

36. Országos Tudományos Diákköri Konferencia

**Real-time hand gesture classification using deep learning for
Hungarian sign language**

Sakir Kanmis

2023

Table of Contents

Table of Contents.....	1
Introduction	2
Related Work	3
Methods	5
LSTM Model.....	5
Training.....	6
K-nearest neighbors Model.....	7
Algorithm.....	7
Training.....	7
Hidden Markov Model.....	8
Approach.....	8
Image Acquisition.....	8
Pre-processing.....	9
Feature Extraction.....	10
Dataset and Data Augmentation	10
Additional Supported Features	10
Results.....	11
Conclusions.....	13
Summary	13
Acknowledgements	14
Funding.....	14
References.....	14
Appendices	20
Appendix A.....	20
Appendix B	21
Appendix C	22
Appendix D	23
Appendix E	24

1. Introduction

The sign language is a crucial method of communication for deaf and speech impaired people. As it was stated by the World Federation of the Deaf, there are approximately 300 sign languages in the world which have been used among 70 million people [1]. Sign language includes the movement of different organs, such as finger, hands, arms, head gestures, and facial expressions. However, in this work recognition is based solely on hand gestures. There are several parameters while modelling sign language, which are handshape, orientation in coordinate system, motion, position etc. In order to get an accurate result, all these parameters must be taken into account. A successful implementation of sign language recognition could remove the communication barrier between sign language users and others. Researchers in the field of sign language recognition have shown an increased interest in developing methods for human-to-human and human-to-computer interaction in recent years. Also, leading tech companies started to pay attention to the field and included sign language solutions into their new projects. Deaf-dumb people rely on sign language interpreters to communicate with others. However, providing experienced and qualified interpreters all the time is quite difficult and unaffordable. So, the implementation of automated sign language translation systems is required to fulfil the equal opportunities in terms of communication. Therefore, human – computer interaction systems should function as sign language interpreter. However, various challenges exist to overcome in vision-based sign language recognition such as dependency on lighting conditions, background complexity, variety of hardware used since different hardware can result in different outcome. Due to these challenges, it is not an easy task to develop a standardized model. [2] This paper proposes a system which recognizes the Hungarian Sign Language for human – computer interaction using deep learning based on hand gestures. To help people with such disabilities, the proposed approach can recognize a set of 12 classes of Hungarian letter, which includes both static and dynamic gestures.

2. Related Work

Previous research work has used various approaches for the recognition of sign language, for example, motion gloves, motion controllers, cameras. Motion gloves for this purpose are costly, have limitations while adapting for hand structure and are not enough comfortable compared to vision-based approach. Also, it takes significant time

and can result in inaccurate outcome due to the wear and tear caused by long term use of the gloves.[3] Due to complex sign gestures and similarities between gestures, sign recognition still has been a challenge for Kinect sensor. [4-5] As a one of the vision-based sign recognition approaches, image processing is cost-friendly and efficient solution. Though, it needs series of processing to recognize hand and fingers, that leads to longer time interval to obtain outcome. Also, skin color and lighting applied are parameters that affect data accuracy and precision [6]. On the other hand, the leap motion controller is compact in size and more affordable than solutions that have been discussed previously. It process the data fast and requires less memory to operate as well. [7]. However, its sampling frequency is inconsistent and in order to reduce its possible effect, post-processing is needed. In [8], authors used Hidden Markov Models(HMM) to detect face expressions out of a sequence of videos with a combination of Gaussian Tree Augmented Naive Bayes Classifier and Bayesian Network Classifiers . In [9], authors also used HMM to classify gestures based on 30 words. The dataset authors used was RWTH-BOSTON-50. Work in [10] proposes a Indian sign language recognition system. Framework of the system starts with pre-processing followed by feature extraction and recognition. In terms of framework, it is similar to work proposed in this paper. In [11], authors used K-nearest neighbors for classification. Feature extraction is done by Histogram of Oriented Gradient (HOG). Authors was able to get an accuracy 94.23% for 26 letters from American Sign Language. The work done by Hasan et al. [12] focuses on sign language recognition for Bangla alphabet. Authors used Atrificial Neural Network (ANN) and Freeman Chain Code (FCC) for feature extraction. Proposed approach has an accuracy of 96.5%. Last but not least, in [13] authors used two pre-trained convolution neural network namely, VGG16 and AlexNet for feature extraction and support vector machines for classification. Proposed work achieved 99.82% accuracy.

In Table 1, different approaches of sign language recognition systems are reviewed and compared in terms of accuracy.

Method	Sample Size	Accuracy
Kinect [14]	40 gestures	82.1%
Leap Motion Controller [15]	24 static samples	82.71
Data Glove [16]	50 static 30 dynamic gestures	80.4
k-NN classification [17]	6x480 samples	83.78
HMM classification [18]	20 gestures	90%

Table 1. Comparison of different approaches in sign language recognition

3. Methods

In our approach, there are three types of model used, respectively, long short-term memory (LSTM) model as pre-classifier, K-nearest neighbors (KNN) model as static gesture classifier, Hidden Markov model (HMM) as dynamic gesture classifier. The pre-classifier is used to distinguish dynamic and static gestures at the beginning. The outcome of the pre-classifier states the classifier that will be used later. Obviously, the main difference between dynamic and static gestures is time-varying changes of gesture. Therefore, time series analysis would be an efficient technique. The interface we used, Keras[19], is a high level API operating on TensorFlow[20]. TensorFlow is a machine learning framework developed by Google. It operates at large scale and in heterogeneous systems. It uses dataflow graphs to illustrate how data moves. The nodes of a dataflow graph are mapped to different machines in a cluster, and within a machine across CPUs, GPUs and Tensor Processing Units (TPUs).

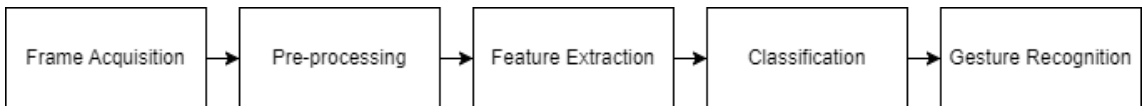


Figure 1. Outline of System

3.1 LSTM Model

The structure of an LSTM[21] unit has input,output,forget gates and cell which regulate the information and states as shown in Fig. 2. Sigmoid functions are associated with the gates which adjust the opening/closing throughout the learning. The long-term

memory in LSTM is called as the cell state. It handles the data to be stored from the previous time step. Forget gate determines if the data will be kept or forgotten. If the forget gate is equal to 0, data is forgotten, if forget gate is equal to 1, data is kept.

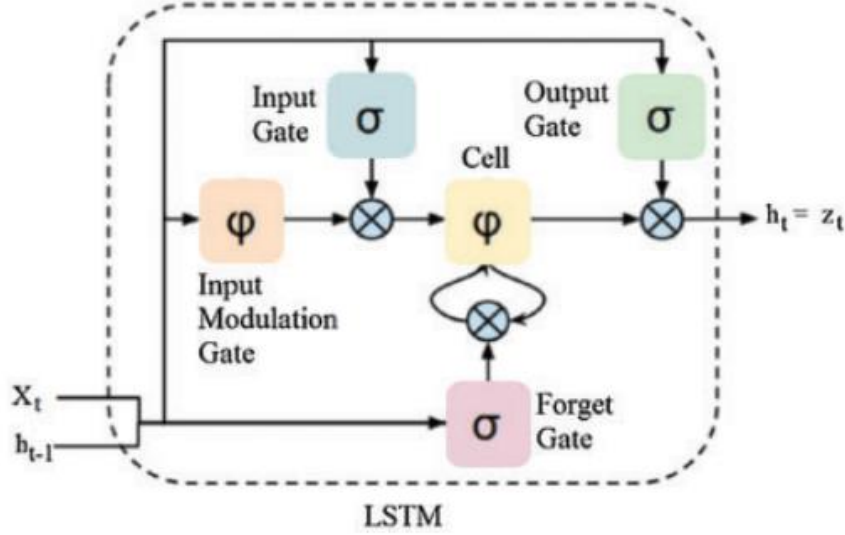


Figure 2. Structure of the LSTM unit.

Eqs. (1) to (6) explain the working of the gates of LSTM.

$$i_t = \sigma(x_t w_{xi} + h_{t-1} w_{hi} + c_{t-1} w_{ci} + \omega_{ibais}) \quad (1)$$

$$f_t = \sigma(x_t w_{xf} + h_{t-1} w_{hf} + c_{t-1} w_{cf} + \omega_{fbais}) \quad (2)$$

$$z_t = \tanh(x_t w_{xz} + h_{t-1} w_{hz} + \omega_{zbais}) \quad (3)$$

$$c_t = z_t \otimes i_t + c_{t-1} \otimes f_t \quad (4)$$

$$o_t = \sigma(x_t w_{xo} + h_{t-1} w_{ho} + c_{t-1} w_{co} + \omega_{obais}) \quad (5)$$

$$h_t = o_t + \tanh(c_t) \quad (6)$$

where, “ i_t ” is the input gate and “ f_t ”, “ o_t ”, “ z_t ” are the forget, output and cell gates. “ c_t ” and “ h_t ” represent output memory activation as a function of time ‘ t ’. Eqs. (2), (3), (5) and (6) are the equations for forget gate, output gates and hidden state.[22]

3.1.1 Training

Training is done with the following parameters, 10 epochs, batch size : 8, 32 neurons, the length of time series : 10, vector dimension of each input : 1024. Adam algorithm is used as optimizer.

Adam optimizer is an algorithm for first-order gradient-based optimization with low memory requirements. It computes adaptive estimation for first and second moments of the gradients; the name Adam originates from adaptive moment estimation. [23]

3.2 K-nearest neighbours Model

The main point of the k Nearest Neighbours (kNN) algorithm is to use a dataset that data points are separated into different classes to predict the classification of a new unclassified sample. Each characteristic of training dataset is considered as a different dimension in the space, and each characteristic is given a coordinate in that dimension, we end up having a set of points in space. Based on distance between two points, we can analyze the similarity between them. We pick k number of data out of dataset to predict class of new sample, prediction is based on election process. [24]

3.2.1 Algorithm

The working of algorithm is listed as following:

1. A positive integer k and a new sample point are specified.
2. k number of entries which are closest to new sample in the dataset are picked.
3. Most common classification of these entries are found.
4. Finally, new sample is classified according to common classification.

3.2.2 Training

During the training phase, dimensionality reduction is done by Principal Component Analysis (PCA). It is a statistical method for reducing dimensionality of dataset. It enhances interpretability but with minimum level of information loss. It is done by creating unrelated additional variables. Finding these new variables will reduce the eigenvalue/eigenvectors problem. PCA is an adaptive data analysis method since variables are created to adapt to various type of data and structure. [25]

3.3 Hidden Markov Model

A hidden Markov model (HMM) is a statistical framework that the system getting modeled is considered as a Markov chain having hidden states. In an HMM, the state is not directly observable, however output, which depends on the state, is observable. A HMM can be defined as following formulas [26]:

- 1) Set of states can be represented as $\{s_1, s_2, \dots, s_N\}$ where N is number of states.
- 2) Transition probability from state i to j is represented as a_{ij}
- 3) A sequence of k observation is represented as $O = O_1, O_2, \dots, O_k$
- 4) Initial probability of N states is represented as $\pi = \pi_1, \pi_2, \dots, \pi_N$
- 5) Observation probability of O_k at state N is represented as $B = b_N(O_k)$

As it is stated with formulas above, a set of parameters of a HMM can be denoted as $\lambda = (A, B, \pi)$. In order to describe a complex process such as hand gesture recognition using HMM, parameters must be selected properly.[27]

3.4 Approach

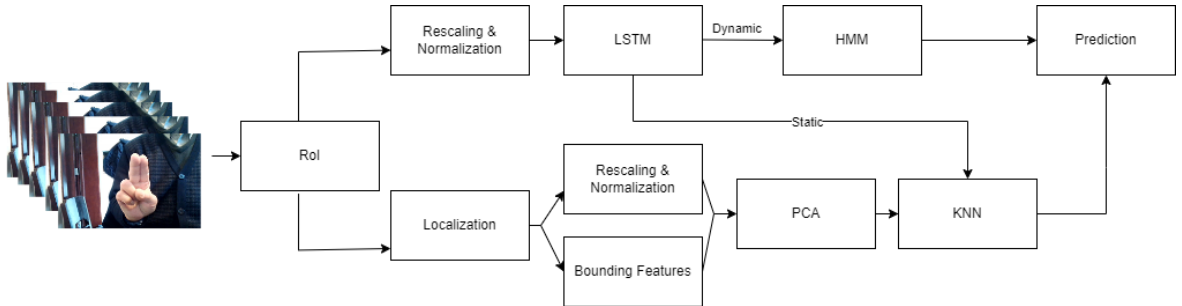


Figure 3. Architecture of the proposed model

3.4.1 Image Acquisition

In our work, user must be in front of the camera to capture hand gesture. Image acquisition is started manually by pressing an assigned keyboard button. Camera is displayed through OpenCV library. OpenCV is an open source library focusing on computer vision applications. It contains tools to be used for image-processing functions and algorithms for applications of object detection, face recognition [28]

3.4.2 Pre-processing

Distortion and unwanted noise can always be present at frames captured through camera. Furthermore, size of the sequence of frames may be quite large to train our model.

In order to use the memory efficiently, eliminate environmental dependencies and enhance extraction of features, preprocessing is crucial. [29] Some parameters such as lighting, background, camera parameters and location make the scene complex to process. Therefore, some series of preprocessing must be applied. First, in order to apply skin color segmentation, color space is changed to HSV. The color space can be transformed by using Eqs. (7)-(9). [30]

$$H = \cos^{-1} \frac{1/2(2R-G-B)}{\sqrt{(R-G)^2 - (R-B)(G-B)}} \quad (7)$$

$$S = \frac{\max(R,G,B) - \min(R,G,B)}{\max(R,G,B)} \quad (8)$$

$$V = \max(R, G, B) \quad (9)$$

After color space transformation is applied, Region of Interest (RoI) cropping is applied to detect hand and remove the noise present. NumPy[31] library is often used during pre-processing stage for image processing.

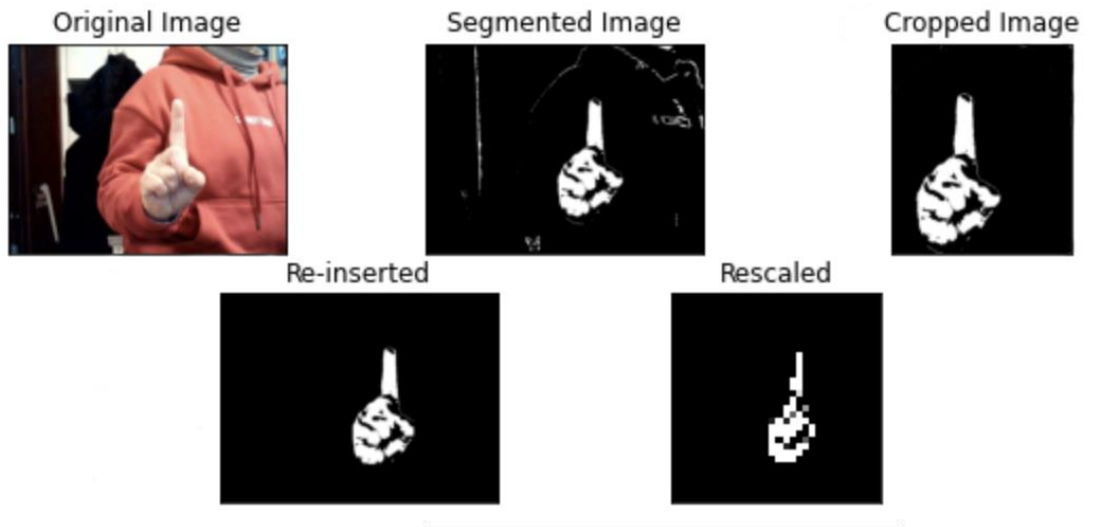


Figure 4. Pre-processing steps

3.4.3 Feature Extraction

Feature extraction is another important aspect to get accurate predictions. By extraction features, we consider the frames as input. Purpose of feature extraction to determine the distinguishing characteristics of classes in the dataset. To achieve good performance, selecting appropriate features is important as well. For example, in case of bad lighting conditions, color feature is not helpful for the system. Instead, we can benefit other features such as hand shape, hand motion etc. In case of dynamic gesture

recognition both temporal and spatial features must be considered. A feature vector is a vector which has n dimension. Computation of feature vector requires memory and time. Generally, as number of prediction class is increased, number of required feature vectors is also increased.[32]

3.4.4 Dataset and Data Augmentation

In order to focus more on architecture of model, instead of creating dataset from scratch, we used a publicly available dataset which was originally developed for American Sign Language by David Lee [33]. We adopted the dataset to use 12 classes, namely, “Á,B,C,D,É,F,G,H,J,L,Ú,Z”. Feeding deep learning models with more data results in better training and accurate performance. For this purpose, we use data augmentation techniques to modify the dataset and create variations of the images which will support the training. In this work, we used random cropping to augment the data. The data is splitted in the purpose of training and test in a ratio of 80:20.

3.4.5 Additional Supported Features

In order to increase accessibility of system, we have implemented Text-to-Speech feature using Google Text-to-Speech (gTTS) library [34]. gTTS provides easier integration of Google text recognition technologies into applications. It is able to send text and receive synthesized audio output from the Cloud Text-to-Speech API. In this way, we aim to help visually impaired peoples in society.

Also, we added bounding box feature for display output to visualize hand detection more clearly. We have used “MediaPipe Hands” [35] pipeline to extract hand landmarks, then drawn a rectangle box to cover all the landmarks. In this way, no matter which hand gesture is performed, hand is always located inside the bounding box.

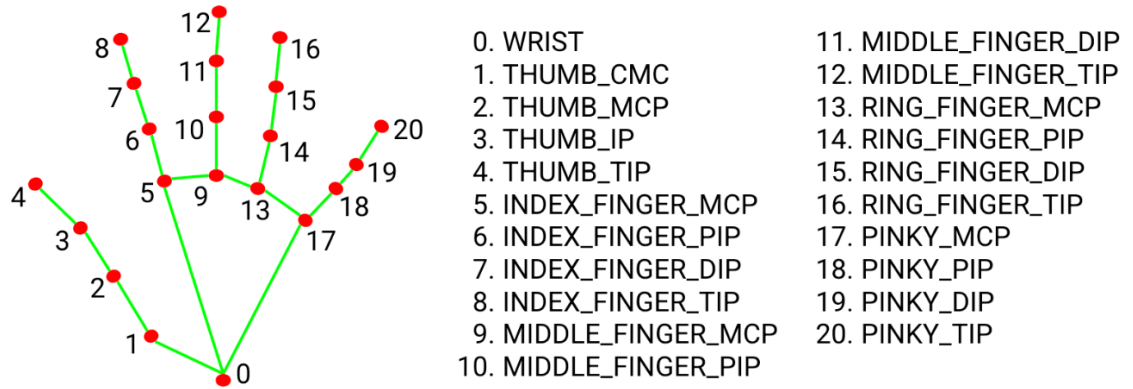


Figure 5. Hand landmarks extracted by MediaPipe Hands

4. Results

This section presents the experimental results of the proposed model. Training and validation are carried out using M1 Macbook Air with 8 GB RAM.

Figure 6 below shows illustrates the confusion matrix of the system. Overall, performance is satisfying. There are some bias in class of “J” caused by dynamic gesture to perform it.

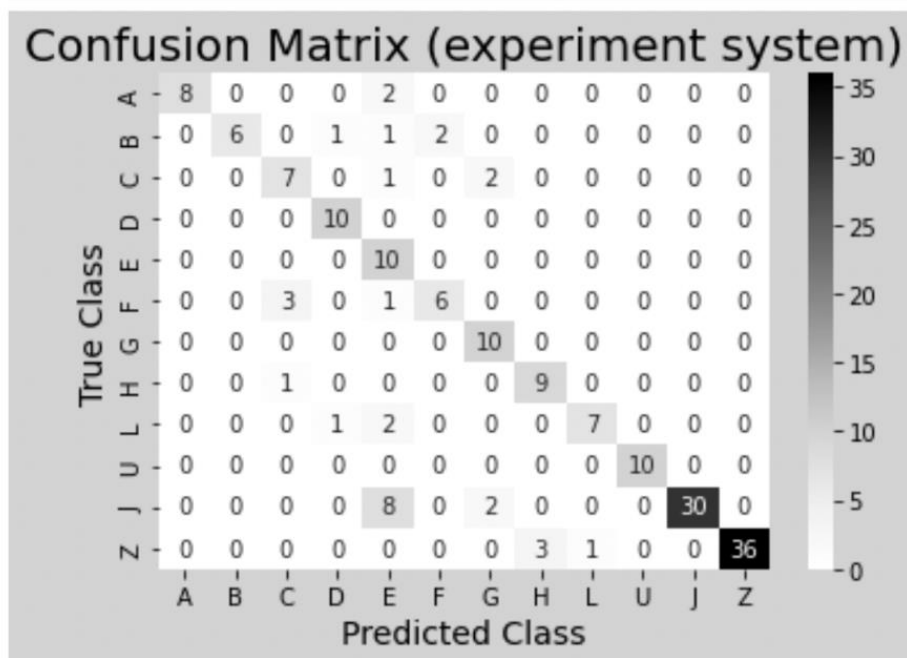


Figure 6. Confusion Matrix

Figure 7 and 8 illustrates the model accuracy and model loss change over each epoch. In total, model is trained for 10 epochs to get the best fit.

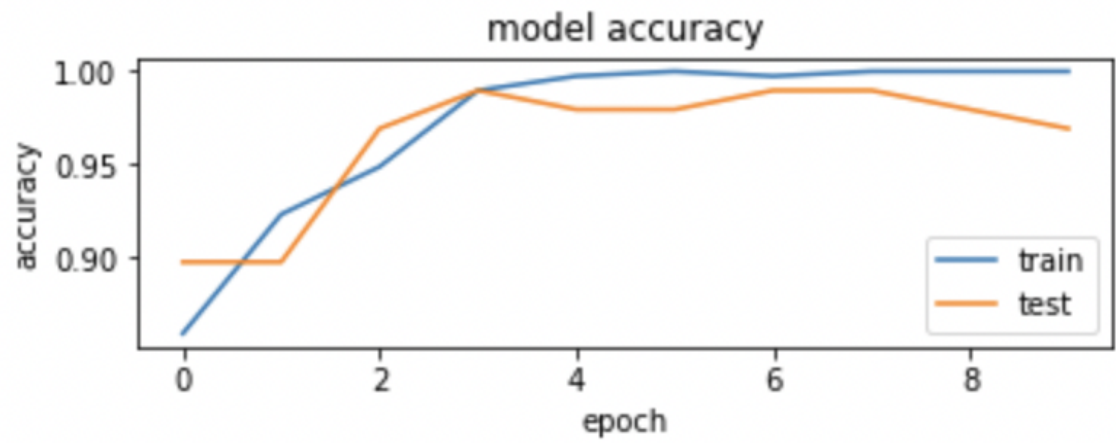


Figure 7. Accuracy Plot of Proposed Model

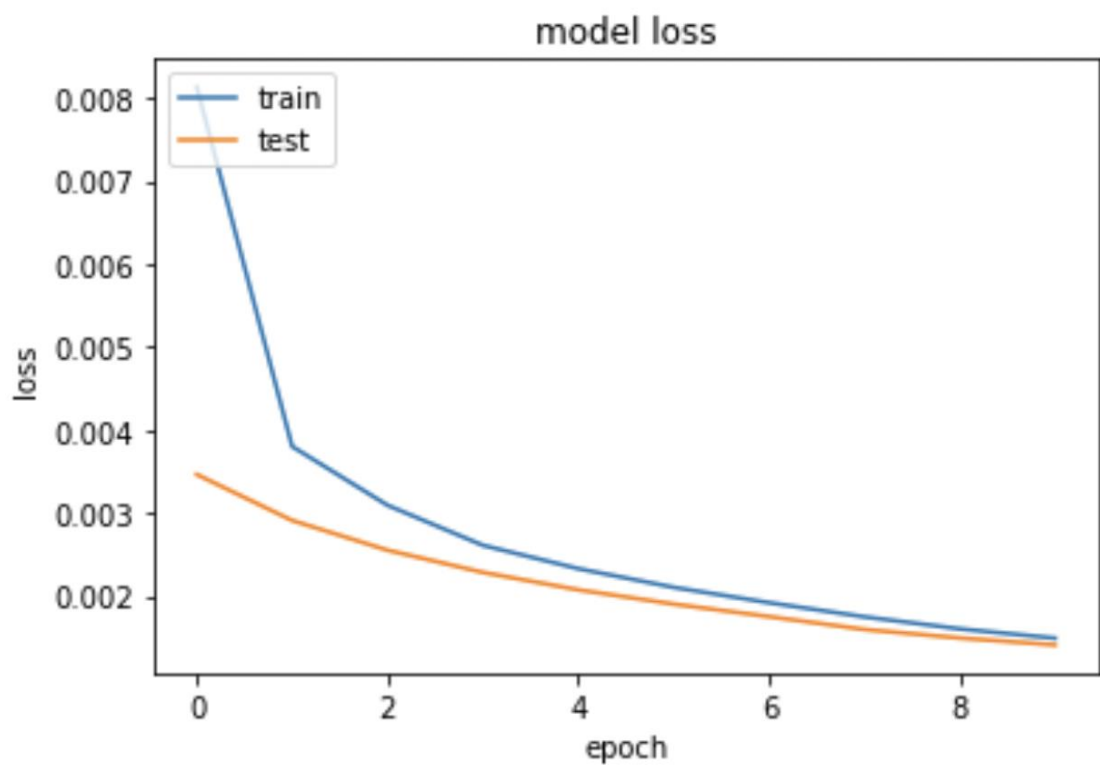


Figure 8. Model loss versus epoch

5. Conclusions

A deep learning model is proposed that can predict static and dynamic hand gestures based on frame series acquired by camera. The architecture consists of a LSTM classifier to distinguish static and dynamic gestures, a KNN classifier to predict static gestures, a HMM classifier to predict dynamic gestures. Pre-processing techniques are applied to frames to detect hand more efficiently and to remove noise that is present. Random cropping is applied to augment dataset. The proposed model achieved 83.9% test accuracy.

6. Summary

Main purpose of the work is to help the speech-impaired people by creating a computer vision-based sign language translator. We have reviewed the previous work done in sign language recognition field and compared their cons and pros such as accuracy, cost, feasibility etc. We have used a public dataset to train our models. Proposed model consists of three different types of classifier, namely LSTM, KNN and HMM. LSTM model decides whether performed gesture is static or dynamic gesture. Depending on the output of LSTM, static gestures are classified once again by KNN model. In a similar manner, dynamic gestures are classified by HMM model. During the classification, extraction of features such as position, shape and movement of hands are directly affect the prediction. Also, pre-processing of images are another important aspect in terms of accuracy and speed of classification. Noise removing and color segmentation leads to higher accuracy and since pre-processed images will take smaller size in memory, it also results in faster classification. Overall performance of the model was satisfying and promising.

Regarding the future improvement on work, a custom dataset can be created specific for Hungarian language since it has special letters. Further experimental tests can be carried out with different type of classifiers and feature extraction techniques to find out the best accuracy. The proposed model can be optimized for mobile devices to publish it as a mobile phone application. In that way, the mobility of the model would increase.

7. Acknowledgements

I would like to thank the University of Pécs, Faculty of Engineering and Information Technology and Huawei Technologies Hungary Kft. for the cooperation in the frame of “UP FEIT Huawei SEEDs Excellence Scholarship” and providing such opportunity for students. I would also like to thank Zsolt Kisander for his support and supervision during the program. Last but not least, I would like to thank Wu Wenyu from University of Pécs, Faculty of Sciences for her suggestions and comments about the work.

8. Funding

The work was supported by the UP FEIT Huawei SEEDs Excellence Scholarship, the Application Work was created with the support of Huawei Technologies Hungary Kft.

9. References

- [1] Murray, J. (2020). World Federation of the deaf.
- [2] Rastgoo, R., Kiani, K., & Escalera, S. (2021). Sign Language Recognition: A Deep Survey. In *Expert Systems with Applications* (Vol. 164, p. 113794). Elsevier BV.
- [3] Tubaiz, N., Shanableh, T., & Assaleh, K. (2015). Glove-Based Continuous Arabic Sign Language Recognition in User-Dependent Mode. In *IEEE Transactions on Human-Machine Systems* (Vol. 45, Issue 4, pp. 526–533). Institute of Electrical and Electronics Engineers (IEEE).
- [4] Sun, C., Zhang, T., Bao, B.-K., Xu, C., & Mei, T. (2013). Discriminative Exemplar Coding for Sign Language Recognition With Kinect. In *IEEE Transactions on Cybernetics* (Vol. 43, Issue 5, pp. 1418–1428). Institute of Electrical and Electronics Engineers (IEEE).
- [5] Wu, J., & Jafari, R. (2017). Wearable Computers for Sign Language Recognition. In *Handbook of Large-Scale Distributed Computing in Smart Healthcare* (pp. 379–401). Springer International Publishing.
- [6] Bheda, V., & Radpour, D. (2017). Using Deep Convolutional Networks for Gesture Recognition in American Sign Language (Version 3). arXiv.

- [7] Naglot, D., & Kulkarni, M. (2016). Real time sign language recognition using the leap motion controller. In 2016 International Conference on Inventive Computation Technologies (ICICT). 2016 International Conference on Inventive Computation Technologies (ICICT). IEEE.
- [8] Ira Cohen, Nicu Sebe, Ashutosh Garg, Lawrence S, Chen and Thomas S. Huang (2003, February). Facial expression recognition from video sequences: temporal and static modeling. *Computer Vision and Image Undertaking* 91
- [9] Zaki, M.M., Shaheen, S.I.: Sign language recognition using a combination of new vision based features. *Pattern Recognition Letters* 32(4), 572–577 (2011)
- [10] Continuous dynamic Indian Sign Language gesture recognition with invariant backgrounds by Kumud Tripathi, Neha Baranwal, G. C. Nandi at 2015 Conference on Advances in Computing, Communications and Informatics (ICACCI)
- [11] Mahmud, I., Tabassum, T., Uddin, M. P., Ali, E., Nitu, A. M., & Afjal, M. I. (2019). Efficient noise reduction and HOG feature extraction for sign language recognition. In *Proceedings of the international conference on advancement in electrical and electronic engineering, ICAEEE 2018* (pp. 1–4).
- [12] M. M. Hasan, M. Khaliluzzaman, S. A. Himel and R. T. Chowdhury, "Hand sign language recognition for Bangla alphabet based on Freeman Chain Code and ANN," *2017 4th International Conference on Advances in Electrical Engineering (ICAEE)*, 2017, pp. 749-753
- [13] Barbhuiya, A.A., Karsh, R.K. & Jain, R. CNN based feature extraction and classification for sign language. *Multimed Tools Appl* **80**, 3051–3069 (2021).
- [14] Keskin C, Kırac F, Kara YE, Akarun L (2013) Real time hand pose estimation using depth sensors. In: *Consumer depth cameras for computer vision 2013*, Springer, London, pp 119–137
- [15] Funasaka M, Ishikawa Y, Takata M, Joe K (2015) Sign language recognition using leap motion controller. In: *Proceedings of the international conference on parallel and distributed processing techniques and applications (PDPTA)*, The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), p 263

- [16] Liang R-H, Ouhyoung M (1998) A real-time continuous gesture recognition system for sign language. In: Automatic face and gesture recognition, 1998. Proceedings. Third IEEE international conference, IEEE, pp 558–567
- [17] Lionnie R, Timotius IK, Setyawan I (2012) Performance comparison of several pre-processing methods in a hand gesture recognition system based on nearest neighbor for different background conditions. *J ICT Res Appl* 6:183–194
- [18] Chen F-S, Fu C-M, Huang C-L (2003) Hand gesture recognition using a real-time tracking method and hidden Markov models. *Image Vis Comput* 21:745–758
- [19] Chollet, F. & others, 2015. Keras. Available at: <https://github.com/fchollet/keras>.
- [20] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems (Version 2). arXiv.
- [21] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. In *Neural Computation* (Vol. 9, Issue 8, pp. 1735–1780). MIT Press - Journals.
- [22] Zhang, R., Liu, Y., & Sun, H. (2020). Physics-informed multi-LSTM networks for metamodeling of nonlinear structures. In *Computer Methods in Applied Mechanics and Engineering* (Vol. 369, p. 113226). Elsevier BV.
- [23] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization (Version 9). arXiv.
- [24] O. Sutton, “Introduction to k-Nearest Neighbor Classification and Condensed Nearest Neighbor Data Reduction” Feb. 2012.
- [25] Salih Hasan, B. M., & Abdulazeez, A. M. (2021). A Review of Principal Component Analysis Algorithm for Dimensionality Reduction. In *Journal of Soft Computing and Data Mining* (Vol. 2, Issue 1). Penerbit UTHM.
- [26] L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, *Proceedings of The IEEE* 77 (2), 1989, pp.257–285.

- [27] Yang, Z., Li, Y., Chen, W., & Zheng, Y. (2012). Dynamic hand gesture recognition using hidden Markov models. In 2012 7th International Conference on Computer Science & Education (ICCSE). 2012 7th International Conference on Computer Science & Education (ICCSE 2012). IEEE.
- [28] Pulli, K., Baksheev, A., Korniyakov, K., & Eruhimov, V. (2012). Real-time computer vision with OpenCV. *Communications of the ACM*, 55(6), 61-69.
- [29] Brian L. Pulito, Raju Damarla, Sunil Nariani, " 2-D Shift Invariant image Classification Neural Network, which overcomes Stability, Plasticity Dilemma", Vol 2, International Joint Conference on Neural Network, San Deigo, June 17-21,1990.
- [30] Shaik, K. B., Ganesan, P., Kalist, V., Sathish, B. S., & Jenitha, J. M. M. (2015). Comparative Study of Skin Color Detection and Segmentation in HSV and YCbCr Color Space. In *Procedia Computer Science* (Vol. 57, pp. 41–48). Elsevier BV.
- [31] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. In *Nature* (Vol. 585, Issue 7825, pp. 357–362). Springer Science and Business Media LLC.
- [32] Mekala, P., Gao, Y., Fan, J., & Davari, A. (2011). Real-time sign language recognition based on neural network architecture. In 2011 IEEE 43rd Southeastern Symposium on System Theory. 2011 IEEE 43rd Southeastern Symposium on System Theory (SSST 2011). IEEE.
- [33] Lee, D. (2022, July 27). American Sign Language Letters Object Detection Dataset. Roboflow. <https://public.roboflow.com/object-detection/american-sign-language-letters>
- [34] Pndurette. (n.d.). Pndurette/gtts: Python library and CLI Tool to interface with Google Translate's text-to-speech API. GitHub. Retrieved November 23, 2022, from <https://github.com/pndurette/gTTS>
- [35] Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M. G., Lee, J., Chang, W.-T., Hua, W., Georg, M., & Grundmann, M. (2019). MediaPipe: A Framework for Building Perception Pipelines (Version 1). arXiv.

Appendices

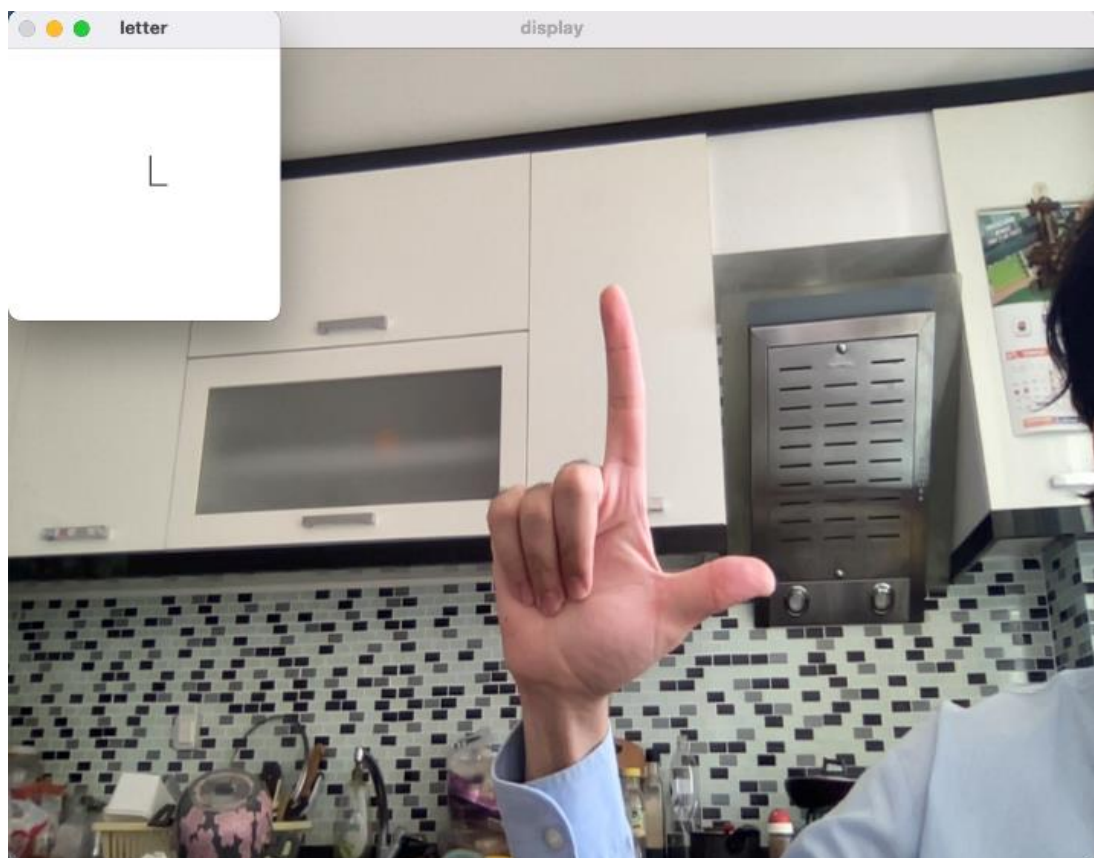
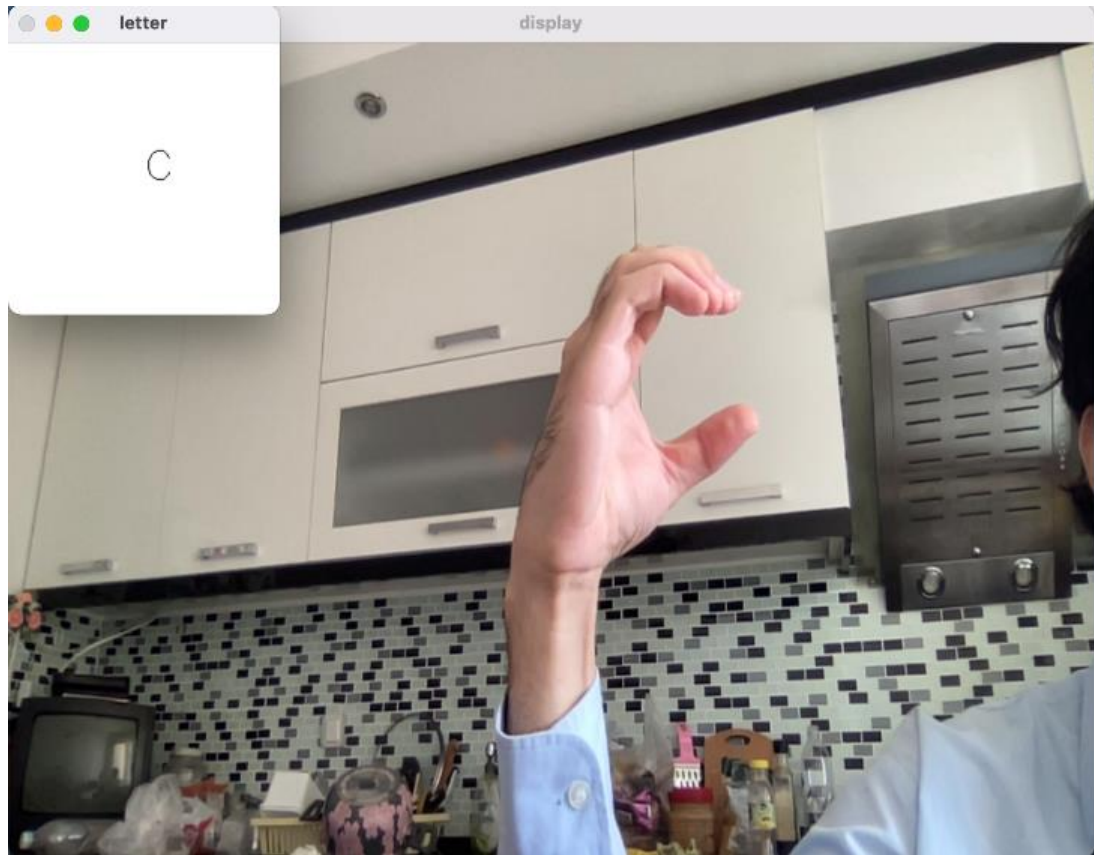
Appendix A

The figure below shows the training stage of LSTM for 10 epochs with a 80:20 training and validation ratio.

```
Train on 392 samples, validate on 98 samples
Epoch 1/10
392/392 [=====] - 2s 6ms/step - loss: 0.3696 - a
ccuracy: 0.8597 - val_loss: 0.3038 - val_accuracy: 0.8980
Epoch 2/10
392/392 [=====] - 1s 2ms/step - loss: 0.2109 - a
ccuracy: 0.9235 - val_loss: 0.2081 - val_accuracy: 0.8980
Epoch 3/10
392/392 [=====] - 1s 2ms/step - loss: 0.1138 - a
ccuracy: 0.9490 - val_loss: 0.1000 - val_accuracy: 0.9694
Epoch 4/10
392/392 [=====] - 1s 2ms/step - loss: 0.0573 - a
ccuracy: 0.9898 - val_loss: 0.0637 - val_accuracy: 0.9898
Epoch 5/10
392/392 [=====] - 1s 2ms/step - loss: 0.0201 - a
ccuracy: 0.9974 - val_loss: 0.0555 - val_accuracy: 0.9796
Epoch 6/10
392/392 [=====] - 1s 2ms/step - loss: 0.0102 - a
ccuracy: 1.0000 - val_loss: 0.0335 - val_accuracy: 0.9796
Epoch 7/10
392/392 [=====] - 1s 2ms/step - loss: 0.0155 - a
ccuracy: 0.9974 - val_loss: 0.0414 - val_accuracy: 0.9898
Epoch 8/10
392/392 [=====] - 1s 2ms/step - loss: 0.0070 - a
ccuracy: 1.0000 - val_loss: 0.0389 - val_accuracy: 0.9898
Epoch 9/10
392/392 [=====] - 1s 2ms/step - loss: 0.0041 - a
ccuracy: 1.0000 - val_loss: 0.0451 - val_accuracy: 0.9796
Epoch 10/10
392/392 [=====] - 1s 2ms/step - loss: 0.0032 - a
ccuracy: 1.0000 - val_loss: 0.0549 - val_accuracy: 0.9694
```

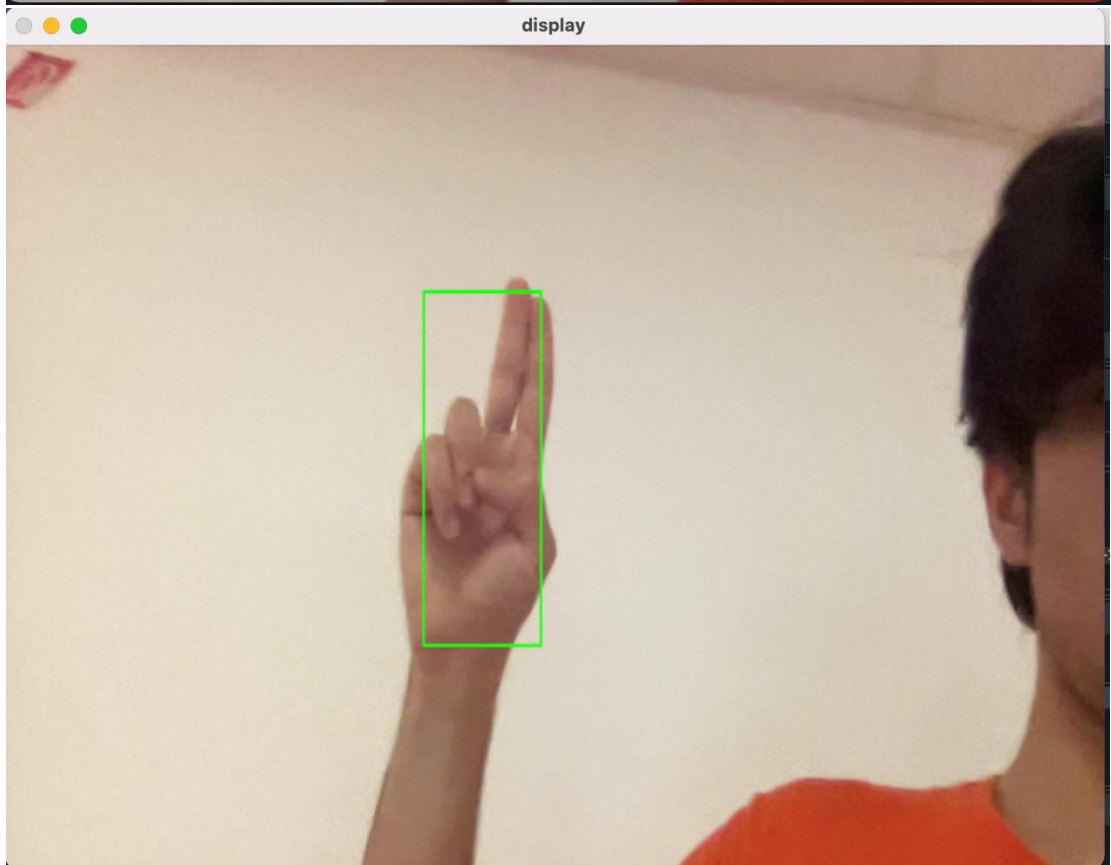
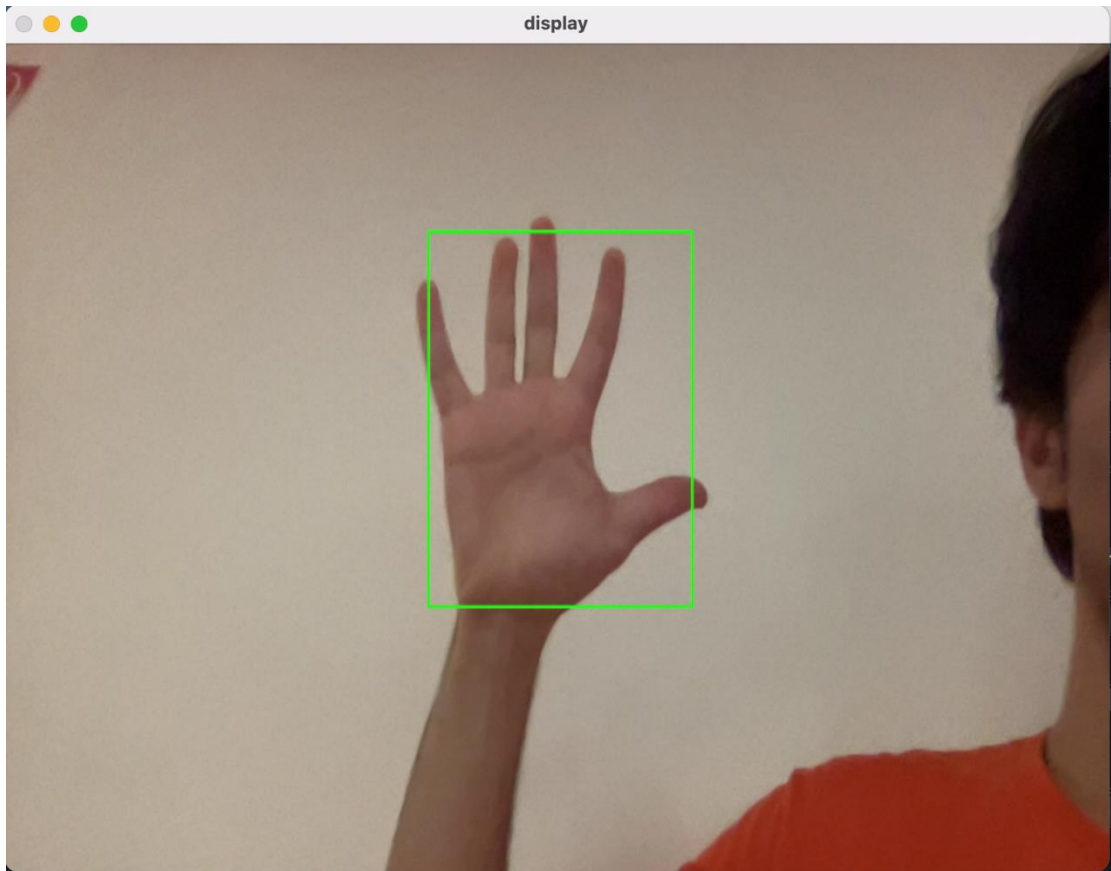
Appendix B

Screenshots below shows the output predictions for static gestures.



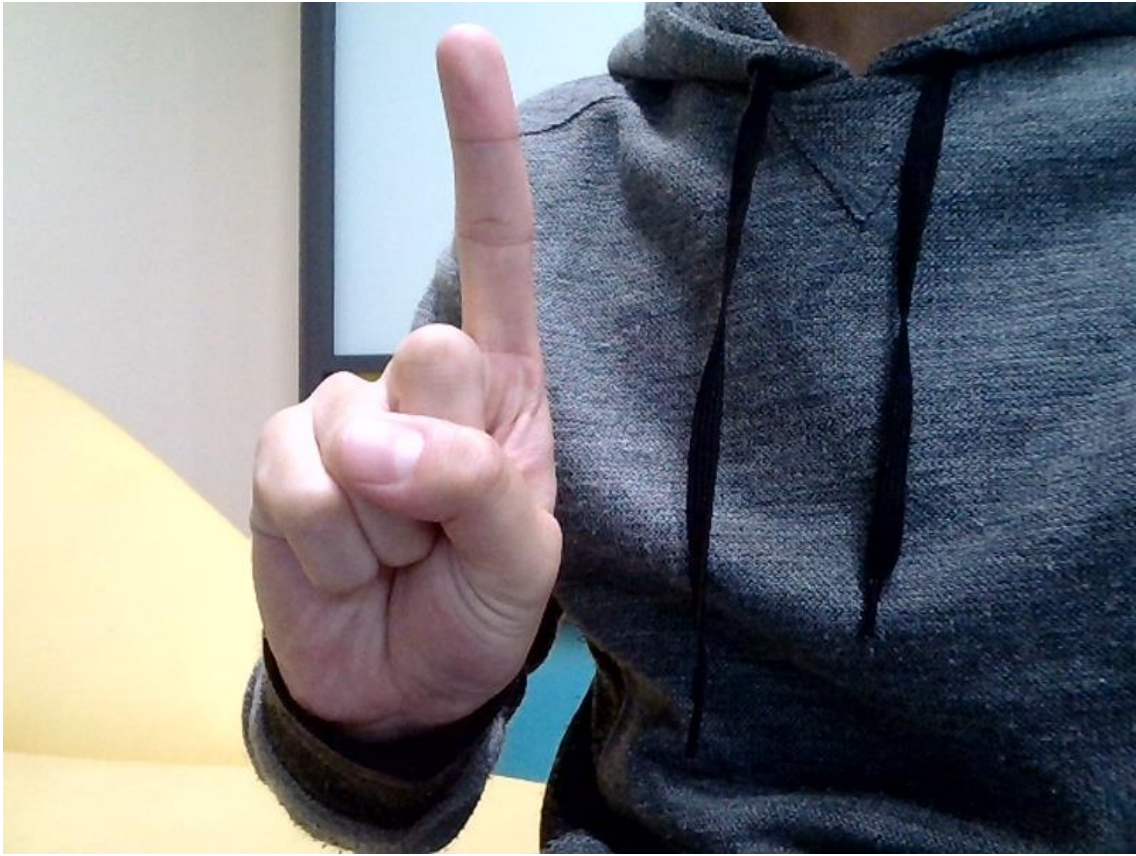
Appendix C

The figures below show the implemented bounding box with different hand gestures.



Appendix D

Figures below illustrate sample images of dataset. (D above, G below)



Appendix E

Figures below illustrate the working of random cropping process. (above original image, below random cropping applied image)

