MotionScript: Sign Language to Voice Converter

Nidhi Kadam
Thadomal Shahani Engineering College, Mumbai University, India

Chaitanya Kakade
Thadomal Shahani Engineering College, Mumbai University, India

Vishal Kaira
Thadomal Shahani Engineering College, Mumbai University, India

ABSTRACT
Sign language serves as a vital mode of communication for the deaf and mute community, yet it presents a significant barrier in their interaction with the larger society, which often lacks proficiency in sign language. This paper presents MotionScript, an innovative sign language to voice conversion system that leverages computer vision, deep learning, Convolutional Neural Networks (CNN), Natural Language Processing (NLP) and Large Language Model (LLM) to facilitate interaction between individuals from the deaf and mute community and the rest of the world. This paper outlines a thorough comparison of four distinct neural network models, utilizing metrics to identify the most accurate model for transforming American Sign Language (ASL) into coherent and meaningful sentences voiced in natural language. This conversion process incorporates essential components such as autocorrection and the integration of a large language model.

General Terms
Computer Vision, Movement Detection.

Keywords
American Sign Language (ASL), Convolutional Neural Network (CNN), Google Text-To-Speech (gTTS), Large Language Model (LLM), Long Short-Term Memory, Machine Learning, Natural Language Processing (NLP), Real-time Conversion, Recurrent Neural Networks (RNN), Residual Networks (ResNet), Stochastic Gradient Descent (SGD), Visual Geometry Group (VGG16).

1. INTRODUCTION
Sign language, a visual language of gestures and facial expressions, plays an important role in the communication process of deaf and mute individuals. However, for those who are not proficient in sign language, it often creates a barrier in their interactions with the society. This communication gap has persisted for generations, limiting the active participation of the deaf and mute community in various aspects of life, from education and employment to healthcare and social engagement.

As an attempt to resolve this issue, our research introduces MotionScript, a sign language to voice conversion system that harnesses the power of cutting-edge technologies. This system utilizes computer vision, machine learning, NLP, and language modelling techniques to bridge the communication gap between sign language and spoken language, enabling real-time and accurate conversion of sign language gestures into voice. MotionScript holds the promise of revolutionizing the way deaf and mute individuals communicate with the world.

The development of MotionScript involved the integration of several technologies. Computer vision, CNN, is employed to detect and track hand movements. These are fundamental components of sign language and, when interpreted accurately, can convey messages. Meanwhile, Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks were implemented to handle the sequential nature of sign language, ensuring that signs are understood within their linguistic context. NLP techniques and LLM, especially T5-Flan Large were incorporated to convert the detected signs into coherent and contextually relevant voice.

2. LITERATURE REVIEW
This section dives into the extensive field of previous research within the field of sign language recognition and its translation to voice. Notably, CNN has emerged as a well-known technique for ASL recognition. Lin et al. [1] made use of advanced image segmentation techniques to isolate the hand within a particular input image, followed by skin modeling and image centering around the primary axis. Yadav et al. [2] focused on translating ASL in real-time. They used camera captured images, preprocessed them, and utilized techniques like hand gesture scans to amplify the gestures. These preprocessed images were introduced in CNN model for label prediction. The predicted label gave an accuracy of 95.8%.

Garcia et al. [3] demonstrated the use of real-time sign language translator using pre-trained GoogLeNet architecture, showing accurate classification of letters 'a' to 'e' even for first time users. Kumar et al. [4] employed time series oriented neural networks for sign language conversion. Efforts to develop lightweight ASL classification models have also been underway, as seen in [5] and [6]. These activities aimed to streamline the deep neural network architecture to reduce computational costs. They utilized EfficientNet models and generated lightweight deep learning models. They have also tried various other techniques like LSTM-RNN and Hidden Markov Models (HMM).

Truong et al. [8] have utilized different techniques like AdaBoost and Haarcascade classifiers and trained their models on ASL dataset. Another concept that has been used is HMM by Yang et al. [9]. Dynamic gestures are addressed in this approach. Tracking of skin color blobs corresponding to the hands into a body-facial space, centered on the user's face to extract gestures from successive video frames was performed. LSTM-RNN techniques adopted by [10, 11] helped in recognizing 26 alphabets from corresponding hand signs. They have extracted features like finger positions, sphere radius and angles between fingers for classification of models. Researchers have tried to build models for different languages such as Arabic, Indian, and other language that exist to recognize them and correctly classify them. In [12] they have used ResNet152 model for classifying hand gestures and used pre-trained deep neural networks to recognize Arabic sign language. An approach to produce synthetic animation was adopted to translate Malayalam language into Indian Sign Language (ISL) as discussed in [13]. HamNoSys (Hamburg Notation System) was utilized for intermediate representation of sign language. In this approach the system accepts the set of words and generates an animated section from the set of words. This system then turns the words into the Hamburg Notation System. The Kerala government has implemented a program

designed to comprehensively parse all developments related to the teaching of sign language and promoting subtle awareness as per [14]. This initiative is predominantly based on the Spanish language, facilitating the conversion of basic words into Spanish. This approach proves advantageous for Spanish-speaking individuals with hearing impairments, as it enables them to grasp sign language more quickly, with the conversion occurring into Spanish instead of English, which is commonly used for ASL.

D. Kelly et al. [15] have introduced a continuous system that utilizes a sequence of sign language gestures to generate an automated training set and extract sign patterns from that set. Their system monitors sentences, determines associated compound sign gestures, and employs instance learning with a density matrix technique for the supervision of noisy texts. Segundo et al. [16], have conducted various experiments to develop a statistical model for converting speech data into sign language for deaf individuals. An animated presentation and a statistical translation module were employed to automate speech recognition. Your provided sentences are generally correct, but there are a few minor adjustments for clarity and completeness.

In a study by Shivashankara et al. [17], the authors explored an optimal approach towards ASL recognition. They employed an efficient system for recognizing ASL gestures. S. Liu et al. [18] addressed the challenge of image classification with a small training sample size. Their work focuses on the adoption of very deep convolutional neural networks for improved image classification, even when faced with limited training data.

Several studies have shown the efficiency and accuracy of CNNs for ASL classification. The synthesis of these diverse studies provides a foundation for the development of a dependable and effective sign language recognition system with potential applications in real-world scenarios for the benefit of the deaf and mute community.

3. DATASET

The ASL [19] dataset consists of 69600 images which was used to train, validate, and test the deep learning neural network models for accurate sign language to coherent text conversion. The images from the actual dataset are shown in Figure 1. This dataset has 29 class labels (‘A’ to ‘Z’, an option for ‘space’, ‘delete’ and ‘nothing’) and for the training purpose letter ‘A’ corresponds to class value 0, ‘B’ corresponds to class value 1, and so on, while ‘space’, ‘delete’ and ‘nothing’ corresponds to class values 27, 28 and 29 respectively. To speed up the training process while maintaining important features of the pixels, the images were downsized to 64 by 64 size. Training set consisted of 55680 images and testing set consisted of 13920 images both belonging to above mentioned 29 classes.

Figure 1. Letter ‘H’ and Letter ‘Y’

4. METHODOLOGY

The following section explains the novel methodology that has been undertaken to convert American sign language into coherent and meaningful sentences in the form of voice.

4.1 Image Processing

To enhance the training process without losing important details in the sign language images, the input images were resized to a compact 64 by 64-pixel size format. This not only made the training process faster but also kept the key aspects of sign language gestures intact. This resizing not only accelerated the training phase but also ensured that crucial details in the sign language gestures were preserved. Another important consideration in our data preprocessing pipeline was to mitigate the risk of encountering the "exploding gradients" issue often associated with convolutional neural networks and transfer learning models. To address this concern, an important step of rescaling the image data to fall within the range of 0 to 1 was employed. By bringing the pixel values into this standardized range after loading them into the NumPy arrays, numerical stability during training was maintained, preventing potential gradient related challenges. Figure 3. in our framework visually outlines the series of steps involved in processing the sign language images, ensuring that our model receives clean, appropriately sized, and rescaled data for effective training. Additionally, it is worth noting that data augmentation techniques were implemented to populate the dataset further.

Data augmentation involves applying various transformations to the existing images, such as rotation, scaling, and horizontal flipping, to create additional training examples. Figure 2 depicts images after the application of data augmentation technique. This approach not only diversifies the dataset but also enhances the model's ability to generalize and recognize sign language gestures under different conditions. Data augmentation technique was a valuable tool for improving the robustness and accuracy of the proposed sign language recognition system. Appropriate augmentation values were thoroughly analyzed and selected with respect to the model architecture and behavior as discussed in section 4.3.

Figure 2. Letter ‘H’ and Letter ‘Y’ after augmentation

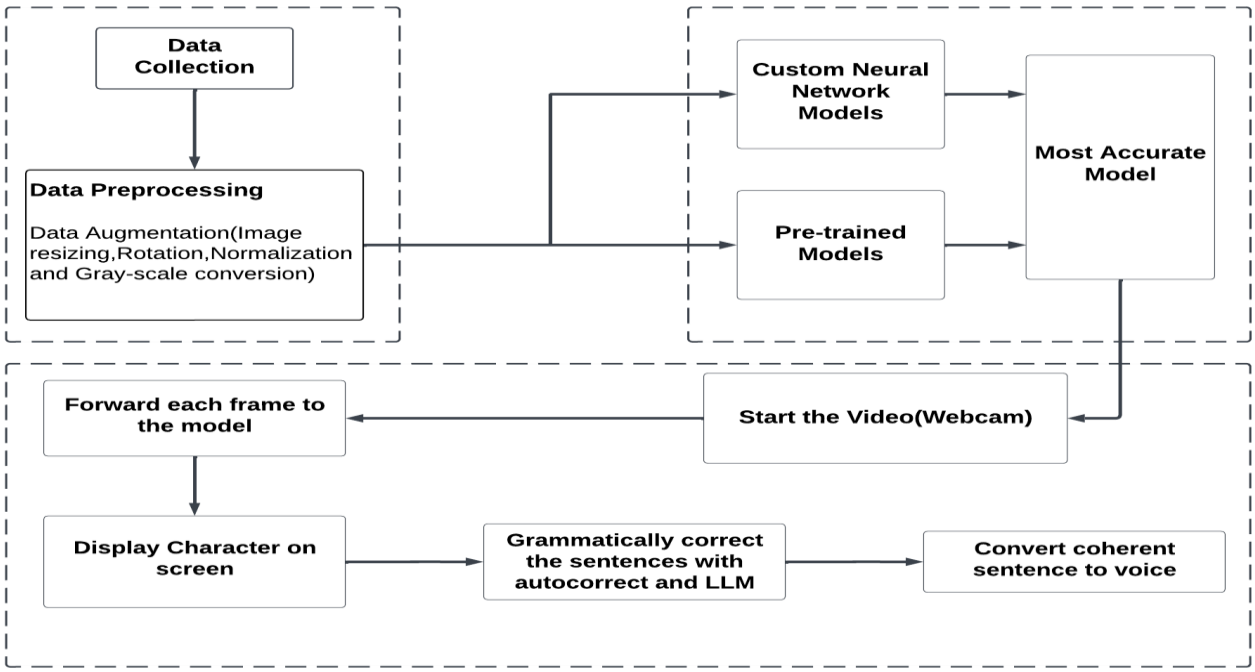

Figure 3. Overall Process Flow

4.2 Model Architecture

The overall model architecture as represented in Figure 3. suggests the flow of the approach. First, the dataset was collected and then augmented using various data augmentation techniques such as rotation range, height shift range and width shift range.

The augmented data was introduced in various neural network architectures and the architecture with the most promising metrics was selected and downloaded. The loaded model was incorporated in a webcam and every single frame from the input video was transferred to the model to perform a prediction of the character. The predicted character was then displayed on the screen to form a sentence or a word which was then corrected using the transformer library (T5Tokenizer and T5ForConditionalGeneration). The corrected sentence was then converted into voice using the gTTS (Google Text to speech) API.

4.3 Model Training

The model is trained on a dataset which consists of 29 different classes with a total of 69600 diverse images of each character present in the English alphabet. Each class contains approximately 2400 images taken under different lighting conditions. The entire dataset was divided into two categories, training data and testing data. The train data had 55,680 images and the test data had 13,920 images. The total number of trainable parameters were around 21,26,365.

4.3.1 Proposed Model 1

The Sign language to voice Converter utilized a carefully crafted CNN architecture. The architecture comprised multiple convolutional layers with batch normalization, dropout layers to mitigate overfitting, and max-pooling layers for effective feature extraction and spatial down-sampling. The final layers include densely connected units with SoftMax activation function, enabling precise multi-class classification. Training efficiency is enhanced through strategic callbacks, including Model Checkpoint for preserving the best model. ReduceLROnPlateau for dynamic learning rate adjustment, and Early Stopping to prevent overfitting. During training, the model undergoes 50 epochs with the Stochastic Gradient Descent (SGD) optimizer and categorical cross-entropy loss. The training progress is continuously monitored, and the model's performance is rigorously evaluated on the validation set, capturing accuracy and loss metrics. This holistic approach, combining a thoughtful CNN architecture and effective callback strategies, resulted in a robust and relatively accurate Sign Language classifier capable of accurate and efficient gesture interpretation and recognition. The training and validation accuracies and losses are visually depicted in Figure 4. and Figure 5. respectively.

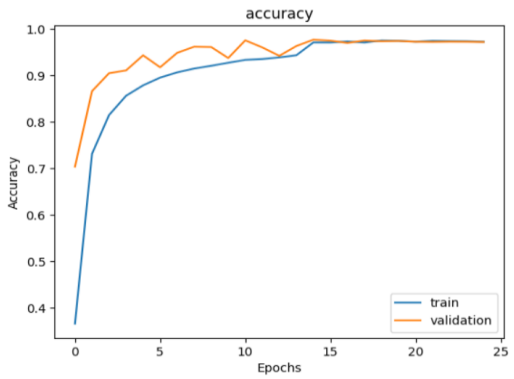

Figure 4 Train and validation accuracy

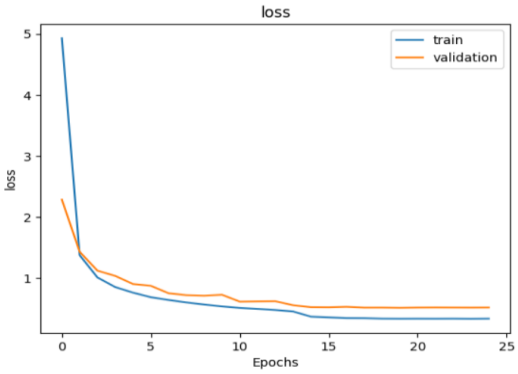

Figure 5. Train and validation loss

4.3.2 *Proposed Model 2*

The CNN exhibits a hierarchical architecture designed for effective feature extraction and classification of the input frames. The model begins with a series of convolutional layers, each comprising a 5 by 5 kernel size. The padding has been set to 'same.' An activation function which is rectified linear unit (ReLU) function has been utilized in the proposed hierarchical CNN architecture, which introduces non-linearity to the model.

The convolutional layers, denoted as Conv2D(32), Conv2D(64), Conv2D(128), and Conv2D(256), sequentially extract hierarchical features from the input data. The number in parentheses represents the number of filters (or channels) in each convolutional layer, determining the depth of feature maps produced by that layer. For example, Conv2D(32) has 32 filters. These convolutional layers are interspersed with max-pooling layers, employing a 2x2 pool size and a stride of 2. Max-pooling contributes to spatial down-sampling, reducing the spatial dimensions of the feature maps, and promotes translation invariance by retaining the most significant information in each region. In summary, the Conv2D layers with varying filter sizes played a crucial role in hierarchically extracting features from the input frames, and the max-pooling layers aided in down-sampling, thereby enhancing the model's ability to recognize and classify features with translation invariance.

To mitigate overfitting and enhance the overall generalization, dropout layers with a dropout rate of 0.3 were strategically inserted after each max-pooling operation. This inclusion served to regularize the model during training, preventing reliance on specific nodes and encouraging a better representation of the underlying patterns in the data.

Following the convolutional layers, the model transitioned to a fully convolutional layer (FCN) for high-level feature aggregation and extraction. The Flattened layer is employed to convert the 3D tensor output of the convolutional layers into a 1D vector, facilitating the connection to densely connected layers. The subsequent dense layers with 256 neurons introduced non-linearity through ReLU activation, contributing to the model's capacity to capture intricate relationships within the data. The final layer of the network consisted of a densely connected layer, having a SoftMax activation function with the units set to 29, which is equal to the number of classes in the classification task. This layer outputs probability distributions over the classes, enabling the model to make predictions. The choice of the SoftMax activation function ensured that the predicted probabilities sum up to 1, facilitating clear class assignments. The training and validation accuracies and losses are visually depicted in Figure 6. and Figure 7. respectively.

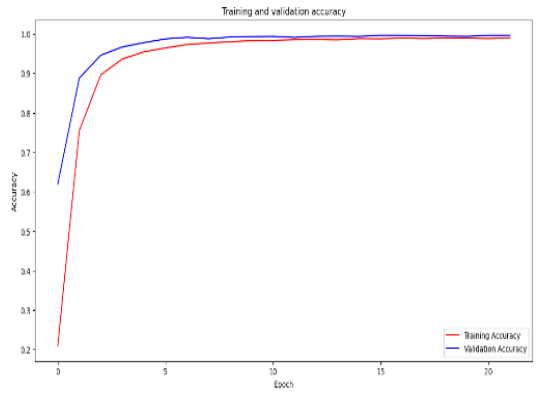

Figure 6 Train and validation accuracy

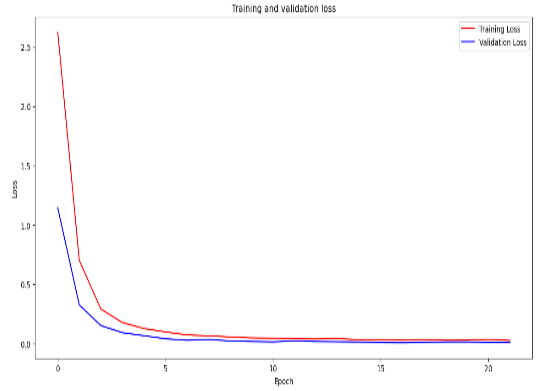

Figure 7. Train and validation loss

4.3.3 *Proposed Model 3*

The proposed CNN architecture for image classification is designed to effectively capture and learn intricate features from the input frames. The model comprises two convolutional blocks, each consisting of a convolutional layer with a 5 by 5 kernel and padding layer that has been set to 'same', followed by max-pooling with a 2 by 2 pool size to down sample the spatial dimensions. Dropout layers with a rate of 0.3 are incorporated after each max-pooling operation to mitigate model overfitting and enhance and improvise the generalization process.

After the convolutional segments of the architecture, the model undergoes a pivotal transition from convolutional layers to fully connected layers. This transition involves flattening the output, essentially reshaping the multi-dimensional feature maps into a one-dimensional vector. This step allows the network to consolidate the spatial information learned by the convolutional layers before progressing to higher-level abstractions.

Following this flattening step, the model introduces a fully connected layer, a critical element that comprises 128 neurons. Each of these neurons is activated by rectified linear units (ReLU). The rectified linear unit activation function allows the network to model complex relationships within the data, enhancing its capacity to capture intricate patterns. To mitigate the risk of overfitting, a dropout layer is implemented after the fully connected layer, with a dropout rate set to 0.5. Dropout is a regularization technique that randomly drops out a fraction of neurons during training, preventing the network from relying too heavily on specific neurons and improving generalization to new data.

The final layer of the network is a densely connected layer. This layer utilized a SoftMax activation function with the number of units set to 29, corresponding to the total number of classes present in the classification task. The SoftMax function normalizes the output into a probability distribution across the different classes, facilitating the assignment of the most probable class for a given input.

In summary, this transition from convolutional to fully connected layers, along with the introduction of activation functions, dropout regularization, and a SoftMax layer, encapsulates the model's capacity to learn intricate features and make informed predictions in a classification scenario with 29 distinct classes. The training and validation accuracies and losses are visually depicted in Figure 8. and Figure 9. respectively.

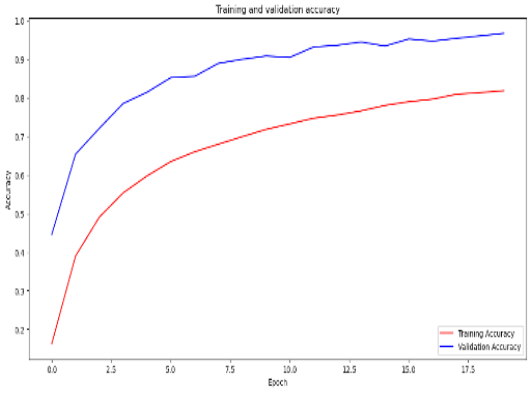

Figure 8 Train and validation accuracy

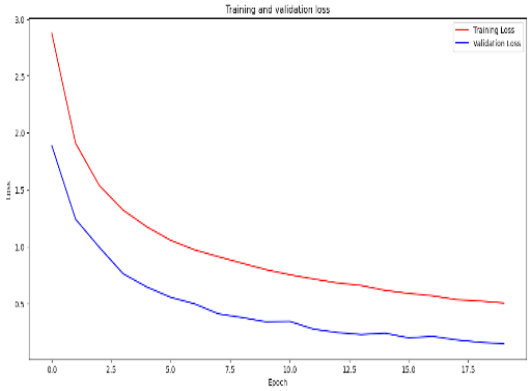

Figure 9. Train and validation loss

4.3.4 VGG 16 Model

The proposed model leverages the VGG16 architecture, a well-established convolutional neural network with pre-trained weights on ImageNet dataset, to address the task of sign language recognition. ImageNet is a widely recognized and extensively used dataset in the field of computer vision and machine learning. Developed by researchers at Stanford University, ImageNet is a large-scale dataset that plays an important role in advancing image classification and object recognition research.

By not incorporating the fully connected layers of the pre-trained VGG16 model and introducing customized layers, the model is tailored to meet specific requirements of the problem of ASL classification. The customization included a flattening layer followed by a densely connected layer featuring 256 neurons, activated by the rectified linear unit activation function. To safeguard against overfitting, a dropout layer was incorporated into the architecture.

During the training process, the categorical cross-entropy loss function, and the Adam optimizer with a learning rate of 0.0001 were employed. The model undergoes training for ten epochs, with each epoch iterating through the training data. The utilization of categorical cross-entropy facilitates the optimization process by measuring the dissimilarity between predicted and true class distributions. The Adam optimizer, known for its adaptive learning rate and efficiency, further refined the model's performance during training. The layers of the pre-trained VGG16 model were frozen during training to retain the knowledge and information that has been captured from the ImageNet dataset, preventing undesired alterations to the base model.

In summary, the proposed model's architecture, extended the VGG16 base model with careful customization, coupled with meticulous choices in loss function, optimizer, and training strategy, helped to create a strategic and informed approach to addressing the complexities inherent in sign language recognition. The training and validation accuracies and losses are visually depicted in Figure 10. and Figure 11. respectively.

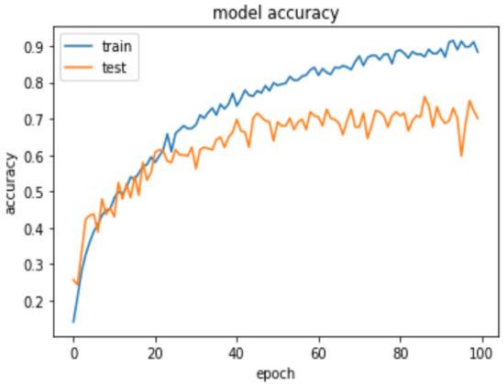

Figure 10 Train and validation accuracy

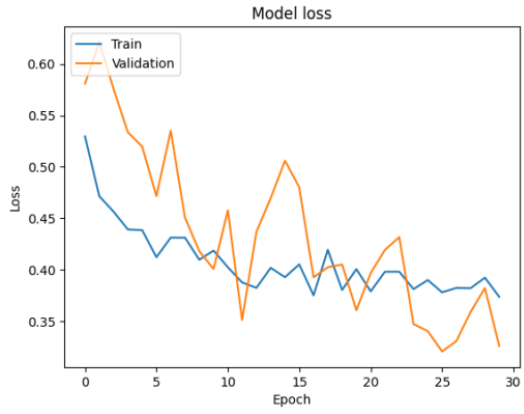

Figure 11. Train and validation loss

Table 1. Comparative analysis of various models used.

Model used	Epochs	Train accuracy	Test accuracy	loss	Val loss
Proposed model 1	50	0.9725	0.9718	0.3394	0.5218
Proposed model 2	**25**	**0.9899**	**0.9963**	**0.0302**	**0.0125**
Proposed model 3	20	0.8745	0.9743	0.561	0.0237
VGG16	30	0.8712	0.9235	0.4025	0.4358

5. RESULT

The analysis report for all the four models that were trained on the dataset as mentioned in section 3, is shown in Table 1 above. The proposed model 2 as described in section 4.3.2 gave the best evaluation metrics. As a result, it was downloaded and incorporated into the web camera where it predicted the characters dynamically as and when the input frame was sent to the CNN model for classification. In the practical implementation, a 2-second timer was introduced following the accurate prediction of each letter and a 4-second timer after the completion of a sentence. These timers served as indicators to control the processing flow of the system. After successfully predicting a letter, the 2-second timer allowed for a brief pause, ensuring stability before processing the next input. The 4-second timer, initiated at the end of a sentence, acted as a signal to halt processing and trigger the output of the corresponding voice. This intentional delay after sentence completion helps in ensuring a coherent and well-paced delivery of the translated sign language into voice. These timers contributed to the overall efficiency and user experience of the proposed system by introducing intentional pauses for smoother transition.

6. GRAMMAR CORRECTION

"MotionScript," is empowered by the T5-Flan Large language model from the transformer library (T5Tokenizer and T5ForConditionalGeneration), marking a pivotal stride in enhancing communication for the deaf and mute community. T5's advanced features, including addition of stop words and grammar correction, elevated MotionScript's precision in converting sign language gestures into clear and grammatically refined text. Impactful integration of T5 within MotionScript, promotes articulate and effective communication across various domains, simplifying language processing for inclusive interactions. The words generated post the classification task was seamlessly transmitted to the LLM. In real-time, this LLM adeptly transformed these words into coherent and meaningful sentences, significantly reducing the overall time required for effective communication. This distinctive feature constitutes the novelty of our approach.

7. VOICE CONVERSION

The gTTS (Google text to speech) API was utilized to convert the corrected sentences into voice in real time by temporarily saving the text and pronouncing once requested by the user.

8. FUTURE SCOPE

While our deep learning model has shown remarkable promise by training on the American Sign Language dataset, there are exciting avenues for future development and expansion. The model developed can be trained on diverse datasets and different sign languages such as the Indian Sign Language, British sign language or any other local version of sign languages. This expansion and diversification can open doors to more inclusive communication, transcending linguistic barriers and benefiting a broader range of communities.

The model that has been developed gives better classification results if the input frames are considerably clear and are captured under proper lighting conditions. To enhance its robustness, we can increase the dataset and make use of advanced techniques such as image augmentation. By introducing controlled noise or variations into the images during training, we can equip the model to perform effectively even in less-than-ideal visual conditions.

In the current iteration of our model, users are granted a 5-second window to input a character via a live webcam, with the system subsequently displaying the corresponding text on the screen. As part of future enhancements, there is an opportunity to optimize the model further, thereby reducing the processing time required for character recognition and text display, ultimately enhancing the efficiency and responsiveness of the system.

The current model, utilizes the T5 – Flan Large model that requires a minimum of three words as input for effective processing and accurate output text generation. The language model can further be enhanced and optimized in future libraries to streamline and reduce this input requirement, thereby enhancing the flexibility and efficiency of the system.

9. CONCLUSION

In this work, a comparative analysis of four distinct custom neural network models were conducted to perform accurate and effective sign language recognition task The goal was to identify the most suitable model that could seamlessly interpret ASL gestures. The rigorous experimentation involved a thorough evaluation of each model's performance, considering factors such as accuracy, efficiency, and adaptability to the unique challenges posed by sign language recognition. Through a systematic comparison, we aimed to discern which model exhibited superior performance, ultimately guiding the decision-making process in selecting the optimal architecture for the task.

Additionally, a novel technique for grammar correction was introduced by integrating it with a fine-tuned Large Language Model. This innovative approach addressed the nuances of sign language translation by ensuring that the generated sentences not only maintained accuracy but also adhered to grammatical norms. The integration of grammar correction with a finely

tuned LLM was a deliberate effort to enhance the coherence and meaningfulness of the translated sign language gestures in real-time.

The comparative analysis and the novel technique of grammar correction contribute to the depth and uniqueness of our work, aiming to not only advance the field of sign language recognition but also to improve the overall communicative experience for the deaf and mute community.

10. ACKNOWLEDGMENTS

The authors extend their sincere gratitude to Ms. Anagha Durugkar, Ms. Sakshi Surve, and Ms. Manisha Dumbre, faculty from the Computer Department at Thadomal Shahani Engineering College, Bandra, for their invaluable guidance and expertise in carrying out this research work. Moreover, a heartfelt appreciation is extended to Rishi Kewalya of Thadomal Shahani Engineering College, Bandra, for his significant contribution to the research.

11. REFERENCES

[1] Hsien-I Lin, Ming-Hsiang Hsu, Wei-Kai Chen, "Human Hand gesture recognition using a convolution neural network", 10.1109/CoASE.2014.6899454, August 2014

[2] Bikash K. Yadav, Dheeraj Jadhav, Hasan Bohra, Rahul Jain, "Sign Language to Text and Speech Conversion", International Journal of Advance Research, Ideas, and Innovations in Technology, www.IJARIIT.com.

[3] Garcia, B., & Viesca, S. A. (2016), "Real-time American sign language recognition with convolutional neural networks, Convolutional Neural Networks for Visual Recognition, 2, 225-232

[4] S. S Kumar, T. Wangyal, V. Saboo and R. Srinath, "Time Series Neural Networks for Real Time Sign Language Translation," 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, 2018, pp. 243-248, doi: 10.1109/ICMLA.2018.00043.

[5] AlKhuraym, Batool Yahya et al. "Arabic Sign Language Recognition using Lightweight CNN-based Architecture." International Journal of Advanced Computer Science and Applications (2022)

[6] X. Han, F. Lu and G. Tian, "Sign Language Recognition Based on Lightweight 3D MobileNet-v2 and Knowledge Distillation," ICETIS 2022; 7th International Conference on Electronic Technology and Information Science, 2022, pp. 1-6.

[7] M. Mahesh, A. Jayaprakash and M. Geetha, "Sign language translator for mobile platforms," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, 2017, pp. 1176-1181, doi: 10.1109/ICACCI.2017.8126001.

[8] V. N. T. Truong, C. Yang and Q. Tran, "A translator for American sign language to text and speech," 2016 IEEE 5th Global Conference on Consumer Electronics, 2016, pp. 1-2, doi: 10.1109/GCCE.2016.7800427.

[9] Yang, Jie and Y. Xu. "Hidden Markov Model for Gesture Recognition." (1994), doi: 10.21236/ada282845.

[10] Lee, C. K. M. et al. "American sign language recognition and training method with recurrent neural network." Expert Syst. Appl. 167 (2021).

[11] Sari, Winda & Rini, dian Palupi & Malik, Reza. (2020), "Text Classification Using Long Short-Term Memory With GloVe Features", Jurnal Ilmiah Teknik Elektro Komputer dan Informatika. 5. 85. 10.26555/jiteki.v5i2.15021

[12] Saleh, Yaser and Ghassan F. Issa. "Arabic Sign Language Recognition through Deep Neural Networks Fine-Tuning." Int. J. Online Biomed. Eng. 16 (2020): 71-83.

[13] M. S. Nair, A. P. Nimitha and S. M. Idicula, "Conversion of Malayalam text to Indian sign language using synthetic animation," 2016 International Conference on Next Generation Intelligent Systems (ICNGIS), Kottayam, 2016, pp. 1-4, doi: 10.1109/ICNGIS.2016.7854002.

[14] V. Lopez-Ludena, R. San-Segundo, R. Martin, D. Sanchez and A. Garcia, "Evaluating a Speech Communication System for Deaf People," in IEEE Latin America Transactions, vol. 9, no. 4, pp. 565-570, July 2011, doi: 10.1109/TLA.2011.5993744.

[15] D. Kelly, J. Mc Donald and C. Markham, "Weakly Supervised Training of a Sign Language Recognition System Using Multiple Instance Learning Density Matrices," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 41, no. 2, pp. 526-541, April 2011, doi: 10.1109/TSMCB.2010.2065802.

[16] R. San Segundo, B. Gallo, J. M. Lucas, R. Barra-Chicote, L. D'Haro and F. Fernandez, "Speech into Sign Language Statistical Translation System for Deaf People," in IEEE Latin America Transactions, vol. 7, no. 3, pp. 400- 404, July 2009, doi: 10.1109/TLA.2009.5336641.

[17] Ss, Shivashankara & S, Dr.Srinath. (2018). American Sign Language Recognition System: An Optimal Approach. International Journal of Image, Graphics and Signal Processing. 10. 10.5815/ijigsp.2018.08.03.

[18] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), 2015, pp. 730-734, doi: 10.1109/ACPR.2015.7486599.

[19] Akash Nagaraj. (2018).*ASL Alphabet* [Data set]. Kaggle. https://doi.org/10.34740/KAGGLE/DSV/29550