Sign Language to Speech Conversion

Joydeep Chatterjee
Department of Applied Optics and Photonics, Technology Campus,
University of Calcutta, Salt Lake, Kolkata, India,
coolfire7139@gmail.com

Suman Kayal
Department of Computer science and engineering
Narula Institute of Technology
Kolkata, India
49sumankayal@gmail.com

Aparna Shaw
Department of Computer science and engineering
Guru nanak institute of technology kolkata, India
aparnashaw17@gmail.com

Annesha Nandi
Department of Computer Science and Engineering
Guru Nanak Institute Of Technology
Kolkata, India
anneshanandi175@gmail.com

Kushal Karmakar
Department of Information Technology
Guru Nanak Institute Of Technology
Kolkata/INDIA
kushaljeet396@gmail.com

Abstract: Creating a model that uses a computer's webcam to capture a person signing gestures for American sign language (ASL), and translate it into corresponding text and speech in real time. The translated sign language hand gesture will be acquired in text which is further converted into audio. In this manner we are implementing a fingerspelling sign language translator. To enable the detection of gestures, we are making use of a Convolutional neural network (CNN). A CNN is highly efficient in tackling computer vision problems and is capable of detecting the desired features with a high degree of accuracy upon sufficient training.

Keywords: Deep Learning, Convolutional neural network.

1. Introduction

Sign language is important for the deaf and hard of hearing community as it allows individuals to communicate through a combination of hand signs, facial expressions and other body movements. Despite its importance, sign language users often find it difficult to communicate with those who don't understand the language. This can be addressed by promoting inclusivity and accessibility.

This can only be assured by being inclusive and accessible.

One such way is the use of technology that reads signs and then returns them as spoken words [1-9]. This would involve recognition of sign language hand gestures and translation into audible speech, and convolutional neural networks have been proposed as one of the best ways of translation given their excellent abilities in image/video sequence recognition/classification.
American Sign Language (ASL) is a natural language that serves as the predominant sign language of deaf communities in the United States and is most commonly used sign language. The focus of this project is to convert ASL sign language into audible speech.

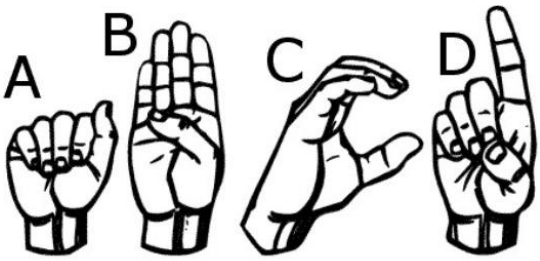

Fig. 1: *First four letters of ASL*

Motivation and Contribution

Motivation:

Communication Barriers: This addresses the huge communication difficulties between those who use sign language and hearing people, leading to loneliness and misinterpretation of gestures.

Inclusivity and Accessibility: As a means of making life more inclusive in education, healthcare, employment among others through smoother interaction.

Technology Advances: Building up on strides in deep learning and computer vision to come up with practical solutions for the deaf and those with hearing impairment, making them applicable in real life.

Social Integration: This is all about bridging the social gap while creating a gracious, harmonious society for everyone to participate in fully.

Contribution:

Real-time Translation: Making systems which can translate sign languages into spoken words instantly thereby improving spontaneous communication.

Educational Tools: Creating instruments that foster teaching and learning sign languages thus increasing its knowledge among non-deaf communities.

Healthcare Improvement: To better medical communications, this will ensure timely transmission of appropriate medical information to persons who are deaf or hard of hearing

Increased Employability: By enhancing their ability to communicate with colleagues, this improves job prospects and workplace integration for those using sign languages.

2. Principles and Methodology Used

a. Data Pre-processing

Data Collection:

The data set consists of images representing each letter of the English alphabet, together with the digit ‘0’ which stands in place of a space. These images were arranged systematically in folders corresponding to each letter (‘A’, ‘B’, ‘C’, …, ‘Z’) and ‘0’.

Data loading:

Images are imported from their respective folders using a glob function that accepts file paths based on the specified pattern. These methods were stored in a dictionary, with keys associated with the labels of the images.

Image Resizing:

All images used in this work have been resized to the standard dimension of 180x180 pixels to provide standard input to the model. In this way, uniformity was provided to the dataset so that processing images through the model was more feasible and thereby easier to learn from.

Label Encoding:

Each character can be mapped against a numeric character, and this data can be converted into some interpretable format. For example, 'A' is written as 1 and 'B' as 2. By continuing this process, '0' comes to be written as 0.

Data Splitting:

The dataset was then split into training and test sessions in an 80:20 ratio. This is the common way to analyze the model performance on unseen data: 80% for training and 20% for testing.

b. Proposed Method

CNN Model Architecture:

Classification was done using a Convolutional Neural Network because it works very well with images. The model architecture includes:

- **Input Layer:** The architecture is such that the input image size is 180x180 pixels, with 3 color channels.
- **Convolution Layers:** There are three convolutional layers as follows—
    - The first Convolution layer uses 16 filters with a kernel of 3x3, followed by ReLU as an activation function and max-pooling to reduce spatial dimensions.
    - The second layer has 32 filters, again followed by ReLU activation and max-pooling.
    - The third layer uses 64 filters, with similar operations and max-pooling.
- **Flatten Layer:** The output from the convolutional layers is then flattened into a 1D array.
- **Dense Layers:** This flattened data is then passed through two fully connected layers:
    - The first dense layer contains 128 neurons with ReLU activation.
    - The final dense layer outputs predictions across 27 classes—one for each letter and one for space.

Model Compilation:

The model was then compiled using the Adam optimizer, which provides fast training. The loss function used was Sparse Categorical Crossentropy with logits because the labels were integer-encoded. The metric of interest was accuracy.

Training:

The model ran for 2 epochs, which means it was trained on the training dataset twice. That training was conducted using the fit function from Keras—sort of a standard way to train machine learning models in Keras.

Prediction and Inference:

After training, test data were passed into the evaluate function to measure the model's performance. This was to demonstrate the accuracy of the model and the level of surety it had in generalizing its knowledge.

Real-Time Application:

For real-time application, a webcam feed is integrated, and flipped horizontally to offer a mirror view for user convenience. A Region of Interest on the screen will help the user place their hand in a correct way to acquire images.

For the ROI of a captured image, the steps included resizing, converting it into grayscale, applying Gaussian blur, adaptive thresholding, and seeking to enhance the outline of the hand. A processed image passed through the trained model to predict the corresponding letter. Then, the predicted letter is converted to speech using the gTTS library to further the interactive experience

by providing auditory feedback. The results were displayed visually alongside the captured image.

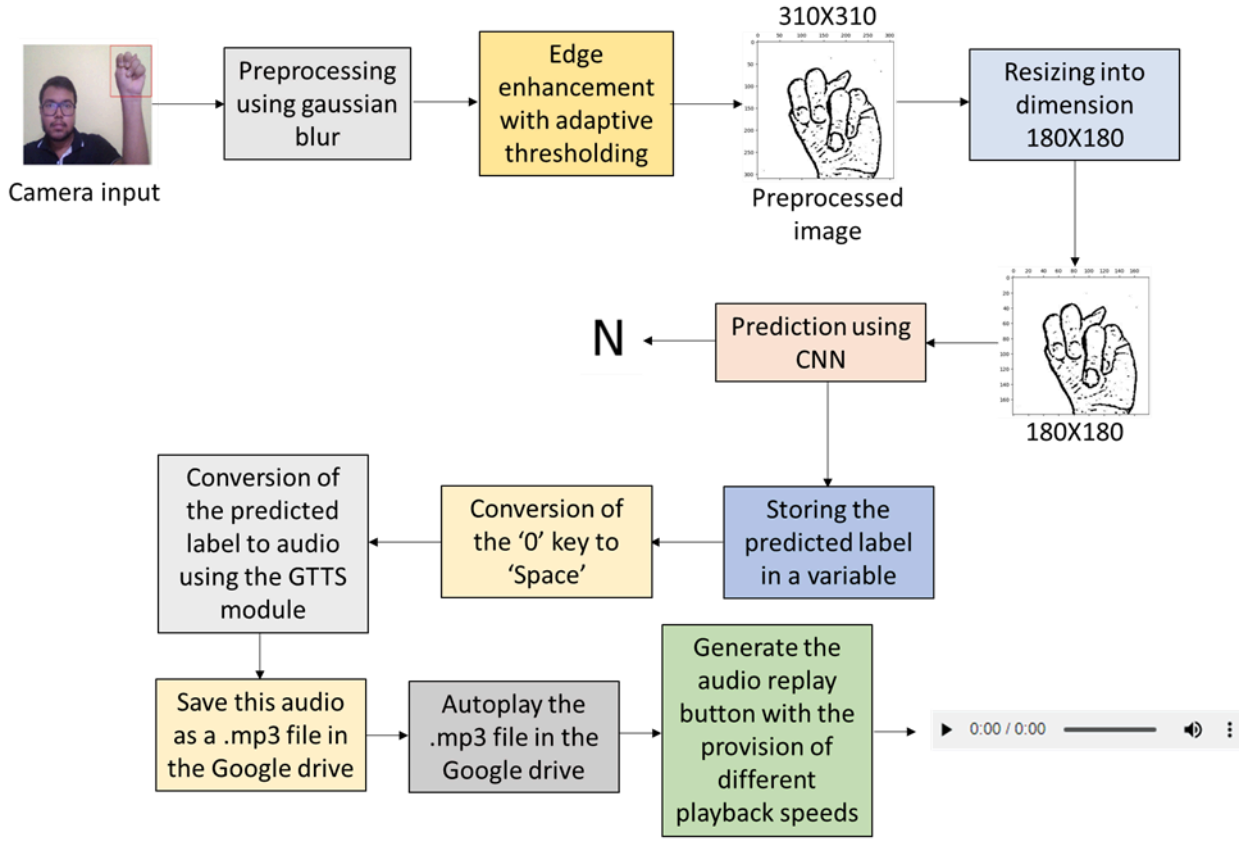

Fig. 2: *The workflow of proposed Methodology*

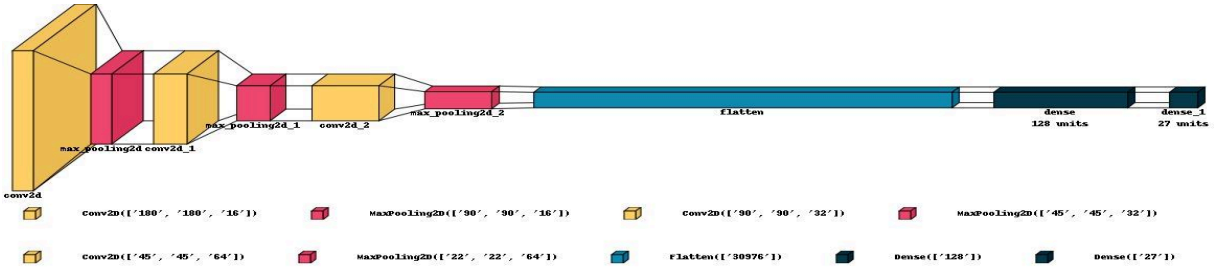

Fig. 3: *The used CNN Model*

3. Experimental Results and Discussions

3.1 Experimental Setup

The experimental arrangement is quite straightforward. It consists of a simple webcam and a computer system. The processing is done in the computer with the Python programming language. The workstation used is a machine with 12 GB RAM with Intel i3 processor.

3.2 Dataset Preparation

The proposed setup has been trained and tested on a sign-language image dataset [10]. Out of the total images, 13690 images (80 %) of this dataset are reserved for training the model and the remaining 3423 image data (20 %) are utilized for the testing purpose. The dataset contains the edge-enhanced hand-signs with set brightness threshold acquired through the standard webcam. It reduces the operational complexities of the image to a large extent (e.g. the challenge of dealing with the wide-range of brightness-variation in the portions of the images other than the edge, the noise and artifacts having the brightness values below the set threshold, interference of any undesired background data etc.). The dimensions of the images in the original dataset have the dimension of 310 pixels X 310 pixels. To reduce the computational load, it has been reduced to 180 pixels X 180 pixels. The dataset contains the signs standing for the letters of the English alphabets and no sign in uniform background denoting the space. Hence, there are 27 unique classes in the present dataset.

Apart from this dataset, some real-time snapshots of the hand-signs were captured through a webcam and they were preprocessed as mentioned before for checking the robustness of the proposed scheme. After capturing the image, the area of interest was filtered out by defining a ROI box.

The current architecture comprises 3 pairs of 2D convolution layers placed one after another. A flattening layer and two dense layers follow them. The ReLU (rectified linear unit) activation function is employed.

3.3 Evaluation Metrics

The optimizer and the loss-function used are the well-known ‘Adam’ optimizer and the ‘Sparse Categorical Cross Entropy respectively. The accuracy parameter has been monitored for the evaluation of the training mechanism. The built model exhibits 99 % accuracy for both the training and the test data. The number of epochs used during the training phase is 2.

3.4 Result

The detected letters are displayed on the output images. The predicted text is converted to speech using the GTTS (Google Text-to-Speech) module. In other words, the current system has the facilities of both the sign language to text as well as sign-language to speech making it suitable for both visually and audibly challenged people.

Fig. 4 (a-h) shows the prediction of different letters for the text “NEW HOME” using the test images from the dataset. The audio is spoken out while executing the code. It can be run again by clicking the play button shown on top of each of the image frames. Fig. 4 (i) shows the capability of the system for the word-level predictions.

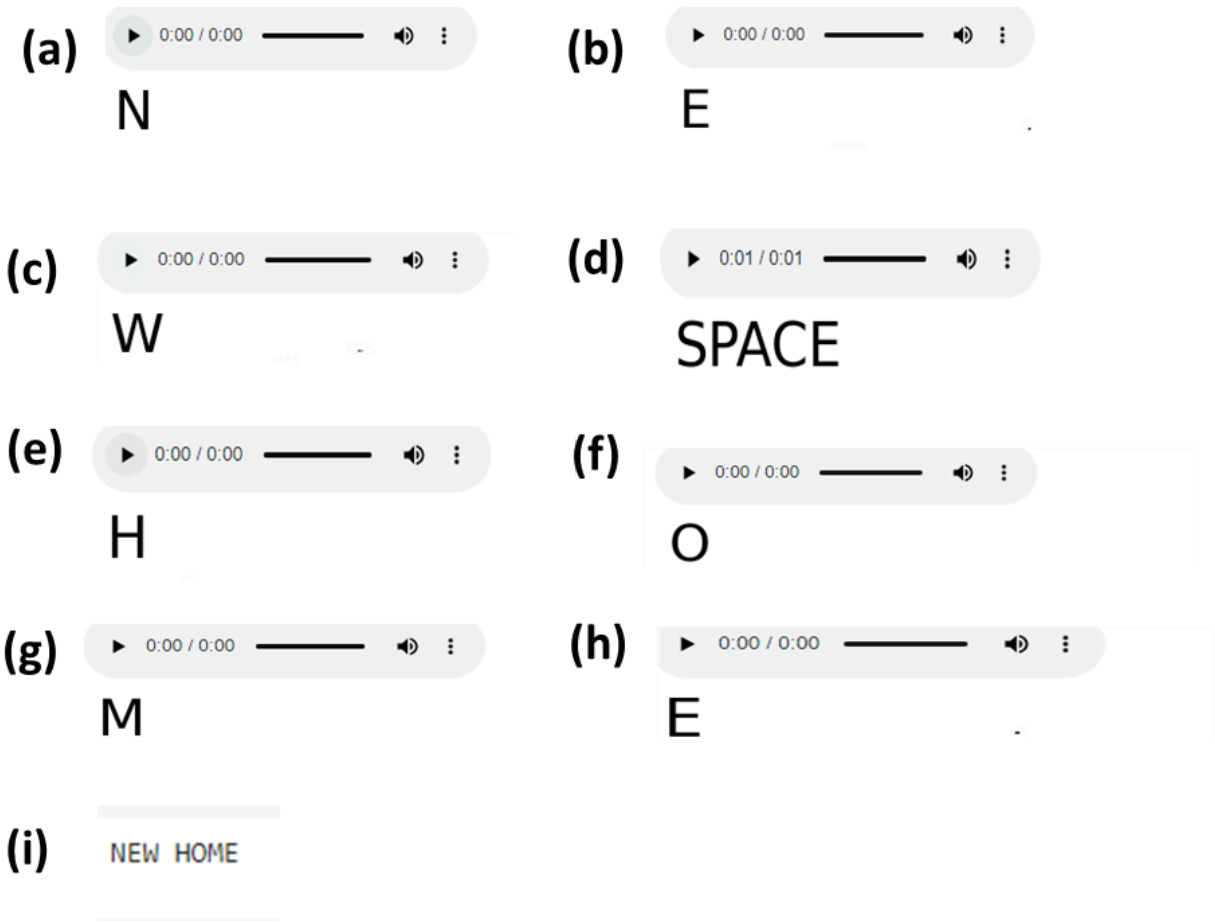

Fig. 4: *(a-h) The predictions for the different letters of the text “NEW HOME” with audio options and (i) the predicted text using the test images from the dataset*

Fig. 5 (a-d) shows the predictions with speech for the different letters read from the hand-signs captured in an interactive real-time mode where the user can input the data letter-by-letter and the termination of the text can be decided by the user. Fig. 5 (e) presents the reconstructed text.

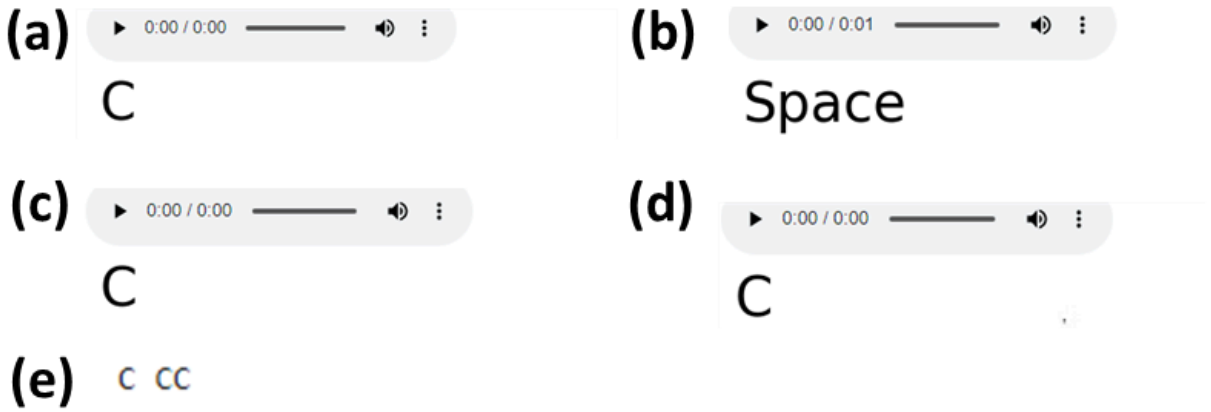

Fig. 5*: (a-d) The predictions for the different letters with audio options using snapshots captured in real-time in an interactive mode and (e) the predicted text using the real-time images*

From the results, it is quite evident that the accuracy of the system is good enough. It works extremely well with the original dataset. But, some wrong predictions can be encountered from the real-time image because of the non-uniformity in the brightness distributions over the field, noise, magnification etc.

3.5 Data Augmentation Result

No data augmentation was required as the number of available samples were sufficient in the dataset.

4. Comparative Performance Analysis

Table 1: *Evaluation of the performance of the proposed model in terms of standard metrics like Accuracy, Precision, Recall, F1 Score and Matthews Correlation Coefficient*

Metric	Score
Accuracy	0.999416
Precision	0.999365

Recall	0.999451
F1 Score	0.999406
MCC	0.999393

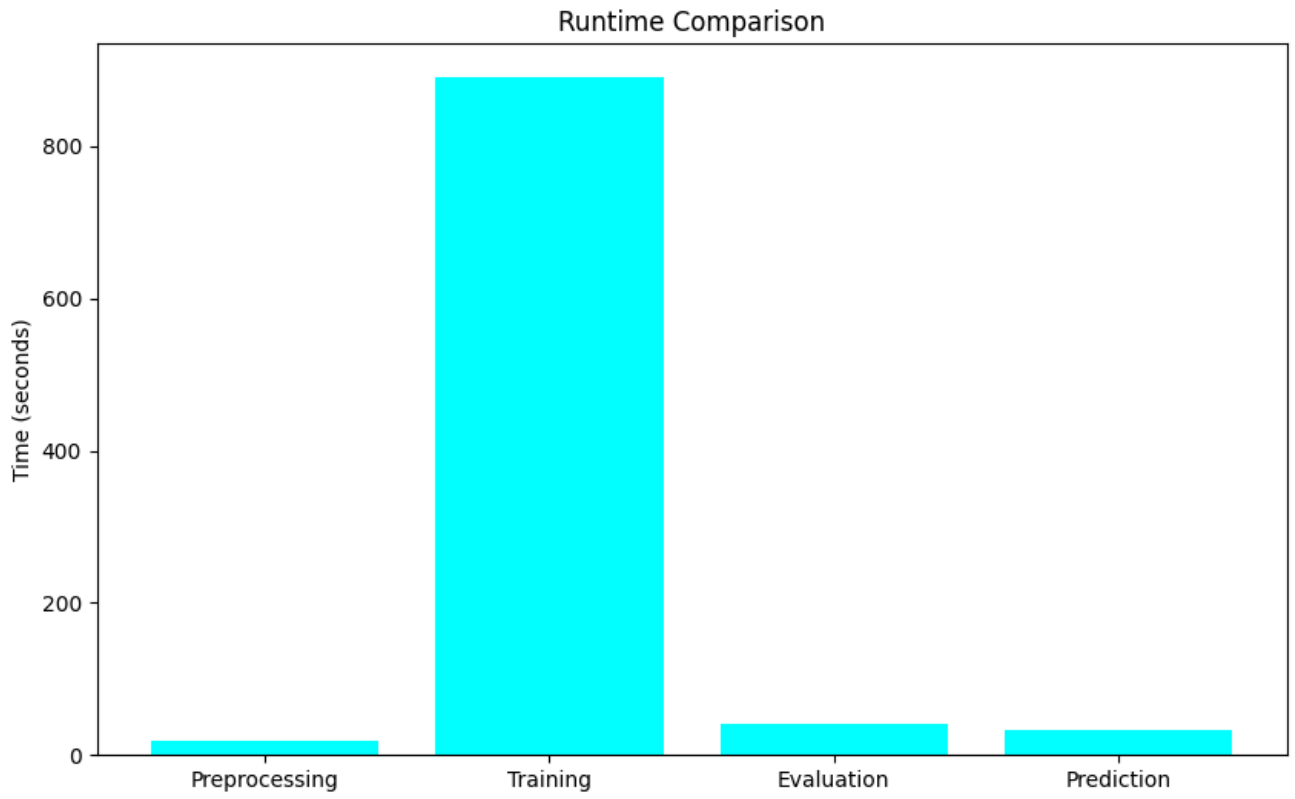

Fig. 6: *A bar-graph Comparing the runtime of different steps used in this project*

Table 2: *A table Comparing the runtime of different steps used in this project*

Methods	**Runtime (seconds)**
Preprocessing	16

Training	52
Evaluation	2
Prediction	32

Table 3: *Comparisons of the different standard metrics form different CNN architectures*

METHODS / METRICS	BASIC CNN	RESNET	EFFICIENTNET	XCEPTION
ACCURACY	0.999415717	0.999415717	0.999416	0.999416
PRECISION	0.999365434	0.999478352	0.999478	0.999478
RECALL	0.999451303	0.99936143	0.999361	0.999361
F1-SCORE	0.999405975	0.999415265	0.999415	0.999415
MATHEWS CORRELATION COEFFICIENT	0.999393312	0.999393485	0.999393	0.999393

Table 4: *Comparisons of the runtimes in phases for different CNN architectures*

METHODS / RUNTIMES (IN SECONDS)	BASIC CNN	RESNET	EFFICIENTNET	XCEPTION
TRAINING	52	157	64	351
EVALUATION	2	6	17	19
TESTING	32	5	17	2

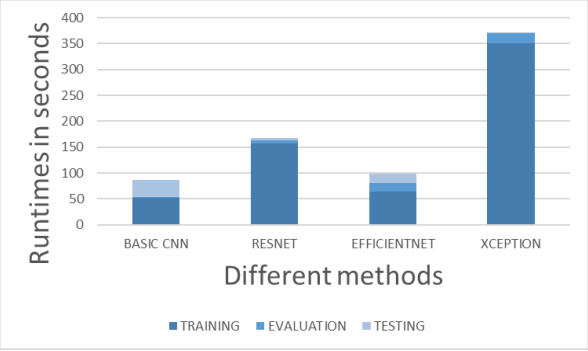

Fig. 7: *A plot comparing the runtimes in different phases for the different methods*

5. Conclusion and Future Scope

5.1. *Limitations*

Shortcomings of this sign language-to-speech conversion project using CNN: In this, the following section, I present some of the effects that might alter the results and prove to afflict the performance of the system. Measuring the challenges faced by addressing such effects can be possible, which could also form a possible line of future work.

1.Dataset Limitation:

The project made use of pre-made datasets as opposed to self-collected datasets. As such, the level of assurance in attaining the level of accuracy required for the real-time applications of life could not be accorded in the performance of the model. The reason, by design, the project dataset could not capture all the variability that exists in natural sign language gestures done in real time, hence impacting its generalizability.

2. Issues Associated with Real-Time Analytics:

Some discrepancies in light intensity between the real-time analysis environment and the conditions in which the sample dataset was taken added to issues in achieving uniform performance in real-time applications. In fact, it has been found that under misfit environmental conditions, model accuracy may waste to behave accurately and reliably in real application scenarios.

3. High-level Accuracy with Respect to Test Data:

The accuracy for the model in using the test sample data was excellent at 99%, but this definitely could not be applied in reality under any conditions, as explained in the preceding section. This is bound to happen because the dataset has limitations, and as to the condition of the environment, real-world conditions were not considered in it.

4. Computational Resource Constraints:

Since all the images would have had to be resized up to the same size to form a sample dataset since there were very few computational resources available, this resize would most probably have compromised the quality of the input images and actually led to a poor model performance.

5. No Image Normalization :
Image normalization was not performed due to the lack of sufficient computational resources. Normalized inputs help in providing uniformity under the given input conditions; the model might have performed and generalized poorly if normalization was not done.

6. Background Consistency Issues:
Most of the images in the sample dataset had black backgrounds; hence, the difference of the image from reality was significantly large. This may have posed another challenge to the model with generalization in diverse environments because of the lack of normalization done on the image.

7. Little Variability in Gestures:

In the dataset, inadequate diversity in hand shapes, variations in signs, and different signing styles of individuals can exist, which may seriously hamper the model's strength during recognition and conversion of a wide range of signs.

8. Real-Time Latency:

Latency issues will generally be a consequence of the very high computational complexity in the CNNs. As a result, the system will not be able to be so responsive in real-time applications. Research on latency control will arise after the implementation of CNNs in practice.

A challenge of making this system scalable for multiple sign languages or dialects would be in its capacity to handle more sign languages and different variations at a regional level. Therefore, the scope of its strength should be propped up further.

Little has been done in the testing and optimization of this user interface where real-time interaction is related. Much optimization is left regarding the user experience, including responsiveness or intuitiveness of a given interface to make it useful during practical application.

5.2. Future Scope

Some of the improvements that can be done in order to make it possible for the system to be effective toward converting the sign languages into speech are as follows

1. Custom Dataset Collection:

Will develop and work with training of a custom set of data, including the sign language, which contains more sign variation in signs for effective application of the model in the real world.

2. Real-Time Adaptation Techniques:

Nature-adaptive algorithms must be implemented to detect changes in illuminations and environment conditions, thus expanding maybe even toward real-time image normalization, ideally supported with data augmentation for better performance consistency.

3.Enhanced computational resources:

This increase in computational power is most likely the use of more images, highly imaged in resolution, coupled with advanced available techniques of image processing, including the suppression of noise and normalization caused by the visual data. Advanced Noise Reduction: For noise elimination, the very best of the state-of-the-art algorithms should be used to give better quality images that are clear. It will help capture and process sign language gestures correctly.

5. Edge Detection and Preservation:

Implementable methods of edge information preservation in image resizing. These techniques could include edge-preserving filter methods or super-resolution methods that assist in making gesture detection more accurate.

Acknowledgement

This work has been partially supported by the MeitY Project, MHRD, Govt. of India. The authors also thank RCC Institute of Information Technology., Kolkata.

Conflict of Interest:

The authors hereby state that there is no conflict of interest.

Declaration of Interest. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

References

[1] A. Ojha, A. Pandey, S. Maurya, A. Thakur, and P. Dayananda, “Sign Language to Text and Speech Translation in Real Time Using Convolutional Neural Network,” Spec. Issue - 2020 Int. J. Eng. Res. Technol., vol. 8, no. 15, pp. 191–196, 2020.

[2] P. Karthick, N. Prathiba, V. B. Rekha, and S. Thanalaxmi, “Transforming Indian Sign Language into Text Using Leap Motion,” Int. J. Innov. Res. Sci. Eng. Technol., vol. 3, no. 4, pp. 10906–10910, 2014, [Online]. Available: www.ijirset.com

[3] P. V. V. Kishore and P. R. Kumar, “Segment, Track, Extract, Recognize and Convert Sign Language Videos to Voice/Text,” Int. J. Adv. Comput. Sci. Appl., vol. 3, no. 6, pp. 35–47, 2012.

[4] A. Shinde and R. Kagalkar, “Sign Language to Text and Vice Versa Recognition using Computer Vision in Marathi,” Int. J. Comput. Appl., vol. 0975 – 888, no. NCAC 2015, pp. 23–28, 2015.

[5] V. A. Adewale and A. O. Olamiti, “CONVERSION OF SIGN LANGUAGE TO TEXT AND SPEECH USING MACHINE LEARNING TECHNIQUES,” J. Res. Rev. Sci., vol. 5, pp. 58–65, 2018, doi: 10.36108/jrrslasu/8102/50(0170)Volume.

[6] C. M. Travieso, J. B. Alonso, and M. A. Ferrer, “Sign Language To Text By SVM,” in Seventh International Symposium on Signal Processing and Its Applications, Paris: IEEE, 2003, pp. 435–438. doi: 10.1109/ISSPA.2003.1224907.

[7] J. Pu, W. Zhou, J. Zhang, and H. Li, “Sign Language Recognition Based on Trajectory Modeling with HMMs,” in Lecture Notes in Computer Science, vol. 9516, G. Goos, Ed., Springer, 2016, pp. 686–697. doi: 10.1007/978-3-319-27671-7 58.

[8] C. Sun, T. Zhang, B. Bao, and C. Xu, “LATENT SUPPORT VECTOR MACHINE FOR SIGN LANGUAGE RECOGNITION WITH KINECT,” in 2013 IEEE International Conference on Image Processing, Melbourne: IEEE, 2013, pp. 4190–4194. doi: 10.1109/ICIP.2013.6738863.

[9] R. S. L. Murali, L. D. Ramayya, and V. A. Santosh, “Sign Language Recognition System Using Convolutional Neural Network And Computer Vision,” Int. J. Eng. Innov. Adv. Technol., vol. 4, no. 4, pp. 137–142, 2022, doi: 10.1109/ICATIECE56365.2022.10046883.

[10] N. Gupta, “Sign Language To Text Conversion Dataset.” Accessed: Aug. 11, 2024. [Online]. Available: https://github.com/emnikhil/Sign-Language-To-Text-Conversion/tree/main/dataSet