International Journal of Computer Applications Technology and Research
Volume 9–Issue 07, 217-224, 2020, ISSN:-2319–8656

A Multi-Modular Approach for Gesture Recognition and Text Formulation in American Sign Language

Sarvesh Joglekar
Student
Dwarkadas J. Sanghvi College of Engineering
Mumbai 400056, India

Hrishikesh Sawant
Student
Dwarkadas J. Sanghvi College of Engineering
Mumbai 400056, India

Aayush Jain
Student
Dwarkadas J. Sanghvi College of Engineering
Mumbai 400056, India

Priya Dhadda
Student
Dwarkadas J. Sanghvi College of Engineering
Mumbai 400056, India

Pankaj Sonawane
Assistant Professor
Dwarkadas J. Sanghvi College of Engineering
Mumbai 400056, India

Abstract: The most important thing in communication is hearing what isn't being said, this goes for the people who are deaf and mute. Sign language is the most natural means of exchanging information amongst these people. The deaf and mute people have to rely on an interpreter or some sort of visual communication while communicating with the rest of society. So, to bridge the gap between the deaf and dumb community and the rest of the world, there is the need for the translation system that will make the entire process of communication fluent. American Sign Language (ASL) is a visual and gestural language i.e. the brain processes linguistic information through eyes. This paper proposes a system that interprets the gestures in sign language enacted by the person to plain text. It is a robust as well as real-time Intelligent Translation System ensuring that none of these two parameters are compromised. This application would capture a live video stream of the user performing ASL from the webcam which then translates the gestures to corresponding text using Image Processing techniques and Deep Learning Models. It is then combined to form words that are passed to the spell checker and then combined to form sentences. The proposed solutions would thus enable quick and effective communication between the deaf and mute community and the rest of the society.

Keywords: Morphological Transforms, Single Shot Detector, Real-time application, Convolutional Neural Network, Deep Learning, Edit Distance.

1. INTRODUCTION

Sign language is a communication medium for the hearing impaired population. It uses a combination of hand-shapes, movement, and orientation of hands or arms and facial expressions to convey meaning. There is no universal sign language and a sign language in a region can also have multiple dialects, just like any other full-fledged language. The unfamiliarity of sign language makes it difficult for these people to converse with the normal population. Hence, there is a need for Sign Language interpreters for translating sign language to language which a layman can understand but, such interpreters are limited in numbers and this option may prove to be expensive. This influenced us to develop a sign language translation system that could automatically convert American Sign Gestures into the corresponding plain contextual text by detecting hand gestures in real-time. The purpose of going ahead with American Sign Language as compared to other Sign languages is that both ASL and English activate the same area of the brain i.e. the left brain. Also, both require building words to form sentences.

A pipeline structure of modules is developed where the output of one module is passed onto the next which performs a certain set of operations on it. In our approach, the Hand Detection and Gesture Recognition components are separated. Hand Detector performs detection and finds out the positions of hand i.e. localization of hand and draws the bounding box around it. This intermediate result is passed to the next module. There have been many instances where this RGB/Grayscale frame is passed on for recognition to neural networks. However, we intend to further perform background elimination and feature extraction before passing further for gesture recognition. This has allowed us to have lightweight neural network architecture for gesture recognition as only the most relevant features are fed to it. This is the key factor that has allowed us to build a real-time application as all lightweight modules performing certain functionality are lined one after another instead of a single heavyweight module.

Also, there would be cases where the lighting conditions would be poor hence this might lead to the incorrect recognition of some of the letters. Hence this paper also implements Spell Corrector which is fed words as input and it checks them for any errors and returns the closest word to the input word. Systems like these can help develop the deaf and mute community. The proposed solution would not only simplify the process of communication at the deaf and mute people's side but also makes it easy for the rest of the society to understand them and thus lead to successful communication.

2. LITERATURE REVIEW

Vedak et al. (2019) in their paper Sign Language Interpreter using Image Processing and machine learning have developed a system for translating Indian Sign Language to speech. It uses two stages 1) Processing and feature extraction 2) Recognition of hand sign.

a) Dataset Preparation:

The video was broken down such that 100 frames would get collected from the video and later augmentation of the data was performed such as to create 250 images. Hence in total, 6000 images were generated, out of which 4800 were used for training while 1200 were also used for testing.

b) Processing of Images for Edge Detection:

To see that the essential features are getting selected, image processing was done in such a way that the useful data was captured whereas the redundant as well as distracting data or noise were neglected or rejected. The images generated for processing were converted to grayscale, processed upon, and then were passed as a binary-image.

c) Feature Extraction and Recognition:

This paper made use of a Histogram of Gradients method for extracting features from the dataset i.e. Histogram gradient was calculated for each cell and a vector was generated.

Vivek Bheda and N. Dianna Radpour (2017) in their paper ―Using Deep Convolutional Networks for Gesture Recognition in American Sign Language have presented a method for using deep convolutional networks to classify images of both the letters and digits in American Sign Language. While the translation process between signs and a spoken or written language is formally called 'interpretation,' the function that interpreting plays is the same as that of translation for a spoken language. In their research, they looked at American Sign Language, which is used in the USA and English-speaking Canada and has many different-dialects. The paper described a deep learning approach for a classification algorithm of American Sign Language. In particular, there has been work done in the realm of sign language recognition using deep CNNs, with input-recognition that is sensitive to more than just pixels of the images. With the use of cameras that sense depth and contour, the process is made much easier via developing characteristic depth and motion profiles for each sign language gestures. The results obtained had 82.5% accuracy on the alphabet gestures and 97% validation set accuracy on digits when using the NZ ASL dataset.

Brandon Garcia and Sigberto Alarcon Viesc (2016) in their paper Real-time American Sign Language Recognition with Convolutional Neural Networks In real-time to translate a video of a user's ASL into text present the development and implementation of an American Sign Language (ASL) fingerspelling translator based on a convolutional neural network. Reconstructing and displaying the most likely word from classification scores (output). To obtain images of the user signing in real-time, a web application that is enabled access to a native camera on a laptop through the browser. Once the user has indicated that they're finished signing, the top-5 letters for each position in the spelled word are passed to a custom unigram language model based on the Brown Corpus. Using these, single word unigram probabilities and a handful of other heuristics, it re-turns the optimal word to the user.

Masood et al. (2018) in their paper American Sign Language Character Recognition using Convolution Neural Network proposes a vision-based system to identify symbols of ASL which includes Alphabets from A to Z and numbers ranging from 0 to 9. The first step of this approach is Image Augmentation and Resizing. Dataset consisting of 2524 images is augmented by performing shear, zoom, horizontal and vertical shifting operations to create a dataset of 14781 images out of which 11085 images are used for training and 3696 images are used for validation. In the resizing process, the image is resized to the size of 224x224 pixels. The second step of this methodology is Image Pre-processing which involves centering the data at the pixel level. For this purpose mean value of RGB over all pixels is subtracted from each pixel value. For image classification, authors make use of a convolutional neural network model that focuses on extracting sparse image features in low dimensional space. The input to the convolutional neural network is a 224x224 RGB image. Each image is passed through a stack of convolutional layers having filters of size 3x3. Pooling layers of this CNN model employ the concept of max pooling. The spatial padding of a convolutional layer is selected such that the resolution is preserved after convolution and for convolution operation stride of 1 pixel is used. Model is equipped with 3 fully connected layers out of which the first 2 layers have 4096 nodes each and make use of the ReLU activation function while the last layer is a Softmax layer which predicts 36 symbol classes. Stochastic Gradient Descent is used to train the model for 4 epochs with a batch size of 128, where learning rate, decay rate, and momentum are set to 0.001, 10^{-6}, 0.9 respectively. This approach yields an average accuracy of 95.54%.

3. METHODOLOGY

Our comprehensive methodology aims at making this product into a real-time usable product. Hence instead of creating a single but larger deep neural network to perform a single task, the architecture is broken down into subparts resulting in smaller modules catering to each of the subparts which are explained below. In this paper, a longer but lighter pipeline of components is designed instead of a heavier one.

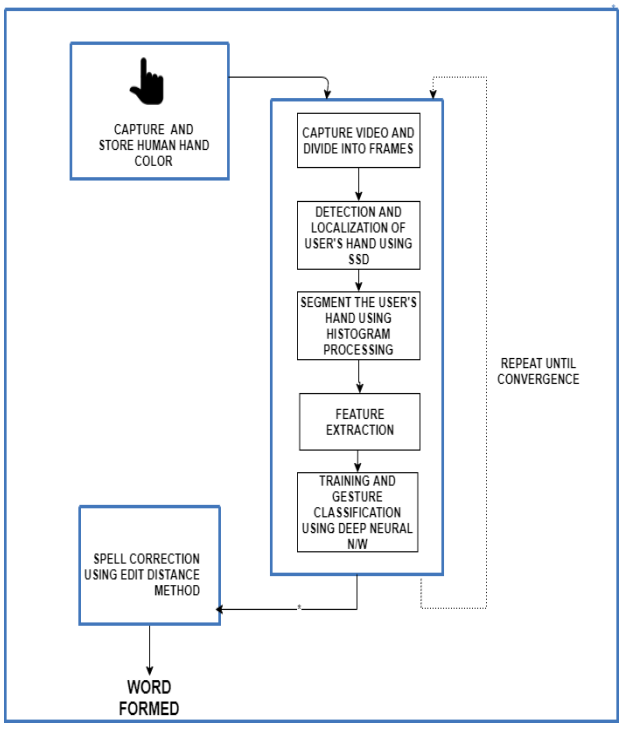

Figure 1. System Architecture

3.1 Hand Detection and Localization

3.1.1 Single Shot Detector

In this paper, we have performed localization of the user's hand using Multibox Single Shot Detectors as demonstrated by (Wei Liu et.al., 2016). It runs CNN only once in a single forward pass and hence it is designed for real-time applications because of its speed. It differs from other detectors because of the usage of multiple layers that provide a finer accuracy on objects with different scales. SSD makes use of anchor boxes to factor the variations in aspect ratio and scale of objects. SSD is used with the base network MobileNet. The activations of this base network are captured which is then passed on to a subnetwork that works as a classifier and a localizer. It applies 3 x 3 Conv. filters for each cell to make predictions and then make use of a Non-Maxima Suppression (NMS) to filter out multiple boxes for a single object to the one closest to the ground truth box during prediction.

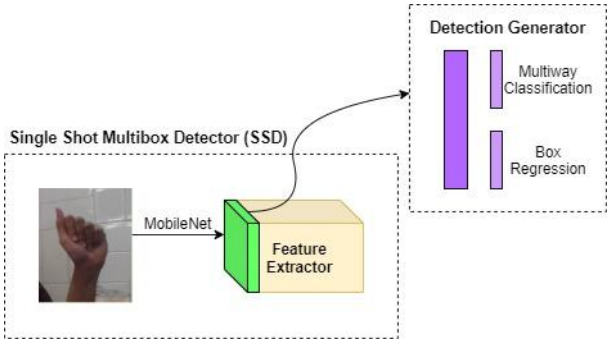

Figure 2. Single Shot Multibox Detector

3.1.2 MobileNet

MobileNet, which is developed by Google is used as our base network for SSD. It is a lightweight model used for mobile applications. MobileNetV1 is the preferred model for our application. It is preferred over MobileNetV2 because it requires lesser memory access as it can also be used on mobile devices as they have slower memory access. Generally, SSD is used with VGG, Resnet but these models have large network architecture and are huge in size around 300 MB. Hence to detect in real-time MobileNet (Andrew Howard et.al., 2017) is used whose architecture consists of 3×3 depthwise convolution followed by a 1×1 pointwise convolution. This reduces the number of parameters remarkably and hence this reduces the amount of floating-point multiplication operations. Since the user's hand would generally be placed near the screen, hence the sacrifice of lesser accuracy obtained by depthwise separable convolutions would not affect the object detection.

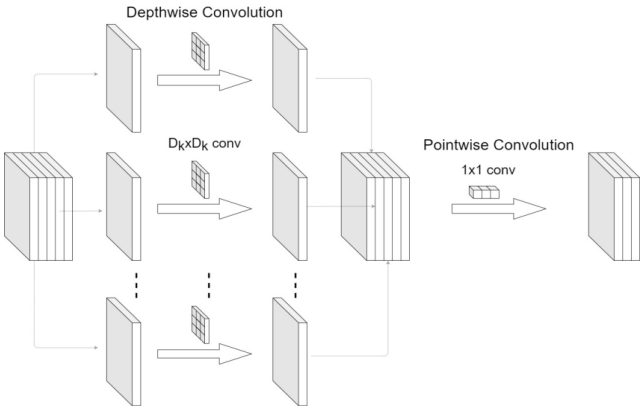

Figure 3. MobileNet Convolutions

3.1.3 Dataset

We have created our dataset by capturing the images of the hand in different postures, backgrounds, and positions. This dataset contains about 800 images. The location i.e. the coordinates of only the palm of the hand is stored which is fed to the network along with the images. These images are augmented by resizing, flipping, and rotating these images. This helps in detecting both the hands irrespective of their orientation. This helps the model to learn well and avoids over-fitting to a certain posture of the hand. 1000 images are been selected randomly for training.

3.1.4 Training and Transfer Learning

A pre-trained model of SSD with MobilenetV1 is used which is trained on the COCO (Common Objects in Context) dataset. Since the COCO dataset contains similar objects and commonly found objects, it is used to further train on the dataset of the hand. This method is known as Transfer Learning. It makes use of knowledge gained while solving one problem applies it to another related problem. Hence the weights present in the above model are further fine-tuned by re-training the model by our dataset, where the use of initial weights helped us to learn quickly on the dataset. This is possible because the weights here are not randomly initialized which is the case when we completely train the model from scratch. The benefits of this kind of training are that it demands less amount of time and computation power. Also, we were able to train the model successfully with less data as mentioned in 3.1.3.

3.2 Segmentation

The previous module results in a bounding box around a user's hand. The image inside the box is captured and passed to this module. Here we aim to grab only the user's skin, thereby eliminating the rest of the background. This helps significantly to reduce noise thereby focusing on the required object i.e. hand. In this paper, the method of histogram storage and back projection is followed. This paper makes use of spatial histogram features in an HSV format (Hue, Saturation, and Value) to accurately capture the data of the human hand. This data format is used over others as it helps to separate the color component and the intensity component which is essential for segmentation (Mavani et.al., 2017). This data is stored in a file and is made use when gesture recognition is to be performed. The back-projection of the stored hand histogram in the file is computed with every frame. Median blur removes the noise that may be present. Gaussian blur is used to smoothen out the wavy structure. This frame is used as a mask that is applied to the image from the previous module to remove the background from the original frame.

Figure 4. Segmentation of hand and background removal

3.3 Image Processing and Feature Extraction

After the removal of background, this stage looks to enhance and extract the features of the hand so that gesture recognition becomes seamless. The predominantly used method is binary thresholding, but it only retains the outer shape thus losing out on other hand features. Hence, a certain morphological transformation known as morphological gradient demonstrated by (Anh et.al., 2012) is performed with the help of a kernel of size 3x3. It is the difference between the dilation and erosion of the image. It mostly affects the pixels near the boundary of the foreground and background and hence proves to be a very good feature extractor in our case. This highlighted the way the fingers are placed in a certain gesture and also the shape of the boundary of the hand better than a plain grayscale image. Few ASL alphabets like M, N, S, and T differ in the varied position of the thumb. Our feature extractor shows the visible difference between these symbols with very little data as it is in grayscale format.

Figure 5. Applying Morphological Gradient Transform

3.4 Gesture Recognition

3.4.1 Dataset Description

A grayscale image with one color channel is generated by an image processing module, which showcases visible edges and dimensional orientation of the hand.

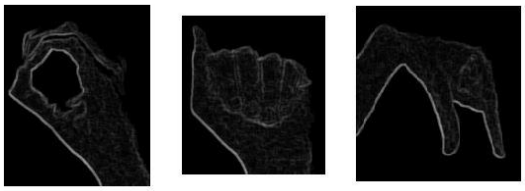

Figure 6. ASL symbols 'O', 'A' and 'Q' respectively after processing

Images of each 39 signs including alphabets and numbers along with sentence-end, switch symbol, and space symbol is generated for training and testing purposes in the same format which is outputted by the image processing module. Grayscale images are used to avoid unnecessary computations and time delay in the image classification stage. We have generated 1602 images for the training process and 855 images for the testing process.

3.4.2 Pre-Processing

For making an image eligible for image classification task it is resized to dimensions 128x128 such that it matches the input parameters of the convolutional neural network. This is the least possible size such that the features of the hand would not be missed.

3.4.3 Image Classification

The supervised learning methodology is used to predict the correct sign. A Convolutional Neural Network CNN is a Deep Learning algorithm that can take in an input image, assign importance to various aspects in the image and be able to differentiate one from the other as demonstrated by (Sultana et.al., 2018). Two CNN models are proposed CNN1 and CNN2, where CNN1 is used to predict alphabets, space, switch and end sentence symbol while CNN2 is used to predict numbers and switch symbol. CNN1 is equipped with three stages of convolution each of which is equipped with one convolutional layer and a pooling layer, convolutional layer in first and second stage uses 32 filters while in third stage it uses 64 filters, after these stages, it has a flattening layer and 3 fully connected layers, two of FC layers are having 2048 and 128 nodes respectively while the last being an output layer makes predictions for 29 symbol classes. CNN2 is equipped with two stages of convolution each of which is equipped with one convolutional layer and a pooling layer, convolutional layer in first and second stage uses 32 filters and after these stages, it has a flattening layer, a fully connected layer having 128 nodes and an output layer which makes predictions for 11 symbol classes. Each convolutional layer uses feature maps of size 3x3. For convolution operation stride of 1 pixel is used and the padding parameter is set to 0. Pooling layers have a pooling block of dimensions 2x2 and in each pooling layer, the concept of max pooling is employed as demonstrated by (Nagi et.al., 2011).

Activation function at each node modifies its input with non-linear function thereby allowing neural networks to model highly complex relationships between features. In each convolutional layer and fully connected layer excluding the output layer Rectified Linear Unit (ReLU) (Agarap, 2018) is used as an activation function while in the output layer a Softmax activation function is used.

A loss function indicates "how good" the model is at making predictions for a given set of parameters while optimization algorithms help us to minimize a loss function (Janocha and Czarnecki, 2017). Categorical cross-entropy is used as a loss function in the model and for optimization purposes, Adam optimizer is implemented (Kingma and Ba, 2014). In the

optimizer β1 and β2 values are set to 0.9 and 0.999 respectively while ε value is set to e^{-7}.

Learning Process: The 1st CNN model is trained for 100 epochs while the 2nd CNN model is trained for 75 epochs at a constant learning rate of 0.001.

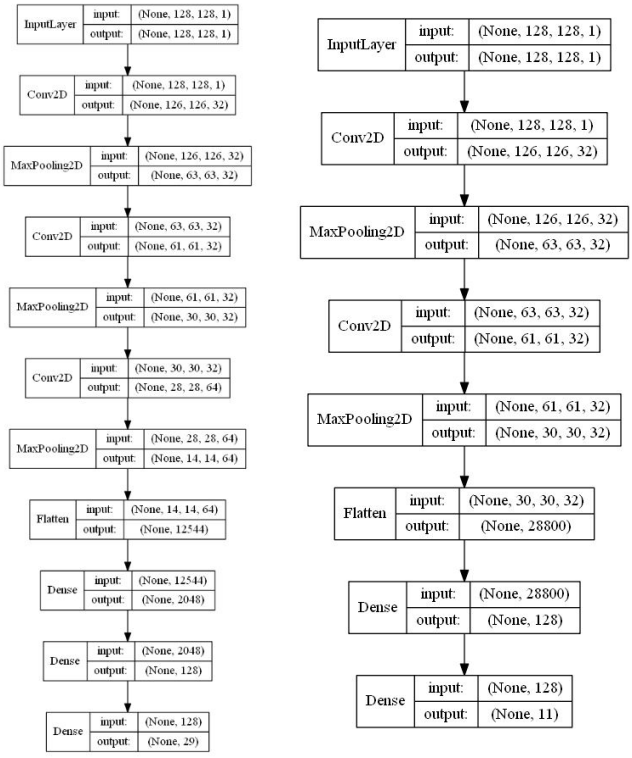

Figure 7. Network Architecture of CNN for ASL alphabets (left) and numbers (right)

3.5 Text Processing and Spell Correction

The previous module leads to a character output. These characters are appended one after another to form a word. We have introduced some special characters for our application like Space, Switch, End so that the whole process of text formation can be achieved. Switch gesture is used to toggle between recognition of ASL alphabets and ASL numbers. When word formation is complete, user gestures 'Space' symbol. The control is then passed to the Spell Corrector demonstrated by Vibhakti et.al., (2015). Such words get appended to form a sentence. When a user wishes to transmit the data, he/she can gesture the End symbol.

Figure 8. Symbol Gesture for Space, Switch and End respectively

The text processing module includes spell checking and revision usefulness to the windows based application. It causes the user to diminish the work, by distinguishing any spelling mistakes and making it more clear setting. Spell checking proposes at least one elective word as the right spelling when an incorrectly spelled word is recognized.

The letters identified by the gesture recognition module, separated by an interval, are joined together to shape a word. The word formed passes through an Edit Distance based spelling rectification module. Edit Distances is a method for placing into numbers how extraordinary two strings (e.g., words) are to each other by checking the most reduced conceivable number of activities required to transform one string into the other.

The module tries to choose the most likely spelling correction from the dictionary containing more than 10,000 words. The Edit Distance strategy distinguishes the greater part of the spelling mistakes and offers a rundown of proposals, one of which would be the right word. Thus, our framework utilizes a calculation dependent on the Edit Distances.

.

4. RESULTS AND DISCUSSIONS

4.1 Performance Metric

Accuracy is used as a metric to judge the performance of CNN models and Spell Checker.

4.2 Result Analysis

Accuracy for the first CNN model i.e. CNN1 is 92.59% while accuracy for the second CNN model i.e. CNN2 is 98.72%. A confusion matrix is presented for each model using a heatmap which depicts the prediction accuracy of each sign. Shades of the colors in the heatmap are spread across a numeric range describing the accuracy metric in terms of percentage.

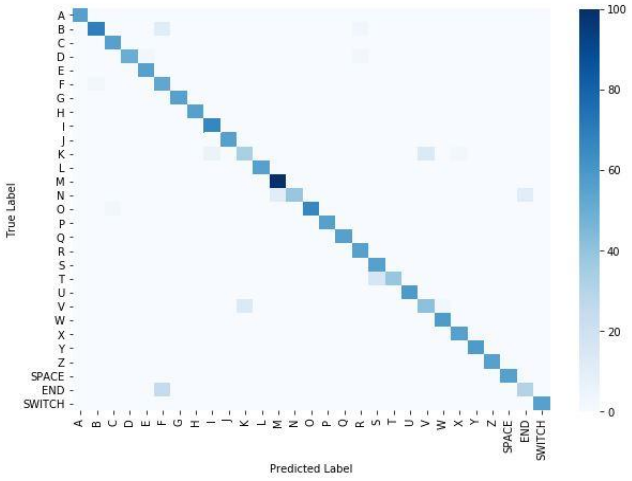

Figure 9. Confusion matrix for CNN1

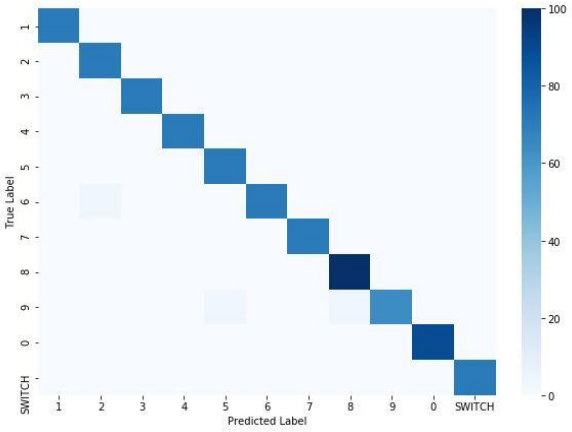

Figure 10. Confusion matrix for CNN2

It is tricky for the spell checker to know whether the incorrectly spelled words like 'san' infers 'sat' or 'say'. The model was checked for its precision on 1687 words. Out of these words, 1549 words depended on the normal results i.e., they were accurately classified as the user intended it to. However, the other 138 words, albeit right in spelling couldn't meet the normal consequences of the user. Therefore the overall accuracy achieved after spell checking is 91.82%. As this is the last step, this accuracy is considered as the accuracy of the complete system.

Figure 11. Analysis of accurately corrected words

4.3 Loss and Accuracy plots for CNN

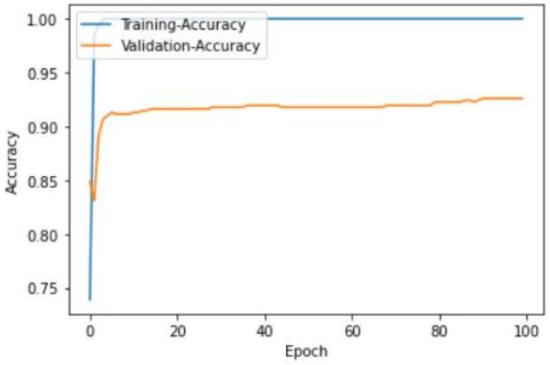

Figure 12. Variation of Accuracy with respect to epoch for CNN1

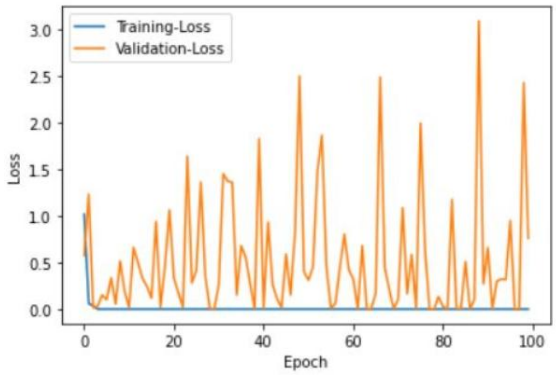

Figure 13. Variation of Loss with respect to epoch for CNN1

From the above plots, it can be seen that the validation accuracy varies up to the 90th epoch and then it becomes nearly constant while validation loss varies till the last epoch. Training Loss and Accuracy achieves constant value early in the training process.

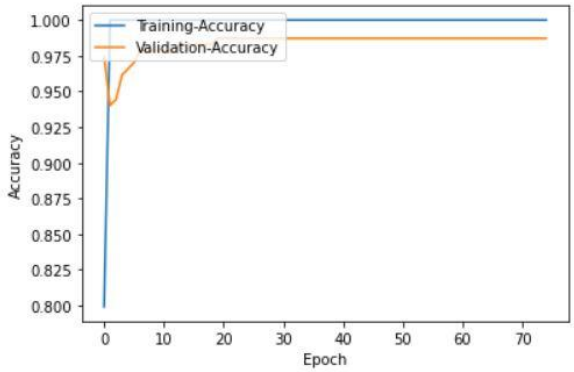

Figure 14. Variation of Accuracy with respect to epoch for CNN2

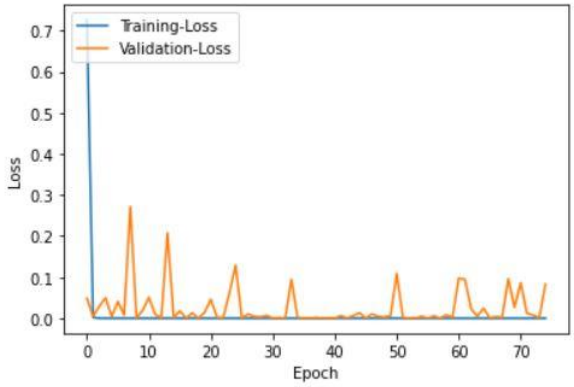

Figure 15. Variation of Loss with respect to epoch for CNN2

From the above plots, it can be inferred that initially validation accuracy varies up to 15th epoch and then it becomes constant while validation loss varies till the last epoch although being nearly constant in the middle of the training process. Training Loss and Accuracy achieve constant value early in the training process thereby showing the same behaviour as CNN1.

4.4 Experimental Observation

When gesturing the symbol Q, there are a few instances where the bounding box captures the forearm as well. When the

CNN1 model is tested in real-time most of the alphabets are classified correctly but certain cases of misclassification are encountered such as K is classified as V because of the close similarity between the signs.

The number of hands to be detected is limited to 1 in the video frame as all the gestures are performed by a single hand. Our model is very much successful in detecting the user's hand with very few false positives as the threshold score is kept at 0.5 for detecting the hand. However, there might be a few moments that the detector loses trace of the hand but detects it again by just a very small movement of the hand. Hand gestures made far away from the camera are not picked as the size of the hand appears small. However, since the user is expected to sit near the laptop or mobile while operating it, this issue is not faced. The bounding box formed around the hand is also optimal in size covering just about the perfect portion of the hand above the wrist.

4.5 Discussion

The live video feed ran at a frame speed around 6 fps on a laptop having modest hardware configurations. Thus the feed was able to capture the user's hand movement as the gestures are done at a steady pace and do not demand super-fast movements. The user had the ease to place his hand anywhere in the frame as the hand detector (explained in 3.1) tracked the user's moving hand. It cropped out the hand from the whole frame and limited the data for the gesture recognition module (explained in 3.4) to learn on. Hence we were able to later use lighter CNN for gesture recognition using a limited dataset (explained in 3.4.1). The absence of the detector would have compelled us to use a heavier network for gesture recognition along with very large training data and this would have affected the real-time ability of our system.

Despite using a lighter CNN, we were able to achieve very good accuracy for gesture recognition and adequate real-time frame speed of the application without overfitting the dataset. It was because the user's hand was segmented from the background and morphological gradient captured the most important features that would be necessary for symbol recognition. Given the accuracy CNN achieved, there could be 1 or maybe 2 alphabets misrecognized in a formed word, however, this word is corrected by the spell checker and 91.82% of the times this corrected word is what the user intended to convey.

Hence the reason for dividing the whole task of identifying gestures into separate hand-designed components like hand detection, background removal using segmentation, and then classifying the gestures instead of a single end to end deep learning system was justified. This made our system robust in terms of achieving accurate results (shown in 4.2) as well as usable in real-time which was the main reason behind building this system.

5. CONCLUSION

In this paper, we implemented an American Sign Language translation system and were able to achieve the results in a real-time environment which is the most important use case of the translation system. While doing so the model was very much robust and hence was able to balance out the trade-off between the two. Our approach of extracting features of the hand using Morphological transforms reduced the task of the classifier. However, this requires the user to operate the application at least in a moderately illuminated environment. In a poorly lit environment, there would be situations that few gestures would be misclassified as the morphological transform would be unable to produce contrast features. Spell Corrector was able to counter this drawback somewhat which corrected the word that was inaccurate by 1 or 2 alphabets. Thus, the whole pipeline of modules made our system quite efficient as well as fast. This work can be further extended to classify not just ASL alphabets and letters but also the predefined gestures for some of the commonly found ASL words. Since we were able to achieve the results in real-time, this could be integrated with Video Conferencing tools which would help the deaf and mute people communicate with the rest of the community.

6. ACKNOWLEDGMENTS

We would like to express our gratitude towards our Head of Department, Dr. Meera Narvekar and our Principal, Dr. Hari Vasudevan for their support in terms of access to library content and the opportunities provided to us.

7. REFERENCES

[1] Vedak, O., Zavre, P., Todkar, A., Patil, M., 2019. Sign Language Interpreter using Image Processing and Machine Learning. International Research Journal of Engineering and Technology (IRJET).

[2] Bheda, V., Radpour, D., 2017. Using Deep Convolutional Networks for Gesture Recognition in American Sign Language. arXiv preprint arXiv:1710.06836.

[3] Garcia, B., Viesca. S.A., 2016. Real-time American Sign Language Recognition with Convolutional Neural Networks. In Convolutional Neural Networks for Visual Recognition at Stanford University.

[4] Masood, S., Thuwal, H.C., Srivastava, A., 2018. American Sign Language Character Recognition Using Convolution Neural Network. In Smart Computing and Informatics (pp. 403-412). Springer, Singapore.

[5] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., Berg, A.C., 2016. SSD: Single Shot MultiBox Detector. Proceedings of the European Conference on Computer Vision (ECCV).

[6] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., &Andreetto, M., Adam, H., 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. ArXiv, abs/1704.04861.

[7] Anh, N.T.L., Kim, Y., Lee, G., 2012. Morphological gradient applied to new active contour model for color image segmentation. Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication.
https://doi.org/10.1145/2184751.2184778

[8] Bhaire, V.V., Jadhav, A.A., Pashte, P.A., Magdum, P.G., 2015. SPELL CHECKER. International Journal of Scientific and Research Publications.

[9] Janocha, K., Czarnecki, W. M., 2017. On loss functions for deep neural networks in classification. arXiv:1702.05659.

[10] Kingma, D.P., Ba, J., 2015. Adam: A Method for Stochastic Optimization. CoRR, abs/1412.6980.

[11] Nagi, J., Ducatelle, F., Di Caro, G. A., Ciresan, D., Meier, U., Giusti, A., Nagi, F., Schmidhuber, J., Gambardella, L. M., 2011. Max-pooling convolutional neural networks for vision-based hand gesture recognition. 2011 IEEE International Conference on Signal and Image Processing Applications, ICSIPA 2011.342-347.10.1109/ICSIPA.2011.6144164.

[12] Agarap, A.F., 2018. Deep Learning using Rectified Linear Units (ReLU). ArXiv, abs/1803.08375.

[13] Sultana, F., Sufian, A., Dutta, P., 2018. Advancements in Image Classification using Convolutional Neural Network. 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), Kolkata, India, pp. 122-129.

[14] Mavani, V., Gurnani, A., Shah, J., 2017. A Novel Approach for Image Segmentation based on Histograms computed from Hue-data. arXiv preprint arXiv:1707.09643.