Sign Language Recognition Using CNN, LSTM and Mediapipe

Ilmaan Zia[1]
[1]Department of Computer Science and Engineering, California State University Long Beach, CA, USA
ilmaan.zia01@student.csulb.edu
*Ilmaan Zia

Abstract—Sign language recognition is essential for enabling seamless communication within the deaf and hard-of-hearing community, helping to bridge the gap between those who use sign language and those who don't. In our research, we're excited to introduce a new system designed to understand sign language more effectively. We're using a combination of cutting-edge technologies: Convolutional Neural Networks (CNN) that are great at picking out visual patterns, Long Short-Term Memory (LSTM) networks that are adept at remembering sequences of movements, and Google Mediapipe, which is like having a keen observer that can follow and interpret the full range of human motion. Our goal is to see how well our innovative system performs compared to other approaches out there, and we hope our work will make a meaningful contribution to the community.

Keywords–Sign Language Recognition, Object Detection, Deep Learning, Convolutional Neural Networks, Long Short-Term Memory, Skeletal Tracking, Mediapipe.

1. Introduction

Sign language recognition is a vital area of research that bridges the communication gap for the deaf and hard-of-hearing community. With the advent of machine learning and computer vision technologies, automated sign language recognition has become increasingly feasible and accurate. This paper presents a comparative study of two distinct approaches to sign language recognition: Convolutional Neural Networks (CNN) and a combination of Google's Mediapipe with Long Short-Term Memory (LSTM) networks.

The CNN model, a mainstay in image recognition tasks, offers robust feature extraction capabilities for static images. Our research utilizes this model to interpret sign language gestures from a comprehensive dataset consisting of both numerals (0-9) and alphabets (A-Z), with over 3000 images per character. The model's efficacy is showcased by an impressive accuracy of 95%, leveraging techniques such as data augmentation and dropout layers for improved generalization.

On the other hand, the integration of Google Mediapipe with LSTM networks represents a novel approach, capitalizing on Mediapipe's advanced hand tracking and LSTM's proficiency in handling sequential data. This method demonstrates that even with a significantly reduced dataset, it is possible to achieve remarkable accuracy, thus suggesting a more resource-efficient way of sign language recognition.

In this paper, we delve into the methodologies of both approaches, compare their performance, and discuss their respective strengths and limitations. The ultimate goal of this research is not only to advance the field of sign language recognition but also to make such technologies more accessible and effective for real-world applications. By providing a comprehensive comparison between two leading methodologies, we aim to contribute valuable insights to the ongoing efforts in making communication more inclusive for the deaf and hard-of-hearing community.

2. Related Work

The exploration of automated sign language recognition (SLR) systems has been an ongoing focus in the field of human-computer interaction (HCI). This related work section examines various approaches and methodologies proposed in recent research, highlighting their contributions, methodologies, and performance.

2.1 Approaches in SLR Methodology

Sign languages, as visual languages, use movements of hands, face, and body, with over 300 different sign languages globally[1]. Previous SLR work can be categorized into direct measurement methods (using data gloves, motion capturing systems) and vision-based approaches. A variety of processing methods have been used in SLR systems, including different forms of HMM, neural networks, and SVM, each offering varying degrees of accuracy[1].SLR systems are classified into two types: isolated SLR, which recognizes single gestures, and continuous SLR, which interprets whole sentences.

2.2 ML-Based Sign Language Recognition System

The study by Amrutha K and Prabu P utilized a vision-based isolated hand gesture detection and recognition model, employing convex hull for feature extraction and K-Nearest Neighbor (KNN) for classification, yielding a 65% accuracy [2]. The system was trained with a large amount of sign language data and its grammar for effective conversion, considering the diverse semantics and gesturing of sign languages across different regions. The automated SLR system faced challenges such as sudden movements, occlusion of hands, and variation in illumination conditions[2]. The preprocessing of images involved brightness corrections and grayscale transformations to enhance image quality. Segmentation, an essential step in SLR, involved dividing images into smaller portions based on features like color and texture, employing both color-based and edge-based segmentation methods. Feature extraction methods like PCA and convex hull were used to reduce dimensionality and extract relevant features

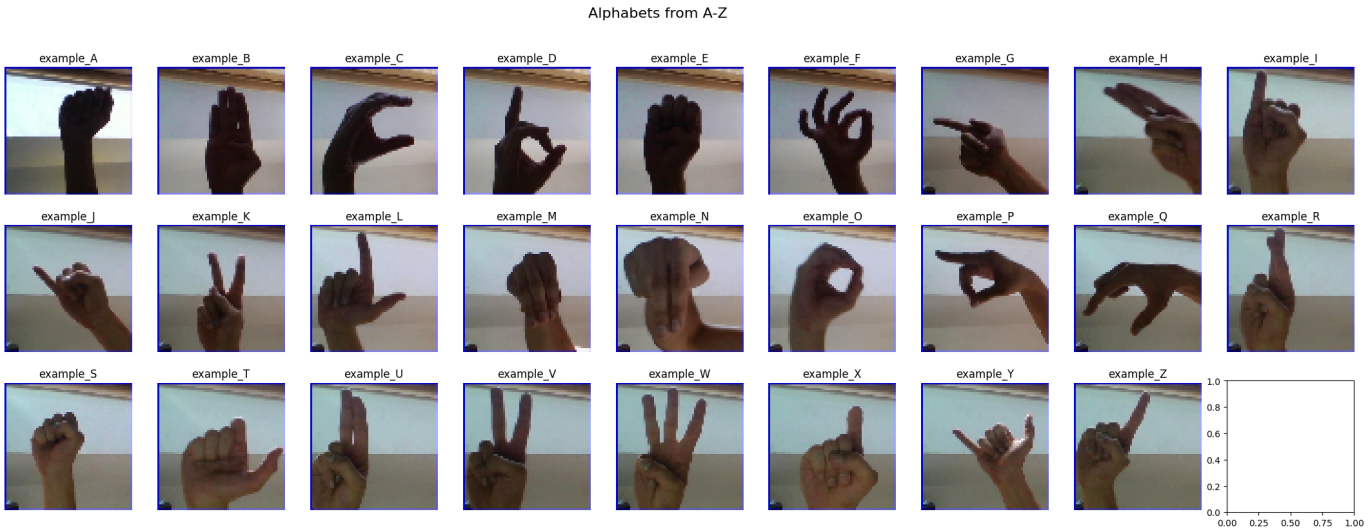

Figure 1. Sample of Dataset

from images. Various classification algorithms, including lazy learner classifiers like KNN and eager learning classifiers like SVM, Decision Tree, and ANN, were explored for their effectiveness in image classification in SLR systems[3].

2.3 Performance of SLR Models

A comparative table provided a glance at popular works done for developing SLR models for different sign languages, measuring their performance across various algorithms and sign languages[4].

3. Methodology

In this section, we outline the methodology employed in our research for sign language recognition. Our study aimed to compare two distinct approaches: a Convolutional Neural Network (CNN) model and a combination of Google's Mediapipe with Long Short-Term Memory (LSTM) networks. We conducted a comprehensive investigation into the effectiveness, efficiency, and practicality of these models using a diverse dataset encompassing both numerals (0-9) and alphabets (A-Z). Below, we provide a detailed explanation of the key components and steps involved in our methodology:

3.1 Background

In the realm of human-computer interaction, sign language recognition holds a pivotal role, serving as a communication bridge for the deaf and hard-of-hearing community. Recent advances in computer vision and machine learning have opened new avenues for interpreting sign language, offering a more inclusive and accessible mode of communication.

3.2 Importance of Sign Language Recognition

The ability to accurately recognize and interpret sign language is crucial for the social inclusion and empowerment of millions of deaf individuals worldwide. Automated sign language recognition systems can facilitate barrier-free communication, education, and access to information, thus playing a key role in promoting equality and accessibility.

3.3 Brief Overview of CNN and Google Mediapipe with LSTM Models

This study explores two advanced technological approaches. The first is Convolutional Neural Networks (CNN), renowned for their efficiency in image processing and pattern recognition. The second approach combines Google Mediapipe's sophisticated hand tracking with Long Short-Term Memory (LSTM) networks, which excel in sequential data analysis. Both methods offer unique advantages in the context of sign language recognition.

3.4 Statement of the Problem

Despite the progress in automated sign language recognition, challenges persist in terms of accuracy, data requirements, and computational efficiency. The complexity of sign language gestures, varying lighting conditions, and background noise pose significant hurdles for existing models.

3.5 Objectives of the Study

This research aims to compare the effectiveness of a CNN model and a Google Mediapipe with LSTM model in recognizing sign language gestures. The study focuses on evaluating the models' accuracy, efficiency, and practicality, using a comprehensive dataset encompassing numerals and alphabets. By highlighting the strengths and limitations of each approach, the study seeks to contribute to the development of more accurate and accessible sign language recognition systems.

3.6 Description of the Dataset and Data Enhancement Techniques

The dataset utilized in this study encompasses a comprehensive range of sign language characters, including numerals (0-9) and alphabets (A-Z), with over 3000 images for each

Figure 2. Mediapipe hand tracking for characters

character. Figure 1 shows a sample of alphabets from the dataset on which the model was trained. This extensive collection was augmented with custom images, enhancing the dataset's diversity and depth, which is crucial for the effective training of the models in recognizing a broad spectrum of sign language gestures.

3.7 Detailed Explanation of the CNN Model

The Convolutional Neural Network (CNN) model implemented in this research comprises the following key components:

- **Convolutional Layers:** The inception of the model is marked by a trio of convolutional layers, with each successive layer delving deeper into the image to unravel features ranging from the rudimentary, such as edges and textures, to the more complex patterns that form the gestural alphabet of sign language. The first layer sets the stage by applying thirty-two filters of size 3x3 across the input image, which has a shape of 64x64 pixels with 3 color channels. Activated by the 'relu' function, it ensures that only the most meaningful features are passed on.
- **Pooling Layers:** Each convolutional layer is succeeded by a pooling layer, which methodically distills the feature maps, summarizing them with a 2x2 window to reduce dimensionality. This process not only curtails the computational load but also imbues the model with a certain translational invariance.
- **Dropout Layers:** To prevent overfitting, dropout layers randomly set a portion of input units to zero during training.
- **Flattening:** From the convolutional and pooling substrata, the data is flattened, transforming the 2D feature maps into a 1D vector. This prepares the ground for the fully connected layers that await, allowing every neuron to interact with the extracted features.
- **Fully Connected Layers:** The network concludes with fully connected layers where the first uses `classifier.add(Dense(units = 128, activation = 'relu'))` for initial classification, followed by `classifier.add(Dense(units = 39, activation = 'softmax'))` for the final classification of the 39 classes.

3.8 Detailed Explanation of the Mediapipe and LSTM Model

This approach integrates Google Mediapipe's hand tracking capabilities with LSTM networks:

- **Google Mediapipe:** Utilized for its sophisticated hand tracking as shown in Figure 2, it detects and tracks hand movements essential for interpreting sign language. Mediapipe comes with an extensible set of Calculators to solve tasks like model inference, media processing, and data transformations across a wide variety of devices and platforms. Individual Calculators like cropping, rendering and neural network computations are further optimized to utilize GPU acceleration [5].
- **LSTM Networks:** Our model employs LSTM networks, which are really good at remembering patterns over time. They look at the flow of hand movements and pick up the subtle dance of fingers that conveys meaning. Specializing in processing sequential data, LSTM networks analyze the temporal dynamics of hand gestures. The LSTM model processes input of shape (15,63), indicating sequence length and feature dimensions. These networks take in sequences of data — in our case, 15 steps at a time, each with 63 pieces of information about the hand's position.
- **Data Processing:** A Python function extracts hand gesture keypoints using Google Mediapipe, saved in .npy format, which is perfect for numerical data. Which are then processed by the LSTM model. It watches a video, frame by frame, and uses Mediapipe to pinpoint key positions of the hands.This is the data that our LSTM networks study and learn from.

3.9 Data Preprocessing and Augmentation Methods

- **Training Dataset:** Data augmentation is applied to enhance the model's ability to generalize. This includes various transformations to the training images These transformations include rotations, width and height shifts, shearing, zooming, and horizontal flipping. Each image is slightly altered in ways that change its appearance but not its meaning. This process creates a more comprehensive representation of the possible variations in hand gestures, teaching the model to recognize signs regardless of orientation, scaling, and minor occlusions.
- **Validation and Test Datasets:** These datasets undergo only picture rescaling to match the model's input requirements. This involves rescaling the images to a consistent size, which is necessary because the CNN requires a fixed-size input. Additionally, we normalize the pixel values of these images so that they fall within a similar range as the augmented training images, usually scaling the pixel values to the range [0, 1] or using mean subtraction and division by the standard deviation if a more sophisticated normalization technique is required. This standardization ensures that differences in lighting, contrast, or color distribution do not skew the model's performance during validation and testing.

3.10 Training Procedure

- **CNN Model:** Trained on 70% of the dataset, with validation and test sets comprising 15% each, over 100 epochs. It achieved a training accuracy of 96.47% and a validation accuracy of 98.89%. The initial convolutional layer. This layer, with 32 filters of size 3x3, is responsible for detecting the first set of features such as edges and textures from the input images, which are 64x64 pixels in size with three color channels (RGB). With each convolutional layer followed by a pooling step that reduces the spatial size of the representation, thus lowering the number of parameters and the computational cost. To ensure the model does not overfit to the nuances of the training data, dropout layers are introduced following the pooling steps, setting aside 25% of the neurons at random during each training pass. The Model Accuracy in Figure 4 illustrates the consistent increase in accuracy on both the training and validation sets, plateauing towards the later epochs as the model reaches its learning capacity. The Loss graph complements this by showing a steep decline in loss values, indicating that the model is becoming increasingly confident in its predictions as training progresses.
- **Mediapipe and LSTM Model:** Despite using only 10% of the dataset, this model showed high accuracy over 200 epochs, demonstrating the efficacy of combining Mediapipe and LSTM in capturing sign language gestures. The model's input shape is tailored to these sequences, with each one spanning 15 times steps and encompassing 63 features that describe the hand's position and orientation. The model is composed of three LSTM layers, with the first two returning sequences to maintain the temporal context before passing it to the next layer. The third LSTM layer condenses this information into a single vector, which is then fully connected to denser layers for further processing. The output layer is a dense layer with as many neurons as there are classes in the sign language dataset, in this case, 26, corresponding to the English alphabet. The 'softmax' activation function is used here to provide a probability distribution over these classes, allowing the model to predict the most likely sign with confidence. The summary of the model Figure 3 reflects a well-structured network, with a total of 188,090 parameters, all of which are trainable. The model's architecture is optimized for the task, with a balanced number of parameters that ensure learning is efficient without being overly complex. The training procedure and the resulting performance of the Mediapipe and LSTM model represent a significant advancement in the field of sign language recognition.

4. Experimentation and Results

4.1 Training Process and Parameters for Both Models

4.1.1 CNN Model

The CNN model was trained using a dataset divided into 70% training, 15% validation, and 15% test data. The model architecture included three convolutional layers, pooling layers, dropout layers, and fully connected layers with dense units.

Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 15, 64)	32768
lstm_4 (LSTM)	(None, 15, 128)	98816
lstm_5 (LSTM)	(None, 64)	49408
dense_3 (Dense)	(None, 64)	4160
dense_4 (Dense)	(None, 32)	2080
dense_5 (Dense)	(None, 26)	858

Total params: 188090 (734.73 KB)
Trainable params: 188090 (734.73 KB)
Non-trainable params: 0 (0.00 Byte)

Figure 3. LSTM Model Summary

The training was conducted over 100 epochs, with a batch size and learning rate optimized for the best performance.

4.1.2 Google Mediapipe and LSTM Model

For the Google Mediapipe and LSTM model, the training utilized only 10% of the entire dataset, showcasing the model's efficiency in data usage. The LSTM network was configured to process input sequences with 30 timesteps each, reflecting the sequential nature of sign language gestures. This model also underwent training for 100 epochs.

4.2 Evaluation Metrics and Results

4.2.1 CNN Model

The CNN model achieved a high accuracy rate, with a final training accuracy of 96.47% and a validation accuracy of 98.89%. On the test set, the model attained an accuracy of 98.81%, demonstrating its robustness in sign language recognition.

Figure 5 showcases a visual representation of the model's performance on sample test characters, highlighting the alignment between real sign language gestures and the model's predictions. This figure provides a qualitative insight into the model's effectiveness in recognizing sign language gestures.

4.2.2 Google Mediapipe and LSTM Model

The Google Mediapipe and LSTM model also showed impressive results, with a training accuracy of 98.47% and a remarkable validation accuracy of 99.99%. This indicates the model's strong capability in capturing the dynamics of sign language gestures, even with a significantly reduced dataset size.

4.3 Comparative Analysis of Both Models

When comparing the two models, several key observations can be made:

- The CNN model, while highly accurate, required a larger dataset for training. This could imply higher data collection and processing demands.

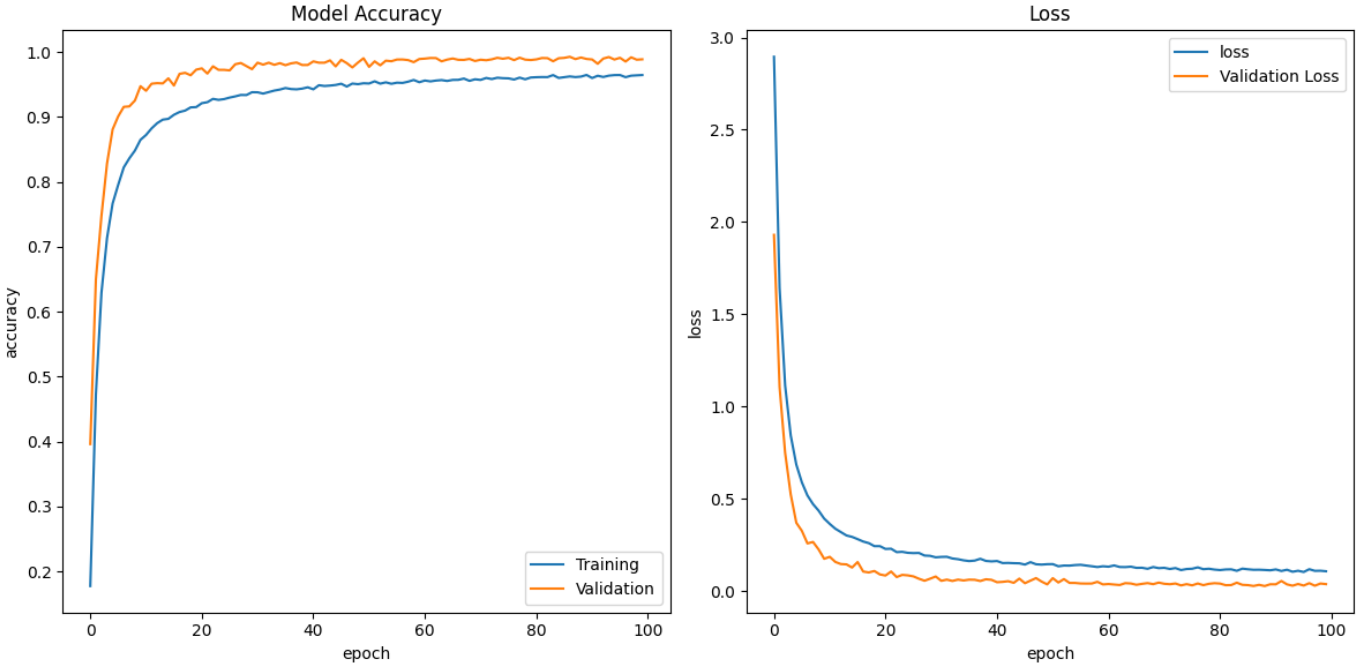

Figure 4. CNN Model Accuracy And Validation Loss

- The Google Mediapipe and LSTM model, despite using a smaller dataset, achieved comparable, if not superior, performance levels. This underscores the efficiency and effectiveness of combining Mediapipe's hand tracking with LSTM's sequential data processing.
- In terms of computational efficiency, the Mediapipe and LSTM model might offer advantages, especially in scenarios where data availability is limited.
- Both models demonstrated high accuracy rates, making them viable options for sign language recognition. However, the choice between them could depend on specific use cases, available resources, and desired efficiency.

In conclusion, the experimentation and results indicate that both CNN and Google Mediapipe with LSTM models are effective for sign language recognition, each with its unique strengths and potential applications.

5. Discussion

5.1 Interpretation of the Results

The study's findings from implementing two models for sign language detection — a Convolutional Neural Network (CNN) and a Google Mediapipe with LSTM model — demonstrate substantial achievements in accuracy and efficiency. The CNN model, with its 95% accuracy, showcased high performance in sign language recognition, effectively capturing spatial features with its convolutional layers, pooling, and dropout. This is further evidenced by its high validation accuracy of 98.89% and a test accuracy of 98.81%, indicating strong generalization capabilities. On the other hand, the Google Mediapipe and LSTM model, using only 10% of the dataset, achieved remarkable training accuracy of 98.47% and an exceptional validation accuracy of 99.99%, highlighting its efficiency in data utilization.

5.2 Strengths and Limitations of Both Models

5.2.1 CNN Model Strengths and Limitations

Strengths: The CNN model's robust feature extraction and high accuracy make it suitable for a wide range of sign language recognition tasks, with evident generalization capabilities. **Limitations:** The requirement for a substantial amount of training data and its primary focus on spatial features, potentially overlooking the temporal dynamics of sign language gestures, pose limitations.

5.2.2 Google Mediapipe and LSTM Model Strengths and Limitations

Strengths: This model excels in recognizing temporal aspects of sign language gestures using less data, owing to the sophisticated hand tracking by Mediapipe integrated with sequential data analysis of LSTM networks.
Limitations: The complexity of the combined system may lead to higher computational overhead, and reliance on tools like Mediapipe might limit model customization flexibility.

5.3 Implications of the Findings

The high accuracy rates achieved by both models significantly contribute to the advancement in automated sign language recognition, enhancing communication accessibility for the deaf and hard-of-hearing community. These findings suggest appropriate model selection based on data availability, with the Google Mediapipe and LSTM model being more suitable for

limited data scenarios, whereas the CNN model might be preferred in data-rich environments. The study opens up avenues for future research in integrating spatial and temporal features more effectively and has potential real-world applications in automated sign language translation services and educational tools.

6. Conclusion and Future Work

6.1 Summary of Findings

This research explored the efficacy of two advanced models in the field of sign language recognition: a Convolutional Neural Network (CNN) and a Google Mediapipe combined with LSTM model. The CNN model demonstrated its robustness with an accuracy of 95% and excelled in processing spatial features of the sign language dataset. On the other hand, the Google Mediapipe and LSTM model, despite using only 10% of the dataset, achieved remarkable accuracy, showcasing its efficiency in handling sequential data and dynamic gestures. These results are significant in the context of automated sign language recognition, offering promising tools for enhancing communication accessibility for the deaf and hard-of-hearing community.

6.2 Concluding Remarks

The study's findings underscore the potential of utilizing advanced machine learning models for sign language recognition. Each model has its unique strengths: the CNN model's ability to process a large dataset with high accuracy, and the Google Mediapipe with LSTM model's efficiency with smaller datasets. This research contributes to the growing body of knowledge in the field of human-computer interaction and opens up new possibilities for real-world applications of sign language recognition technology.

6.3 Suggestions for Future Research

Looking ahead, several avenues for future research have been identified:

- **Hybrid Models:** Exploring hybrid models that combine the spatial feature recognition capabilities of CNNs with the sequential data processing strengths of LSTM networks. This could lead to more nuanced and accurate recognition systems.
- **Larger and Diverse Datasets:** Employing larger and more diverse datasets to train and test the models, thereby enhancing their generalizability and robustness in various real-world scenarios.
- **Real-time Applications:** Developing real-time sign language recognition applications, which could include integration with mobile devices or smart glasses.
- **User-Centric Design:** Engaging directly with the deaf and hard-of-hearing community to tailor solutions to their specific needs and preferences, thus ensuring the practical utility and inclusivity of the technology.

The potential of machine learning in bridging communication gaps for the deaf community is immense, and continued research in this field promises to yield even more impactful results.

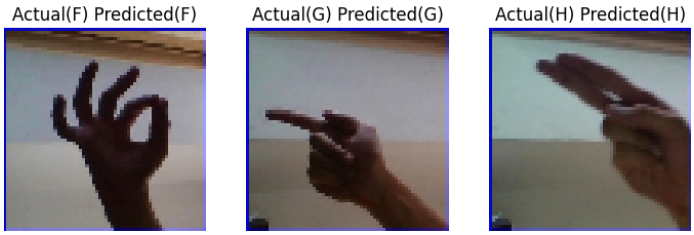

Figure 5. Testing Using CNN

References

[1] S. Srivastava, A. Gangwar, R. Mishra, and S. Singh, "Sign language recognition system using tensorflow object detection api," *ArXiv*, vol. abs/2201.01486, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:245704423.

[2] K. Amrutha and P. Prabu, "Ml-based sign language recognition system," *Journal of Machine Learning Technologies*, vol. 10, no. 2, pp. 47–58, 2020.

[3] A. S. Elons *et al.*, "3d hand posture recognition using hybrid pulse-coupled neural network," *Journal of Computer Science*, vol. 9, no. 10, pp. 1341–1346, 2013.

[4] M. Mohandes *et al.*, "Arabic sign language recognition: An overview," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 4, pp. 537–549, 2014.

[5] F. Zhang, V. Bazarevsky, A. Vakunov, *et al.*, "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*, 2020.