American Sign Language Prediction Using LSTM

Yash Jitendra Modi
Computer Science and Engineering Department
University of Texas at Arlington
Arlington, Texas, USA
yxm6296@mavs.uta.edu

Jay Bijal Shah
Computer Science and Engineering Department
University of Texas at Arlington
Arlington, Texas, USA
jxs0971@mavs.uta.edu

***Abstract*—This research introduces an innovative American Sign Language (ASL) prediction system utilizing Long Short-Term Memory (LSTM) networks, seamlessly integrated into a real-time classification framework within a Flask web application. The model accurately interprets six distinct ASL gestures through live video input, employing the MediaPipe library for holistic hand tracking. The system incorporates a confidence threshold to ensure robust predictions, and its user-friendly interface displays dynamic visual representations of recognized gestures in real-time. A Flask server facilitates continuous video streaming with ASL predictions and establishes a text feed for broadcasting interpreted ASL sentences. The proposed system demonstrates its efficacy in providing immediate and reliable communication support for individuals with hearing impairments, offering a promising solution for enhancing accessibility in diverse contexts.**

***Index Terms*—LSTM, MediaPipe, Gesture Recognition, Real-time, Holistic**

I. Introduction

In contemporary society, effective communication is essential for fostering inclusivity and understanding among diverse populations. For individuals with hearing impairments, American Sign Language (ASL) serves as a primary means of expression, yet existing technological solutions for ASL interpretation often lack real-time accuracy and accessibility. This research addresses the critical need for improved ASL communication tools by introducing a pioneering system that integrates Long Short-Term Memory (LSTM) networks into a Flask web application for real-time ASL prediction. With an increasing emphasis on inclusivity and accessibility, particularly in digital communication, the significance of developing accurate and efficient tools to bridge communication gaps for the hearing-impaired cannot be overstated.

Despite recent advancements in gesture recognition and machine learning, a comprehensive and real-time ASL prediction system that seamlessly integrates into user-friendly applications remains an open challenge. This research identifies and addresses this gap in the literature, focusing on the development of a robust and practical solution that not only predicts ASL gestures accurately but also provides a dynamic and visually intuitive interface for users. By leveraging LSTM networks, known for their ability to capture temporal dependencies, the proposed system aims to overcome the limitations of existing technologies and contribute to the advancement of assistive tools tailored to the unique communication needs of individuals proficient in ASL.

This paper outlines the architecture and implementation of the ASL prediction system, emphasizing its potential impact on communication accessibility. By offering a detailed examination of the model's integration into a Flask web application, real-time gesture recognition capabilities, and user interface design, the research demonstrates a concerted effort to fill the existing gap in accessible ASL communication tools. The subsequent sections delve into the technical aspects, performance evaluation, and user experience, aiming to establish this work as a substantial contribution to the ongoing discourse surrounding assistive technologies for individuals with hearing impairments.

II. Related Work

A. Prior Approaches to Sign Language Recognition:

Early attempts at Sign Language (ASL) recognition predominantly relied on handcrafted features and conventional machine learning algorithms. The paper [1] introduces a continuous Indian Sign Language (ISL) gesture recognition system, leveraging single or both hands for gestures, with background-invariance and efficient frame overlapping. Employing Discrete Wavelet Transform (DWT) for feature extraction and Hidden Markov Model (HMM) for testing, the proposed system demonstrates effectiveness across diverse backgrounds with low time and space complexity. This work contributes to addressing the challenge of recognizing continuous sign language gestures, offering a practical solution for natural human-computer interaction.

B. Integration of Deep Learning:

The landscape of ASL recognition has undergone a transformative shift with the advent of deep learning. In this context, the paper [2] introduces a vision-based application using deep learning for sign language translation, employing CNN (Inception) for spatial features and RNN for temporal features. The dataset is derived from the American Sign Language Dataset. The research outlines challenges like skin tone and clothing variations, suggesting potential improvements with alternative RNN architectures and considering Capsule Networks over Inception in CNN.

C. Contributions from Diverse Datasets:

The availability of comprehensive datasets has significantly propelled the progress of ASL recognition research. Notable

datasets such as ASLLVD, ASL-LEX, and WLASL encompass a wide range of sign variations and linguistic contexts, facilitating the development and evaluation of ASL models. These datasets serve as critical resources for training and testing ASL recognition systems, enabling the exploration of variations in signing styles and linguistic nuances.

D. Real time detection:

This [3] study not only proposes a real-time sign language detector using deep learning for words identification and gestures detection but also emphasizes practicality. The research integrates LSTM with MediaPipe holistic landmarks, achieving around 92% accuracy for continuous signs, and a YOLOv6 model with 96% accuracy for static signs. The provided code significantly contributes to the ASL recognition domain by seamlessly combining deep learning, diverse dataset utilization, and real-time application. The integration of LSTM networks for temporal modeling and real-time MediaPipe hand tracking underscores the system's adaptability and addresses challenges posed by dynamic signing gestures, providing a robust solution for real-time ASL communication.

III. Sign Language Dataset

In our quest to develop a resilient hand gesture recognition model, we initially explored existing datasets; however, the high diversity of classes posed a significant overfitting challenge. To address this issue, we shifted our strategy towards the creation of a bespoke dataset, outlined in this paper. Employing the Mediapipe module, we meticulously captured 30 videos, each containing 30 frames as .npy files, for six distinct hand gestures such as "Hello, goodbye, Thank you!, yes, no and I love you" and stored keypoints of each frames as numpy array files. Notably, to enhance the model's adaptability across varying hand sizes, the dataset was expanded by recording the same gestures performed by three different individuals. The visual representation of our data collection framework, depicted in the accompanying image, underscores the efficacy of the Mediapipe module in extracting essential keypoints.

Fig. 1. Data Collection Frame.

Also, we limited the duration of each label's gesture recording to 30 frames to ensure a clear endpoint for the gestures [4]. Additionally, we divided the dataset into a 95-5 split for training and testing sets. Furthermore, we applied a k-fold cross-validation technique to achieve balanced training.

IV. Keypoint extraction using MediaPipe

A. Utilizing MediaPipe to Extract Features:

In this research, we utilized the MediaPipe library's Holistic module for facial, hand, and pose feature extraction from video data. The Holistic module identifies critical landmarks on the face, hands, and body, forming the basis for subsequent analyses. To ensure the reliability of this process, we implemented confidence thresholds 'min_detection_confidence' and 'min_tracking_confidence'. The former dictates the minimum confidence level for initial landmark detection, filtering out less reliable detections, while the latter controls the confidence required for successful landmark tracking across frames. Both thresholds were set to 0.5, striking a balance between sensitivity and reliability in landmark detection and tracking. This choice was determined through empirical testing, aligning the thresholds with the characteristics of our input data and the study's requirements.

B. Extracting Features from Landmarks:

The captured frame undergoes a color conversion process, transitioning from BGR to RGB format, a crucial step for compatibility with the MediaPipe model. The mediapipe_detection function is then employed, which not only converts the frame to RGB but also processes it utilizing the Holistic model provided by MediaPipe. Subsequently, the frame is converted back to BGR format to ensure consistency within our processing pipeline. The resulting image, along with the landmark detection outcomes, is obtained from this operation. To provide a visual representation of the detected landmarks, the draw_styled_landmarks function is applied, enhancing the interpretability of the landmark detection results on the processed image.

C. Extracting Landmark Key Points:

In our investigation, we utilized MediaPipe to determine the X, Y, and Z coordinates in three-dimensional space for 21 carefully selected key points on the hands. This [**?**] computation produced a total of 126 key points, obtained by multiplying the number of hands, the number of key points per hand, and the three spatial dimensions. Subsequently, we present a detailed breakdown of these key points, providing a thorough comprehension of hand gestures and movements in our study.

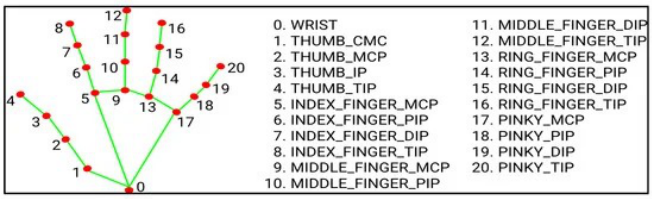

Fig. 2. Data Keypoints.

D. Drawing Landmarks and Connections:

Our video analysis pipeline relies heavily on the draw_styled_landmarks method, which gives a visual representation of the landmarks that have been discovered and their relationships on the processed image. This function improves our understanding of the model's perception and tracking abilities by using the MediaPipe drawing module to generate stylized landmarks for particular body regions.This visualization stage is essential to our research because it allows us to analyze the model's perception of hand movements and interactions qualitatively, which gives us important information into how well our video analysis framework functions overall.

V. Model Architecture

We provide our sequential model with batches of 30 frames, each containing keypoints generated by MediaPipe, resulting in an input size of 30 X 126 for our model. The architecture of the model, outlining the nodes utilized in each layer, is detailed below for your reference.

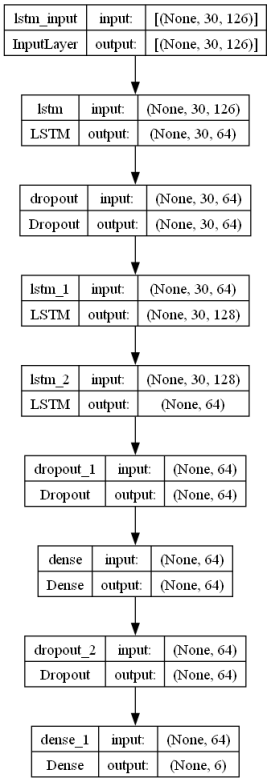

Fig. 3. LSTM Sequential Model.

In constructing our model architecture, we leveraged three crucial types of layers: LSTM, Dropout, and Dense layers. LSTM layers were strategically incorporated to handle extensive sequential data [5], providing the model with the ability to effectively predict intricate hand motion gestures. To prevent over fitting, Dropout layers were introduced, acting as a regularization mechanism. The architecture comprises three LSTM layers, three Dropout layers, and two Dense layers. The final Dense layer outputs six values, each corresponding to one of our distinct class labels.

In our model architecture, all layers, except the final one, employ the ReLU activation function to introduce non-linearity. The last layer utilizes the softmax activation function, aligning with the requirements of multi-class classification [6]. Among the two LSTM layers in our model, both with return sequences set to true, the final LSTM layer differs as it doesn't have this attribute. A dropout of 0.5 is implemented in our model to curb overfitting. This thoughtful combination of activation functions, LSTM configurations, and dropout contributes to the model's effectiveness in capturing complex patterns in hand gesture data.

For optimization, we chose the Adam optimizer, renowned for its effectiveness in training deep neural networks. To address the multi-class classification nature of our task, we employed categorical cross-entropy loss for precise loss calculation during training. This meticulous design ensures the model's proficiency in accurately classifying diverse hand gestures, making it a robust solution for our specific application.

VI. System Flow

In our system architecture, the data collected from MediaPipe undergoes a multi-step process, integrating feature extraction and sequence modeling to enable accurate recognition of complex actions. The initial phase involves the use of the MediaPipe library for face, hand, and pose detection, where confidence thresholds control the sensitivity of the detection and tracking processes. These features are then passed into a Long Short-Term Memory (LSTM) neural network for sequence learning and modeling.

To optimize and assess the model's performance, a Stratified K-Fold cross-validation strategy is utilized. This involves multiple folds, each with a consistent LSTM model instance. Visualizing and saving the model architecture ensures transparency, while TensorBoard logs training metrics. Early stopping mitigates overfitting, stopping training when validation loss improvements cease. The mean loss and accuracy across folds provide insights into the model's generalization, contributing to a comprehensive understanding. The fine-tuned LSTM model is returned for seamless integration into the broader action recognition system.

The script is organized into a Flask web application for easy deployment, leveraging MediaPipe for accurate landmark detection. A pre-trained deep learning model facilitates efficient ASL gesture recognition. The integration of Flask, OpenCV, and MediaPipe makes the system versatile and practical, suitable for various domains in real-world applications.

A. Video Feed and Keypoint Extraction

- The system begins by initializing the video feed using OpenCV's `cv2.VideoCapture(0)`.
- Each frame is processed through the MediaPipe Holistic model for full-body landmark detection.

- Key points, representing hand gestures, are extracted from the landmark results using the `extract_keypoints` function.

B. Loading the Saved Model

- A pre-trained deep learning model (`ASLmodel.h5`) is loaded using TensorFlow's `tf.keras.models.load_model` method.
- This model has been trained to recognize American Sign Language (ASL) gestures based on extracted key points.

C. Sequence Extraction and Prediction

- The extracted key points are appended to a sequence, creating a temporal representation of hand poses.
- The sequence is maintained with a length of 30 frames, forming a sliding window for analysis.
- The loaded model predicts ASL gestures from the sequence, providing likelihood scores for each predefined action.
- Predictions exceeding a specified threshold trigger the formation of a sentence, updating in real-time.

D. Streaming Real-time Video Feed and Text Feed

- The processed frames, including the visualized sentence, are streamed as a continuous video feed using Flask.
- The stream is in the form of multipart data, allowing for dynamic updates and real-time display in a web-based application.
- In addition to the video feed, a text feed is generated continuously, providing a textual representation of the recognized ASL sentence.
- Both the video feed and text feed are designed for real-time updates, enhancing accessibility and usability.

VII. Predictions & Evaluations

The model is trained to predict six distinct ASL gestures, some of which are exampled in the image below. This approach ensures robust performance and generalization despite limited data availability.

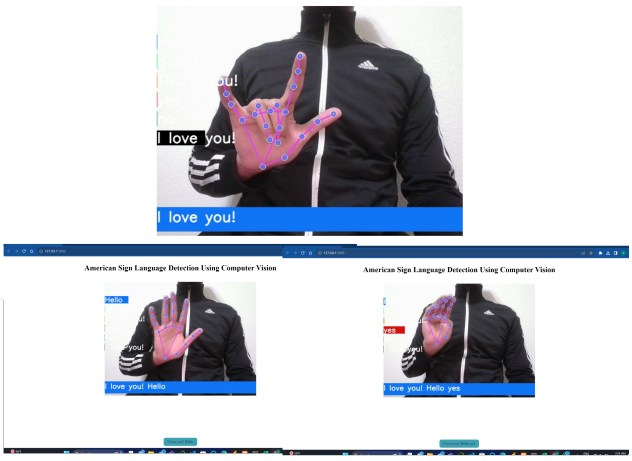

Fig. 4. Output

The dataset underwent a 95-5 training-testing split, resulting in a commendable training accuracy of 94% and testing accuracy of 92.59%. The model demonstrated robust performance in detecting motion gestures. Additionally, in stratified K-fold cross-validation with 5 folds and random state = 42, the average accuracy stood at 89.88%. Notably, we successfully reduced the training loss from 93% to 33%.

To optimize the model output, we implemented techniques including early stopping, threshold, and patience frames for better alignment [7]. A confidence threshold of 0.8 was set to enhance the precision of gesture predictions. Furthermore, we limited our consideration to 30-point sequences to mitigate gesture end detection issues. These strategic measures collectively contribute to the model's overall effectiveness and reliability.

VIII. Challenges

1) Choice of LSTM over Other Models: Selecting the appropriate model architecture is crucial for accurate sign language detection. We experimented with various models, including CNN and RNN, on our image dataset. The decision to employ LSTM was based on achieving the best accuracy in our specific use case of sign language recognition. The temporal dependencies inherent in sign language gestures make LSTM well-suited for capturing sequential patterns and improving overall performance.

2) Video-Based Approach Justification: Opting for a video-based approach [8] over image-based methods was driven by a comprehensive review of existing literature and research. Our investigation revealed that, for real-world applications, a video-based approach provides richer context and more robust sign language interpretation. This decision was informed by insights gained from studying relevant papers and articles in the field.

3) Gesture Ending Detection Challenge: In the video-based approach, a significant challenge arose in accurately detecting the end of a gesture. To address this, we established a constraint where both training and test gestures must be completed within a fixed number of frames (30 frames). This constraint ensures that the model is trained and tested on complete gestures, minimizing potential inaccuracies caused by incomplete or partial gestures.

4) Selection of MediaPipe for Feature Extraction: Extracting meaningful features is a critical step, and we explored various approaches, including the use of OpenCV's Haarcascade. However, we ultimately chose MediaPipe due to its effectiveness in our approach. MediaPipe's capabilities in face, hand, and pose detection proved to be more advantageous, providing accurate and comprehensive keypoint information crucial for sign language recognition.

5) MediaPipe Keypoints Reduction for Improved Accuracy: While incorporating pose, hands, and facial landmarks from MediaPipe initially, we encountered challenges in achieving consistent and accurate model performance. To address this, we focused solely on hand keypoints for feature extraction

in sign language recognition. This reduction in keypoints improved the model's accuracy, as the pose and facial information were found to be less relevant for our specific application, leading to more stable and reliable results.

IX. Conclusions

This research introduces a novel ASL prediction system, seamlessly integrating LSTM networks into a real-time classification framework. The system, implemented as a Flask web application, accurately interprets six ASL gestures through live video input, providing immediate and reliable communication support for individuals with hearing impairments. The integration of LSTM networks and real-time hand tracking demonstrates the system's adaptability and addresses challenges posed by dynamic signing gestures. The proposed solution offers a promising avenue for enhancing accessibility and inclusivity in various domains. The integration of technologies like Flask, OpenCV, and MediaPipe underscores the practicality and versatility of the developed system, making it a valuable contribution to assistive technologies for the hearing-impaired.

X. Future Aspects

Looking ahead, our American Sign Language (ASL) prediction system is poised for significant enhancements. Immediate plans include expanding the system's sign vocabulary to encompass a broader range of ASL gestures, fostering inclusivity in communication. To bolster accuracy, we aim to increase the model's complexity by exploring advanced architectural configurations, delving into deeper neural networks and intricate temporal modeling. Additionally, the integration of facial keypoints and pose analysis will offer a more holistic interpretation of ASL expressions. Continuous dataset enrichment remains a priority, ensuring adaptability to diverse user scenarios. User interface improvements, such as refined visual representations and customization options, will enhance the system's user-friendliness. This multifaceted approach underscores our commitment to advancing the ASL prediction system as a cutting-edge and inclusive tool for individuals with hearing impairments, contributing to the evolution of accessible communication technologies.

XI. Contribution

We both initiated the collaborative effort by delving into the project details for this course project. Eventually, we collectively decided to pursue American Sign Language Detection. The subsequent sentences outline our respective contributions. We both were helpful to each other in all the aspects.

⇒ **Yash**:

- Developed a data collection pipeline for our project model using OpenCV and mediapipe.
- Hosted the project on the web app using flask API and live video feed from OpenCV.
- Implemented K-fold cross validation and early stopping to overcome overfitting issues.
- Drafted sections I, II, IV, VI, and IX of the project report for this project.

⇒ **Jay**:

- Examined and worked with MediaPipe keypoints and ASL signs to identify right match for this project.
- Configured Sequential LSTM model architecture as per the project requirements and data complexities.
- Worked on hyper parameter tuning for achieving higher accuracy.
- Composed section III, V, VII, VIII, X of the project report.

References

[1] K. Tripathi, N. Baranwal, and G. Nandi, "Continuous dynamic indian sign language gesture recognition with invariant backgrounds," 08 2015, pp. 2211–2216.

[2] K. Bantupalli and Y. Xie, "American sign language recognition using deep learning and computer vision," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 4896–4899.

[3] A. KASAPBAŞI, A. E. A. ELBUSHRA, O. AL-HARDANEE, and A. YILMAZ, "Deepaslr: A cnn based human computer interface for american sign language recognition for hearing-impaired individuals," *Computer Methods and Programs in Biomedicine Update*, vol. 2, p. 100048, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666990021000471

[4] A. Vaitkevičius, M. Taroza, T. Blažauskas, R. Damaševičius, R. Maskeliūnas, and M. Woźniak, "Recognition of american sign language gestures in a virtual reality using leap motion," *Applied Sciences*, vol. 9, no. 3, 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/3/445

[5] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using lstms," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 843–852.

[6] A. Farzad, H. Mashayekhi, and H. Hassanpour, "A comparative performance analysis of different activation functions in lstm networks for classification," *Neural Computing and Applications*, vol. 31, 07 2019.

[7] W. Li, W. W. Y. Ng, T. Wang, M. Pelillo, and S. Kwong, "Help: An lstm-based approach to hyperparameter exploration in neural network learning," *Neurocomputing*, vol. 442, pp. 161–172, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231221003337

[8] P. Vitaly, M. Alexander, and P. Andrey, "Recognition of hand gestures on the video stream based on a statistical algorithm with pre-treatment," in *Proceedings of 15th Conference of Open Innovations Association FRUCT*, 2014, pp. 105–111.