark Allen Cabutaje, Kenneth Ang Brondial, Alyssa Franchesca Obillo, Mideth Abisado, Shekinah Lor Huyo-a, and Gabriel Avelino Sampedro. 2023. Ano Raw: A Deep Learning Based Approach to Transliterating the Filipino Sign Language. In *2023 International Conference on Electronics, Information, and Communication (ICEIC)*. IEEE, 1–6.

[5] Siddhartha Pratim Das, Anjan Kumar Talukdar, and Kandarpa Kumar Sarma. 2015. Sign language recognition using facial expression. *Procedia Computer Science* 58 (2015), 210–216.

[6] Carmela Louise L Evangelista, Criss Jericho R Geli, Marc Marion V Castillo, and Carol Biklin G Macabagdal. 2023. Long Short-Term Memory-based Static and Dynamic Filipino Sign Language Recognition. In *2023 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*. IEEE, 235–240.

[7] Jasmine DC Hidalgo, Kayla Joy R Pantanilla, Almira E Castro, and Mickaela R Alfon. 2023. Employability of Persons With Disabilities. *International Journal of Academic Management Science Research* 7, 4 (2023), 29–36.

[8] Muhammed Kocabas, Salih Karagoz, and Emre Akbas. 2018. Multiposenet: Fast multi-person pose estimation using pose residual network. In *Proceedings of the European conference on computer vision (ECCV)*. 417–433.

[9] Dimitrios Konstantinidis, Kosmas Dimitropoulos, and Petros Daras. 2018. Sign language recognition based on hand and body skeletal data. In *2018-3DTV-conference: The true vision-capture, transmission and display of 3D video (3DTV-Con)*. IEEE, 1–4.

[10] Wee Kiat Lim. 2021. Hand Gesture Detection and Sequence Recognition. https://weekiat-lim.medium.com/hand-gesture-detection-sequence-recognition-7f3215f88dde.

[11] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Yong, Juhyun Lee, et al. 2019. Mediapipe: A framework for perceiving and processing reality. In *Third workshop on computer vision for AR/VR at IEEE computer vision and pattern recognition (CVPR)*, Vol. 2019.

[12] Myron Darrel Montefalcon, Jay Rhald Padilla, and Ramon Llabanes Rodriguez. 2021. Filipino sign language recognition using deep learning. In *2021 5th International Conference on E-Society, E-Education and E-Technology*. 219–225.

[13] Myron Darrel Montefalcon, Jay Rhald Padilla, and Ramon Rodriguez. 2022. Filipino sign language recognition using long short-term memory and residual network architecture. In *Proceedings of Seventh International Congress on Information and Communication Technology: ICICT 2022, London, Volume 4*. Springer, 489–497.

[14] Liberty Notarte-Balanquit. 2023. Filipino Sign Language: Filipino Sign Language Numerals and the Expansion of Deaf Linguistic Repertoire (online lecture). https://www.youtube.com/watch?v=4vBEN00ecGw

[15] Sylvie CW Ong and Surendra Ranganath. 2005. Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 27, 06 (2005), 873–891.

[16] Razieh Rastgoo, Kourosh Kiani, and Sergio Escalera. 2021. Sign language recognition: A deep survey. *Expert Systems with Applications* 164 (2021), 113794.

[17] Joanna Pauline Rivera and Clement Ong. 2018. Facial expression recognition in filipino sign language: Classification using 3D Animation units. In *Proc. the 18th Philippine Computing Science Congress (PCSC 2018)*. 1–8.

[18] Freya Silva-Dela Cruz and Estrella Calimpusan. 2018. Status and challenges of the deaf in one city in the philippines: towards the development of support systems and socio-economic opportunities. *Asia Pacific Journal of Multidisciplinary Research* 6, 2 (2018), 33–47.

[19] Marcella L Sintos. 2020. Psychological Distress of Filipino Deaf: Role of Environmental Vulnerabilities, Self-Efficacy, and Perceived Functional Social Support. *Asia-Pacific Social Science Review* 20, 3 (2020).

[20] Isaiah Tupal, Melvin Cabatuan, and Michael Manguerra. 2022. Recognizing Filipino Sign Language with InceptionV3, LSTM, and GRU. In *2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*. IEEE, 1–5.

[21] Isaiah Jassen Lizaso Tupal and Cabatuan K Melvin. [n. d.]. FSL105: The Video Filipino Sign Language Sign Database of Introductory 105 FSL Signs. *Available at SSRN 4476867* ([n. d.]).