SIGN ME UP: AN ANDROID SIGN LANGUAGE TRANSLATOR USING A CONVOLUTIONAL NEURAL NETWORK

CARLOS MIGUEL EUROPEO CANONIZADO

SUBMITTED TO THE INSTITUTE OF COMPUTER SCIENCE

UNIVERSITY OF THE PHILIPPINES LOS BAÑOS

IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS

FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

MAY 2019

The Faculty of the Institute of Computer Science
of the University of the Philippines Los Baños
accepts the Special Problem entitled

SIGN ME UP: AN ANDROID SIGN LANGUAGE TRANSLATOR USING A CONVOLUTIONAL NEURAL NETWORK

CARLOS MIGUEL EUROPEO CANONIZADO

In partial fulfillment of the requirements for the

Degree of Bachelor of Science in Computer Science

____

PROF. JAIME SAMANIEGO

Adviser

____

Date Signed

____

PROF. JAIME SAMANIEGO

Director, Institute of Computer Science

____

Date Signed

SIGN ME UP: AN ANDROID SIGN LANGUAGE TRANSLATOR USING A CONVOLUTIONAL NEURAL NETWORK

ABSTRACT

CARLOS MIGUEL EUROPEO CANONIZADO. 2019. SIGN ME UP: An Android Sign Language Translator Using A Convolutional Neural Network. University of the Philippines Los Baños, College of Arts and Sciences, Institute of Computer Science.

Sign language, although widely used by both the deaf and hearing community, is still a challenge for other people. SIGN ME UP is an Android application which is designed to translate sign language even offline. It uses a CNN for translating sign language to text and a series of reference images for translating text to sign language.

Through retraining the Inception-v3 model, it gained the highest and lowest confidence levels of 97.82% and 27.11% respectively, affected by similar gestures. An average of 60.03% confidence level was obtained for all gestures while averages of 81.94% and 97.42% for the top-1 and top-5 accuracies were obtained.

ACKNOWLEDGEMENTS

There are a lot of people I am very grateful for. First, I would like to thank my parents, Drew Europeo and Nigel Canonizado, for consistently supporting me since day one. I know they had doubts especially when I was not yet in BS Computer Science but all their encouragements helped me gain my confidence of pushing through all the problems I faced.

I would also like to thank my two advisers during my stay as an undergraduate. First is Sir Romel Daya when I was a freshman in the College of Development Communication, he was there to support me and my plans. Sir Romel is like a brother to me. Then there is Sir Jimmy Samaniego, who was also very supportive. He helped me whenever I had problems enrolling in CMSC 56 during the midyear when I was not yet a BS Computer Science student. Sir Jimmy was always there when I needed guidance with my SP and he always has good comments and suggestions. I would not have done it without them.

To my organization, the Young Software Engineers' Society, I thank you for molding me into the person that I am today. It was a pleasure leading you all during my senior year and I loved every single part of my term. I hope the members keep loving and enjoying every single moment with the organization. Continue to inspire others and keep the fire burning.

To all the friends I have gained and lost, thank you. To my roommates and high school friends who are still in touch with me, I appreciate all the bonding moments we had. I wish that we keep in touch as we move on to another chapter in life.

Lastly, I would like to thank my other half, the one who is always there by my side. You helped me grow so much and college was a lot easier with you. Every problem was bearable because of you and you were there for all the ups and downs. I love you so much, you know who you are.

TABLE OF CONTENTS

I. INTRODUCTION

Sign language is a system of communication that has been around for years. Its primary usage is to enable communication between the deaf and hearing people. The World Health Organization (WHO) estimated that by 2050, more than 900 million people will have disabling hearing loss (WHO, 2018). Given this large estimation, it is problematic that only a small percentage of people actually know how to use sign language. One of the most widely used sign language is the American Sign Language (ASL). The ASL gained its popularity after William Stokoe, Dorothy Casterline, and Carl Croneberg published their Dictionary of American Sign Language on Linguistic Principles in 1965 (Wilcox, 1991).

A. Background of the Study

When it comes to motion gesture technology, the earliest implementation was made possible through the use of gloves with sensors. Depending on the hand movement, signals were decoded by a computer by mapping a combination of signals to unique gestures (Sharma, 2015). Today, technological advancements allow programmers to develop tools that deal with the likes of motion gesture recognition, without the use of any external equipment aside from cameras.

Another technological advancement which will make this study feasible is the existence of Artificial Intelligence (AI). AI is a broad topic and one of its use many uses is the implementation of neural networks. A neural network is a set of connected processors or neurons which can be activated to produce a desired idea or result (Schmidhuber, 2015). Neural networks have different types but for image-related tasks, a convolutional neural network (CNN) is widely used due to its advantages for image classification or categorization (Wu, 2016).

B. Objectives of the Study

This study aims to develop an Android application to translate sign language. Specifically, this study intends to fulfill the following:

1. translate user-captured sign language to text using a retrained convolutional neural network called Inception-v3 model;
2. translate user-inputted text to sign language using a series of image definitions; and
3. translate sign language regardless of the hand's orientation or size.

C. Significance of the Study

Being able to translate sign language to text or vice-versa will be beneficial not only for the deaf community but also for those who are aspiring to learn sign language. Moreover, the communication barrier between the deaf and non-deaf community will be minimized since they will have an accessible translator through this developed mobile application.

Those who are trying to learn sign language will find the text to sign language useful while those who just want to translate sign language can simply use the sign language to text feature. Both of these features do not require any internet connectivity.

In addition, most of the early sign language translators require the use of expensive devices such as smart gloves. Therefore, this application can be a cheaper alternative since smartphones are accessible and normal users can easily use the application without any additional setup.

II. REVIEW OF RELATED LITERATURE

For several decades, understanding the deaf community has become a major challenge. Sign language, despite its many variations, is one of the most effective methods that allow the better communication between the deaf and the non-deaf community. Several studies have been conducted to further allow translation of sign language without actually learning the language.

A. Previous Studies on Sign Language Translation

1. Support Vector Machine

One implementation was done with the help of a Support Vector Machine (SVM). An SVM is used for analyzing data through classification, regression, and outliers detection. Additionally, an SVM is a supervised machine learning algorithm that is reliant on plotted data points.

Villamor (2018) was able to utilize an SVM in sign language translation through the following process: first, given an input image, the background and hand are identified. After that, the features of the hand are detected, examples of these features are the center of the palm, finger defect points, and finger vectors. These data are then plotted in an n-dimensional space, where n is the number of features. Once the points are plotted, the image is classified by finding the difference of two sets of points.

2. Color Segmentation and Neural Networks

Another method that was used in translating sign language is color segmentation and neural networks. For this method, Akmeliawati (2007) used custom-made gloves to gather necessary data from the signer. Since there is a lot of unwanted data captured in an image or video, the frames were segmented according to the color that was visible in the gloves. This makes the translation easier since the program expects certain colors for each feature like yellow for the palm, purple for the thumb, red for the index and ring finger, and blue for the middle and pinky finger.

This set-up produces data which are called vectors. A vector is a collection of numbers and for each finger, there is one xyvector giving a total of five xyvectors per hand. The weights of the data from these vectors are then fed into a neural network, to be specific, an artificial neural network (ANN). The ANN that Akmeliawati used had three neural networks: one for recognizing the alphabet, another for numbers, and the third one for word signs. After traversing these layers, a value will be returned if ever the ANN correctly classifies the input gesture. The use of color segmentation and ANN allowed Akmeliawati to detect dynamic signs such as fingerspelling.

3. 3D Trajectory Matching with Kinect

Chai (2014) utilized Kinect by using its hand tracking technology. Through the hand tracking, a 3D trajectory was obtained which was then normalized through linear resampling. Normalization was done due to the differences in hand motion speed. After that, an existing gallery trajectory was used for the existing definitions and the input trajectory was aligned to get the recognition result.

B. Convolutional Neural Network

Convolutional neural networks (CNNs) are similar to traditional neural networks in the sense that they are comprised of neurons that are optimized through learning. Each node or neuron will still receive an input and perform activation functions. The main difference is that CNNs are primarily used for pattern recognition with images.

O'Shea (2015) states that one of the largest limitations of regular neural networks is the lack of computational power required to compute data from images. Handwritten digits can be viable in regular neural networks since the image dimensions could be as small as 28 x 28. However, if we take a larger colored image input of 64 x 64, then the number of neurons for the first layer increases to an overwhelmingly large number of 12,288.

With traditional neural networks, there is a minimum of three layers: the input, output, and hidden layers in between. In CNN, there are additional layers namely: convolutional layer, pooling layer, and fully-connected layers. The process of recognizing an image in CNN is done by passing an input image to the input layer. The input layer will hold the pixel values of the image. Next, the convolutional layer will compute for the output of the neurons. The output is then passed to the pooling layer which will reduce the number of activation parameters by downsampling through a rectified linear unit (ReLu). This allows images of higher resolutions to be processed. After that, the fully-connected layers will attempt to produce a probability classification of each class.

The classification of high resolution images using convolutional neural network was emphasized in a study conducted by Krizhevsky (2012). His deep convolutional network was trained to classify 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest. The model had a top-1 error rate of 37.5 percent and a top-5 error rate of 17 percent.

C. TensorFlow and Inception-v3

TensorFlow is an open source platform for machine learning created by Google. It provides a diverse selection of models to use for free, and one of them is the Inception-v3 model. This model is a convolutional neural network trained for the ImageNet Large Visual Recognition Challenge using data from 2012 (Szegedy, 2014).

The model produced a top-5 error rate of 3.46 percent. Its architecture can be seen below:

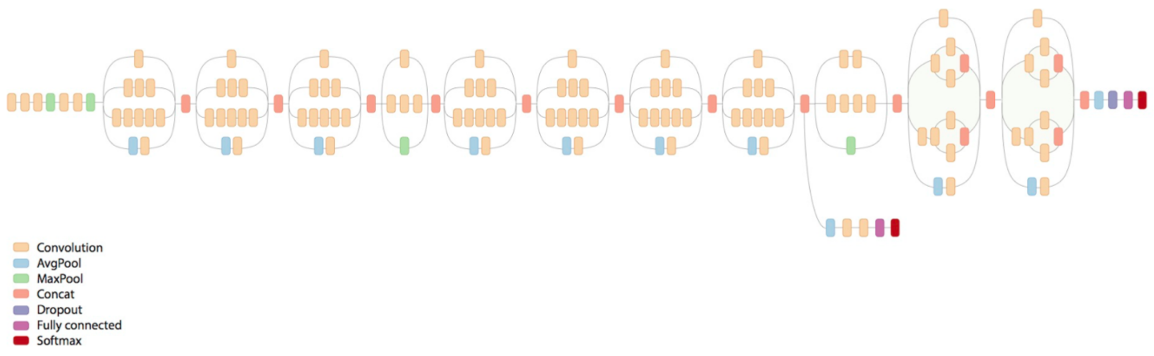


Figure 1. Inception-v3 Architecture

III. METHODOLOGY

A. System Requirements

The following were used used to develop the Android application:

1. A smartphone running on Android version 5.1.1 or higher for running the application;
2. Android Studio version 3.2.1 or higher for creating the application;
3. Python 3.7.1 for using scripts in model training;
4. Conda 4.5.12 for creating a virtual environment for TensorFlow; and
5. TensorFlow version 1.0 for the Inception-v3 retraining.

B. Android Application

The Android application is named “SIGN ME UP”. It has two functionalities: one for translating sign language to text, and another for translating text to sign language. The home screen along with the icon of the application can be seen in Figure 2.

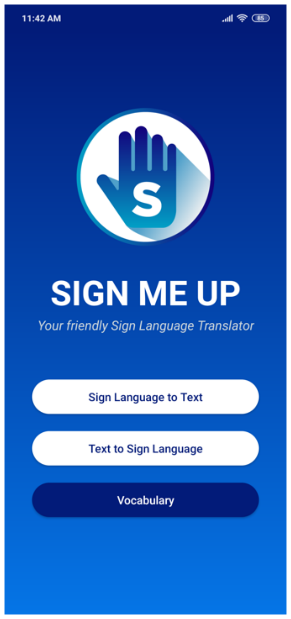
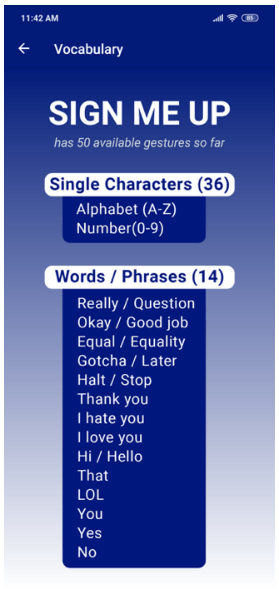


Figure 2. Home Screen

C. Vocabulary

A total of 50 definitions were used. However, all the definitions are only available in text to sign language and a subset was used for sign language to text. The vocabulary can be seen in Figure 3.



11:42 AM

Vocabulary

SIGN ME UP

has 50 available gestures so far

Single Characters (36)

Alphabet (A-Z)
Number(0-9)

Words / Phrases (14)

Really / Question
Okay / Good job
Equal / Equality
Gotcha / Later
Halt / Stop
Thank you
I hate you
I love you
Hi / Hello
That
LOL
You
Yes
No

Figure 3. Vocabulary Screen

D. Data Gathering

All images were gathered from random respondents in the University of the Philippines Los Baños. In sign language to text, only a subset of the vocabulary was used. These definitions are: the alphabet, yes, no, that, hi/hello, and okay/good job for a total of 31 gestures. Each image was also captured with a similar controlled background. Sample data is shown in Figure 4.

For every gesture, 40 people's hands were captured. The skin tone of the respondents were mostly fair to tan. All images were then flipped to cover the left and right-handed gestures, which added up to 80 pictures per gesture, with a total of 2,480 data.

The specifications of the camera used in data gathering can be found in Table I.

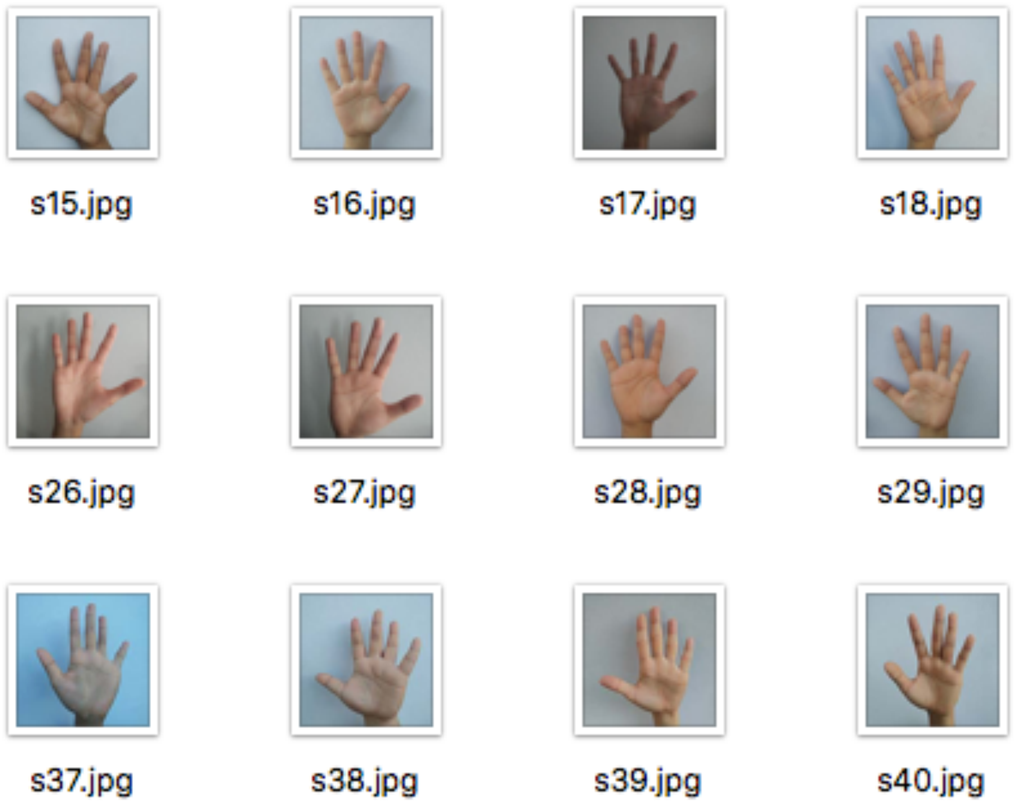


Figure 4. Sample Data

TABLE I: Camera Specifications

Model	OPPO A37
Camera	8 MP
Image Resolution	2448 x 2448 Pixels

E. Text to Sign Language

The definitions available were mapped to a single reference photo. If the gesture involves movement, then the picture has a silhouette of a moving hand.

Every input is checked to see if there are special characters. If there are any, the application will simply return an error message. Otherwise, the output will produce a series of images depending on the input of the user. It will first check if the input exists in the words available, if not, the words will be spelled out. Sample outputs can be seen in Figure 5 and Figure 6.

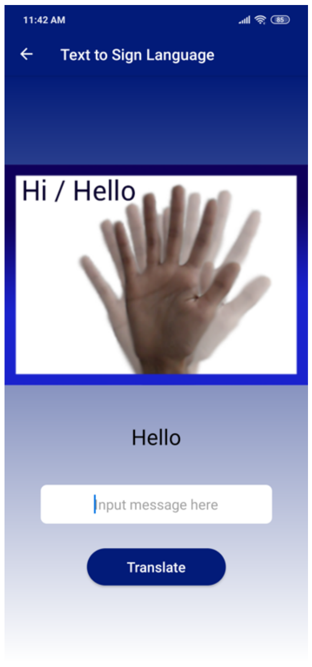


Figure 5. Sample Result

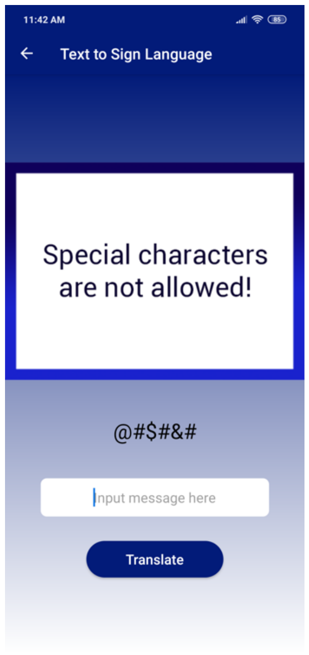


Figure 6. Invalid Input

F. Sign Language to Text

1. Retraining Inception-v3

In order to classify the gestures, a convolutional neural network was used. Specifically, the Inception-v3 model was retrained so that the labels produced by the model are the definitions available for translation.

This process is called transfer learning. From the old top/results layer of Inception-v3, a new layer was trained using the data gathered to fit the pretrained model. Out of the 2,480 data with 80 images per gesture, only 75 images per gesture were used for the training and validation. The remaining five images per gesture were used for testing.

For the training itself, a total of 8000 epochs or training steps with a learning rate of 1 percent was used.

2. Loading the Model in the Application

Before using the model, it was optimized to reduce the pre-processing of the application and to reduce the size of the package. But optimizing was not enough because the size of the model was too big and unsuitable for mobile phones. This issue was addressed by quantizing the optimized model.

Quantization was needed since majority of the space taken up by the model/output graph is from the weights or the large blocks of floating point numbers. Quantizing the weight of the networks helped in reducing the model's size. After this, the model was loaded in the Android application.

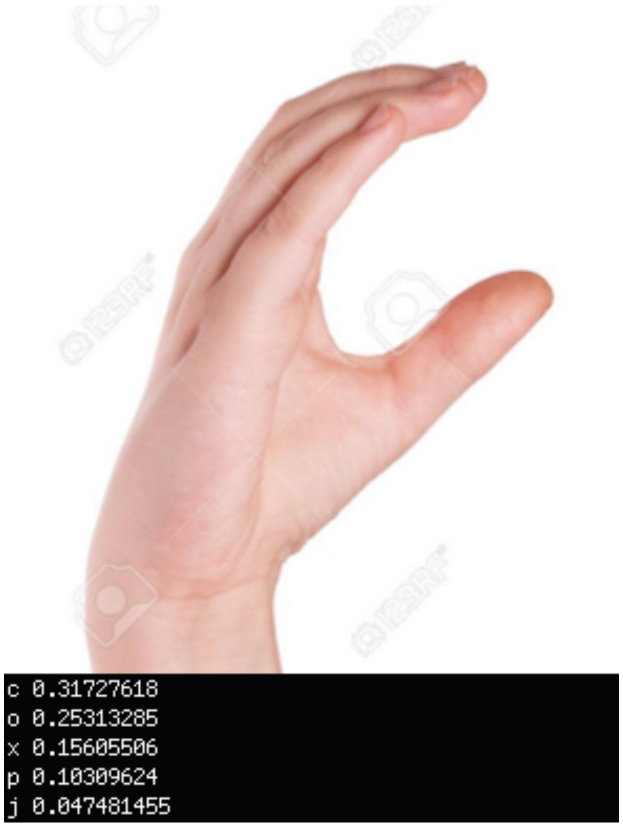
3. Getting Input

The application's sign-to text feature primarily relies on the device having a camera. Once the user chooses to translate sign language to text, the camera will open and with a press of the translate button, the device will translate each frame through the model.

4. Showing Results

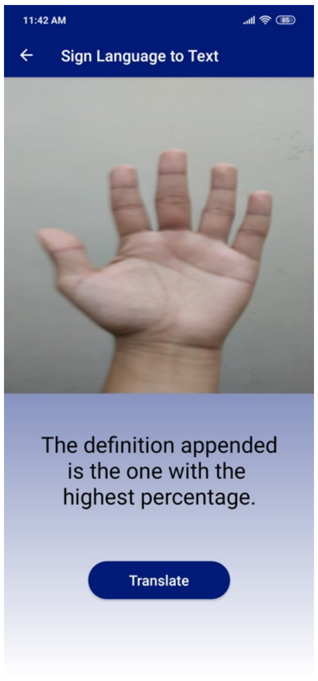
When the model is done processing a frame, the definition with the highest confidence level will be appended. The confidence level must be above the required threshold for it to be considered as an acceptable prediction.

After testing for the appropriate threshold, a minimum confidence level of 20 percent is required for a result to be returned to give way for similar gestures such as the letter c and o (see Figure 7). Anything less than that will not return anything as the model will most likely predict it as an invalid gesture. The initial look of the screen can be seen in Figure 8.



```
c 0.31727618
o 0.25313285
x 0.15605506
p 0.10309624
j 0.047481455
```

Figure 7. 31.72% letter c (CORRECT)



The definition appended is the one with the highest percentage.

Figure 8. Initial Screen

G. Evaluation

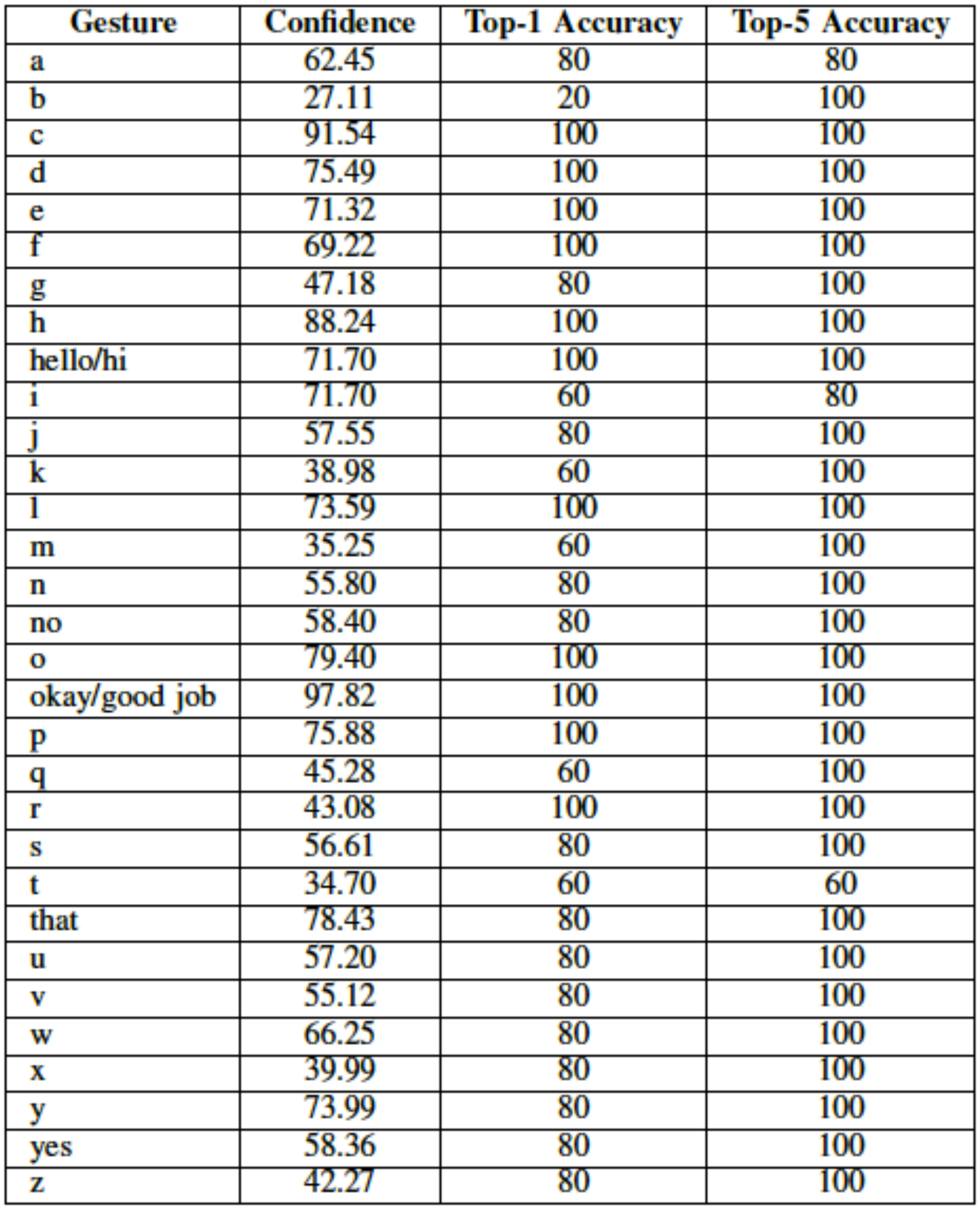
The results from the tests were measured and evaluated with the following metrics:

1. Confidence level - refers to the certainty of the model's prediction.
2. Top-1 accuracy rate - determines the rate at which the gesture was translated correctly.
3. Top-5 accuracy rate - notes how often the target label was predicted within the five most probable guesses.

IV. RESULTS AND DISCUSSION

Out of the 2,480 data, five images per gesture were taken to test the accuracy of the model. All images had a controlled background. Table II shows the mean confidence level during the testing phase, along with the top-1 and top-5 accuracy rates (in percentages).

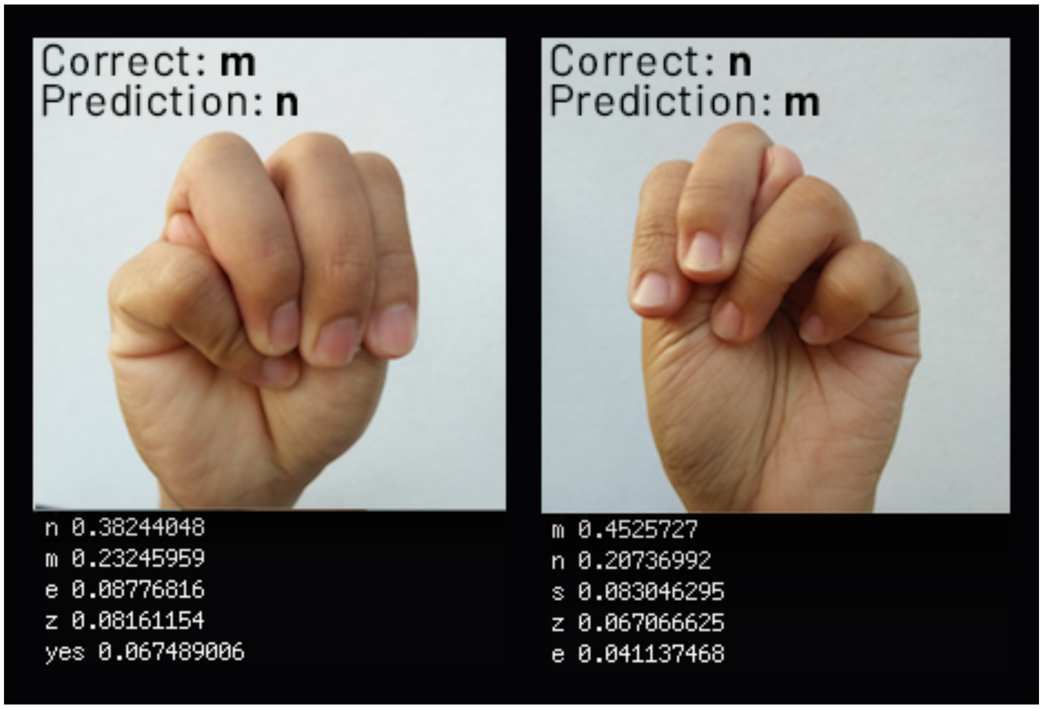
TABLE II: Results from testing five images per gesture



Gesture	Confidence	Top-1 Accuracy	Top-5 Accuracy
a	62.45	80	80
b	27.11	20	100
c	91.54	100	100
d	75.49	100	100
e	71.32	100	100
f	69.22	100	100
g	47.18	80	100
h	88.24	100	100
hello/hi	71.70	100	100
i	71.70	60	80
j	57.55	80	100
k	38.98	60	100
l	73.59	100	100
m	35.25	60	100
n	55.80	80	100
no	58.40	80	100
o	79.40	100	100
okay/good job	97.82	100	100
p	75.88	100	100
q	45.28	60	100
r	43.08	100	100
s	56.61	80	100
t	34.70	60	60
that	78.43	80	100
u	57.20	80	100
v	55.12	80	100
w	66.25	80	100
x	39.99	80	100
y	73.99	80	100
yes	58.36	80	100
z	42.27	80	100

An average of 60.03% confidence level was obtained for all gestures while averages of 81.94% and 97.42% for the top-1 and top-5 accuracies were obtained.

Gestures with low confidence level do not necessarily entail that the model fails to correctly classify them. Since most of the features of the hands are similar, gestures that look like one another can affect the confidence levels. However, the training data used became highly influential in the accuracy of the model. The test cases are from gestures with controlled backgrounds only because the model finds it difficult to translate gestures when the background is noisy. An example can be seen in Figure 9 where the predictions for m and n were switched.



Correct: **m**
Prediction: **n**

```
n 0.38244048
m 0.23245959
e 0.08776816
z 0.08161154
yes 0.067489006
```

Correct: **n**
Prediction: **m**

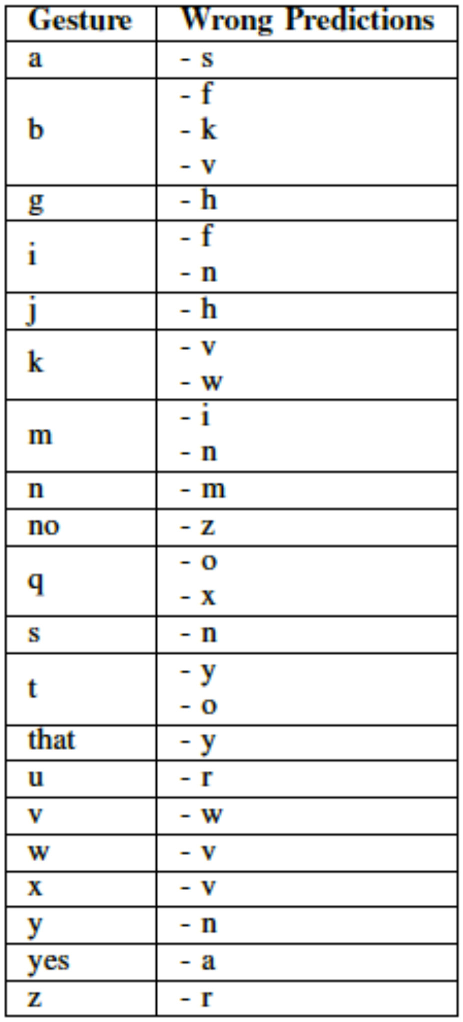
```
m 0.4525727
n 0.20736992
s 0.083046295
z 0.067066625
e 0.041137468
```

Figure 9. Left to Right: m classified as n and n classified as m

Okay/good job is the gesture with the highest confidence level (97.82%) since it is the most unique gesture out of the 31 available. Meanwhile, the gesture with the lowest confidence level is b with 27.11%. The more the gestures have similarities, the lower the produced confidence level will be since the probability will be distributed. Although with similar gestures, they can still be classified correctly according to the top-1 accuracy rates. If not, then the target label will most likely be part of the top-5 predictions.

There were gestures with perfect predictions, namely: c, d, e, f, h, hello/hi, l, o, okay/good job, p, and r. However, this does not guarantee that the model will be able to perfectly predict these gestures all the time. The gestures which produced inaccurate predictions can be found in Table III along with the gestures they were confused with.

TABLE III: Wrong predictions - these predictions happened at most one time except for b = f which occurred two times out of five test cases



Gesture	Wrong Predictions
a	- s
b	- f
	- k
	- v
g	- h
i	- f
	- n
j	- h
k	- v
	- w
m	- i
	- n
n	- m
no	- z
q	- o
	- x
s	- n
t	- y
	- o
that	- y
u	- r
v	- w
w	- v
x	- v
y	- n
yes	- a
z	- r

V. CONCLUSION

Translating sign language to text using convolutional neural networks is a feasible approach. It is powerful and compared to other neural networks, it can work with high resolution images. However, a significant amount of training data is needed to make the model more accurate.

Since the training data had a controlled background, when the background becomes too noisy and the edges of the hand are not properly defined, the model faces difficulty in translating. With these constraints, SIGN ME UP is not ready for public use.

Further optimization is required to speed up the sign language to text feature. Detecting moving gestures was not achieved aside from looking at the most significant frame due to the delays of retrieving the predictions.

In terms of enhancing the previous sign language translator from Villamor's study, the application can not only translate sign language to text but also the other way around. When it comes to sign language to text, the model can still translate gestures whenever the hand's color is similar to the color of the background.

VI. RECOMMENDATIONS

One of the biggest challenges in implementing sign language translators is the translation of moving gestures. If the application could be optimized more so that the capturing of the images and returning of the results are faster, then the frames can be chained to check if the gesture being done is correct. Otherwise, another approach or model may be more appropriate in solving this issue. Other rooms for improvement are as follows:

A. Data Gathering

It is highly recommended to have a larger data set when retraining the model. As much as possible, try to consider different skin tones and hand sizes. To add, be more strict with the lighting conditions and minimize shadows. Make sure that the signer knows the correct form and make the gestures uniform with little to no differences. Lastly, having training data with both controlled and uncontrolled backgrounds will be beneficial.

B. Model Optimization

Even with the optimization methods done in this study, the model can be made lighter if emerging technologies such as TensorFlow Lite, are used. Models built for mobile applications such as MobileNet can also be used for faster results.

C. Interface

Redesign the other screens to make it more appealing. Allowing the user to turn on the phone's torch without going through the phone's settings could be useful while translating sign language to text. In text to sign language, having short clips for moving gestures will be better than the single reference photo with hand silhouettes.

BIBLIOGRAPHY

A. Krizhevsky, I. Sutskever, and G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Advances In Neural Information Processing Systems.

C. Szegedy, V. Vanhoucke, J. Shlens, and Z. Wojna (2014). Rethinking the Inception Architecture for Computer Vision.

G. S. Villamor and J. M. Samaniego (2018). A Gesture-to-Speech Assistive Application.

J. Schmidhuber (2015). Deep Learning in neural networks: An overview. Neural Networks.

J. Wu (2016). Introduction to Convolutional Neural Networks.

K. O'Shea and R. Nash (2015). An Introduction to Convolutional Neural Networks.

P. Sharma and S.Sharma (2015). Evolution of Hand Gesture Recognition: A Review. *International Journal of Engineering and Computer Science.*

R. Akmeliawati, M. P.-L. Ooi, and Y.C. Kuang (2007). Real-time Malaysian Sign Language Translation using Color Segmentation and Neural Network. *2007 IEEE Instrumentation & Measurement Technology.*

S. Wilcox and P. Wilcox (1991). Learning to see: Teaching American Sign Language as a Second Language.

World Health Organization (2018). Retrieved from: http://www.who.int/news-room/fact-sheets/detail/deafnessand-hearing-loss

X. Chai, G. Li, Y. Lin, Y. Tang, and X. Chen (2015). Sign Language Recognition and Translation with Kinect.