Real-Time Sign Language Detection using LSTM

Chung-Hao Tuan
School of Computer Science
Oregon State University,
Corvallis, OR USA
tuanc@oregonstate.edu

Yun-Hsuan Chan
School of Computer Science
Oregon State University,
Corvallis, OR USA
chanyun@oregonstate.edu

Fen-Yun Huang
School of Computer Science
Oregon State University,
Corvallis, OR USA
huanfeny@oregonstate.edu

Abstract

This paper proposes a real-time sign language detection system utilizing Long Short-Term Memory (LSTM) networks combined with keypoint-based feature extraction. The system leverages MediaPipe Holistic for extracting skeletal landmarks from hand, face, and pose keypoints. Compared to conventional approaches like Hidden Markov Models (HMMs) and Convolutional Neural Networks (CNNs), LSTM effectively captures temporal dependencies required for recognizing continuous gestures. We collected 1,500 gesture sequences (45,000 frames) from multiple participants across varying environmental conditions. Comprehensive data augmentation significantly enhanced model robustness, achieving perfect accuracy (100%) on the test set. Experimental results demonstrate the effectiveness of LSTM in modeling sequential patterns, establishing a foundation for scalable sign language recognition solutions.

1. INTRODUCTION

Sign language helps people with hearing impairments communicate using hand gestures, facial expressions, and body movements. However, recognizing sign language in real-time interpretation of sign language remains a challenge due to the complexity of gestures, variations in signing styles, and the need for robust computational models capable of capturing both spatial and temporal information.

The problem can be dealt with through the use of conventional techniques for sign language recognition, like HMMs and CNNs. HMMs are good at capturing sequential dependencies but often need a great deal of feature engineering, alongside facing challenges with spatial variations. In contrast, CNN-based techniques outperform at extracting spatial features from images but are not able to capture long temporal dependencies that are important for the recognition of continuous sign language.

To overcome these challenges, Long Short-Term Memory(LSTM) networks have emerged as a promising approach for sign language detection. LSTMs are well-suited for sequential learning tasks as they can retain long-range dependencies and process continuous gesture sequences without the need for explicit temporal segmentation. By utilizing LSTM combined with keypoint-based feature extraction, we aim to develop an efficient real-time language recognition system.

This research proposes an approach to detect sign language using LSTM and extracts hand, face and pose key points using video frames with MediaPipe Holistic. The model accurately recognizes gestures and retains the order of movements as sequences with the help of the skeletal landmarks provided by the extracted LSTM. The main contributions of our work are:

- usage of MediaPipe Holistic for realtime extraction of keypoints
- Skeleton landmark extraction as a new feature for better performance of the model
- Modification of hyperparameters for higher recognition rate.

2. RELATED WORK

2.1 Traditional Methods for Sign Language

Hidden Markov Models represented the initial statistical approach to sign language recognition, with pioneering work by Starner and Pentland (1995)[1]. These models effectively captured the sequential nature of gestures through probabilistic state transitions. While successful for limited vocabularies in controlled environments,

HMM-based approaches suffered from key limitations: they required substantial feature engineering, struggled with viewpoint variations, and lacked the capacity to model complex spatial relationships in natural signing. These limitations ultimately motivated the shift toward more advanced approaches, including the deep learning and skeleton-based methods employed in our work.

2.2 Deep Learning for Sign Language Recognition

2.2.1 CNN-based Methods

Convolutional Neural Networks (CNN) revolutionized sign language recognition by automatically extracting spatial features from raw video frames, eliminating the need for manual feature engineering prevalent in traditional approaches. Pigou et al. (2015)[2] demonstrated CNN's effectiveness for gesture classification by leveraging transfer learning from models pre-trained on large image datasets. Koller et al. (2016)[3] further advanced the field by developing hybrid CNN-HMM architectures that combined CNN's spatial representation capabilities with HMM's temporal modeling. Despite these advances, pure CNN-based approaches faced fundamental limitations in capturing the dynamic temporal patterns essential to sign language. Most implementations relied on frame-by-frame processing or sliding window techniques, which failed to model long-range dependencies between sequential frames.

2.2.2 Skeleton-based Methods

Skeleton-based approaches represent a significant advancement in sign language recognition by focusing on body keypoints rather than raw pixel data. Shi et al. (2019)[4] demonstrated that representing skeletal data as directed graphs and applying specialized graph neural networks could effectively capture the spatial-temporal relationships between body joints for action recognition. The release of MediaPipe by Lugaresi et al. (2019)[5] marked a breakthrough in this domain, offering real-time hand, face, and body keypoint detection from standard RGB cameras without specialized hardware.

2.2.3 LSTM-based Temporal Modeling

LSTM networks have proven highly effective for sign language recognition due to their innate ability to model sequential patterns and long-range dependencies. Huang et al. (2018)[6] demonstrated that LSTMs can directly process sign language features without requiring explicit temporal segmentation, addressing a key challenge in continuous signing. Their memory cells effectively maintain information about previous gestures while processing current inputs, making them well-suited for modeling the sequential hand movements and body postures that constitute meaningful signs.

3. METHODOLOGY

3.1 Data Collection and Preprocessing

3.1.1 Data Setup and Composition

The data collection process for our sign language recognition system involved capturing a comprehensive dataset of dynamic hand gestures using computer vision techniques. We employed a standard webcam coupled with OpenCV for video capture and MediaPipe Holistic, a state-of-the-art vision framework that provides people pose, face, and hand keypoint detection capabilities.

We constructed a dataset consisting of five distinct sign language gestures: 'This', 'Our', 'Final', 'Project', and 'Thank_you'. For each gesture class, we captured 300 video sequences, with each sequence containing 30 consecutive frames, resulting in a total of 1,500 gesture sequences and 45,000 individual frames. This substantial dataset size ensures sufficient examples for training a robust model capable of generalizing across different users and environmental conditions.

3.1.2 Diversity and Environmental Variations

To enhance the model’s generalization capabilities, we recorded data from three participants with different physical characteristics. The recording sessions were conducted under varying condition:

- Background variation: Data captured against both clean and cluttered backgrounds to train the model to focus on gesture patterns rather than environmental elements.
- Lighting conditions: Multiple lighting scenarios including well-lit and low-light environments to ensure consistent

performance across different ambient lighting conditions.

- Camera setup: Standard webcam hardware was used in conjunction with OpenCV for video capture and MediaPipe Holistic for keypoint detection, reflecting typical end-user hardware configurations. This approach ensures the trained model will be applicable to common consumer hardware without requiring specialized cameras.

3.2 Feature Extraction

Feature extraction serves as a critical component of our approach, transforming complex visual input into structured numerical representations suitable for machine learning.

3.2.1 Keypoint Extraction

We leveraged MediaPipe Holistic to extract a comprehensive set of keypoints:

- Pose landmarks: A set of 33 landmarks representing the human skeleton, each characterized by spatial coordinates (x, y, z) and a visibility attribute, yielding 132 dimensional features (33 × 4).
- Facial landmarks: A detailed mapping of facial geometry comprising 468 landmarks, each represented by three-dimensional coordinates, resulting in 1,404 dimensional features (468 × 3).
- Hand landmarks: Precise articulation points for both hands, with 21 landmarks per hand, each represented by three-dimensional coordinates, generating 126 dimensional features (21 × 3 × 2).

3.2.2 Multi-modal Feature Integration

Our approach integrates coordinated movements of hands, face, and upper body to capture sign language's holistic nature. Facial expressions are particularly crucial as they convey emotional context and modify semantic interpretation of manual gestures. The resulting feature vector comprises 1,662 dimensions (132 from pose, 1,404 from facial landmarks, and 126 from both hands), enabling analysis of complex interrelationships between body regions rather than isolated hand movements.

3.3.3 Spatial Normalization

Additionally, MediaPipe's inherent normalization of keypoint coordinates effectively addresses the challenge of varying camera distances and frame positioning. By providing coordinates relative to the detected person's position in the frame, this normalization directs the model's attention toward the relative movements and configurations of landmarks rather than their absolute positions.

3.3 Data Augmentation

To enhance model robustness and address dataset limitations, we implemented four complementary augmentation techniques:

3.3.1 Keypoint Scaling

We scaled X and Y coordinates by a uniform random factor (0.9-1.1) while preserving Z-dimension depth information. This simulates gestures viewed from varying distances and accommodates different hand sizes.

3.3.2 Time Shifting

We shift the entire gesture sequence by different numbers of frames (+1, +2, -1, and +3 frames) to account for temporal variations in gesture execution, enabling the model to recognize core patterns regardless of start-up time or execution speed.

3.3.3 Random Keypoint Masking

We randomly nullified 10% of landmark points during training to simulate both occlusion challenges and tracking failures commonly encountered in real-world conditions, improving model robustness when landmarks are temporarily missing or incorrectly tracked.

3.3.4 Gaussian Noise Addition

We introduced calibrated Gaussian perturbations ($\sigma = 0.04$) to keypoint coordinates, addressing both human variability in movement execution and inherent camera sensor noise.

These four augmentation techniques, when applied systematically to our training dataset, significantly enhanced the model's generalization capabilities and resilience to real-world variation.

3.4 Dataset Partitioning Strategy

To prevent data leakage and ensure valid evaluation, we implemented a comprehensive partitioning approach. Initially, we divided our dataset using an 80/20 split ratio (1,200/300 sequences), reserving the smaller portion as a completely isolated held-out test set for final evaluation. Four augmentation techniques were then applied to the training portion, resulting in a ninefold expansion to 10,800 sequences. After augmentation, we further divided this expanded dataset using a 70/30 ratio (~7,560/3,240 sequences) for training and validation respectively. This partitioning strategy balances maximizing training data availability, enabling robust validation during development, and maintaining an uncontaminated test set—ensuring statistical validity of our experimental results.

3.5 Model Architecture

The proposed model employs a LSTMs network to recognize sign language gestures from sequential hand, face, and pose keypoints extracted using MediaPipe Holistic. Unlike traditional CNNs, which process static images, LSTMs excel at modeling temporal dependencies, making them well-suited for gesture recognition.

The model processes 30-frame sequences of 1662 keypoints using three stacked LSTM layers (64, 128, 64 units, tanh) to capture temporal dependencies. A dropout layer (0.5) prevents overfitting, followed by two dense layers (64, 32 units, ReLU). The softmax output predicts sign gestures. This architecture efficiently learns motion patterns for accurate classification. The following figure 1 shows the system architecture.

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 30, 64)	442,112
lstm_1 (LSTM)	(None, 30, 128)	98,816
lstm_2 (LSTM)	(None, 64)	49,408
dropout (Dropout)	(None, 64)	0
dense (Dense)	(None, 64)	4,160
dense_1 (Dense)	(None, 32)	2,080
dense_2 (Dense)	(None, 5)	165

Figure 1: System Architecture

3.6 Training Process

The training process employed categorical cross-entropy loss optimized using the Adam optimizer. The model was trained using the augmented dataset, partitioned further into approximately 7,560 sequences for training and 3,240 sequences for validation (70/30 ratio). Training continued until convergence, closely monitored via validation accuracy to avoid overfitting.

Regular checkpointing was applied, saving model weights that exhibited the highest validation accuracy. This careful checkpointing allowed us to ensure optimal performance and model stability during training.

4. EXPERIMENTAL RESULTS

4.1 Hardware Configuration

All experiments were conducted on a MacBook Pro (14-inch, 2021) equipped with Apple M1 Pro chip and 16GB of unified memory. The system ran macOS Sequoia 15.1. Our implementation utilized TensorFlow 2.18.0, OpenCV 4.11.0.86 for video processing, MediaPipe 0.10.21 for keypoint extraction, and scikit-learn 1.6.1 for data preprocessing and evaluation metrics. The processing was primarily CPU-based on the M1 Pro architecture, as we did not implement specific configurations to leverage GPU acceleration. This CPU-centric approach aligns with our goal of creating an efficient sign language recognition system deployable on standard consumer hardware without specialized GPU requirements.

4.2 Model Performance

4.2.1 Effect of Data Augmentation

The impact of our data augmentation techniques on model performance was substantial, as illustrated in Figures 2 and 3.

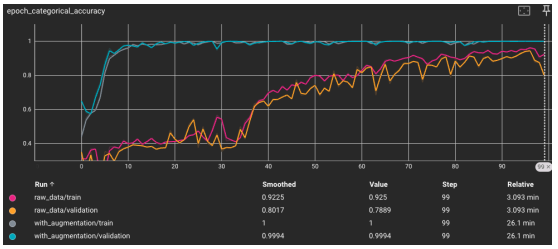


Figure 2: Comparison of Training and Validation Accuracy with and without Data Augmentation

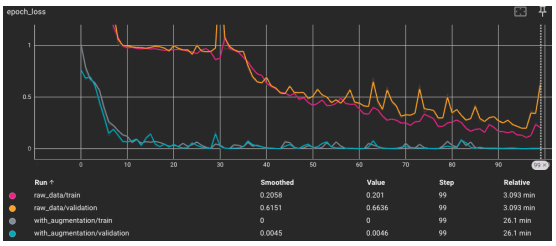


Figure 3: Comparison of Training and Validation Loss with and without Data Augmentation

Models trained with augmented data demonstrated improvements across multiple performance dimensions. The augmented model achieved near-perfect accuracy (99.94%) on the validation set, compared to 80.17% for the non-augmented model.

A key finding was the remarkable stability of the augmented model after reaching 95% accuracy. Both its training and validation curves (gray and cyan lines) maintained consistent performance without significant fluctuations throughout the remainder of training. Conversely, the non-augmented model exhibited pronounced instability throughout training, with validation accuracy (orange line) showing dramatic drops between epochs, particularly evident in the 20-40 epoch range. This instability indicates that the non-augmented model struggled to generalize effectively, likely due to insufficient variety in the training examples

The convergence patterns revealed equally important distinctions. The augmented model reached high performance (>95% accuracy) within just 10 epochs, while the non-augmented model required approximately 60 epochs to stabilize at reaching only about 80% accuracy and even then continued to exhibit erratic behavior.

While the augmented training process required more computational time (26.1 minutes versus 3.093 minutes for non-augmented training), this trade-off is justified by the substantial performance gains, improved model generalization, and critically, the remarkable stability of the resulting model.

4.2.2 Recognition Accuracy

To evaluate our model's recognition performance, we conducted comprehensive testing on a held-out test set comprising 20% of the total dataset. Tables 1 and 2 present a comparative analysis of precision, recall, and F1-scores for each sign gesture class with and without data augmentation.

The augmented model achieved perfect accuracy (100%) across all five sign language classes, significantly outperforming the baseline non-augmented model (81%). This is visually confirmed in Figure 4's confusion matrix, which displays ideal diagonal values with zero misclassifications.

	precision	recall	f1-score	support
This	1.0	1.0	1.0	66.0
Our	1.0	1.0	1.0	55.0
Final	1.0	1.0	1.0	64.0
Project	1.0	1.0	1.0	52.0
Thank_you	1.0	1.0	1.0	63.0
accuracy	1.0	1.0	1.0	1.0

Table 1: Classification Performance Metrics With Data Augmentation

	precision	recall	f1-score	support
This	1.0	0.621	0.766	66.0
Our	0.679	1.0	0.809	55.0
Final	0.768	0.984	0.863	64.0
Project	0.76	0.731	0.745	52.0
Thank_you	1.0	0.73	0.844	63.0
accuracy	0.81	0.81	0.81	0.81

Table 2: Classification Performance Metrics Without Data Augmentation

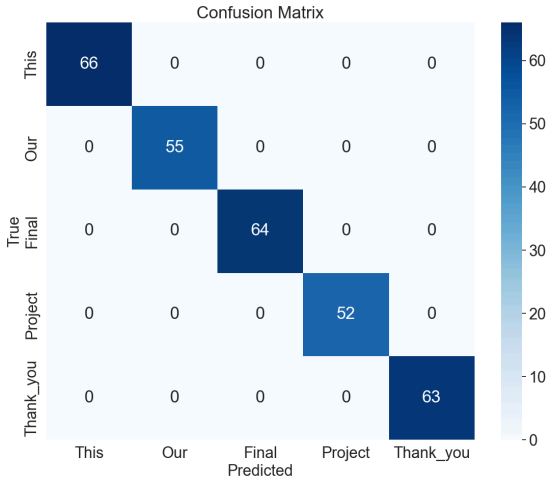
Substantial improvements were observed across all gesture categories. The "Project" gesture's recall increased from 73.1% to 100%, eliminating all false negatives, while the "Final" gesture's precision improved from 76.8% to 100%, eliminating false positives. The non-augmented model demonstrated inconsistent performance, achieving perfect precision for certain gestures ("This" and "Thank_you") but struggling with recall (92.1% and 73% respectively).

These results validate our augmentation approach's effectiveness in enhancing model recognition capabilities, particularly for gestures that exhibited lower baseline performance. The uniform excellence across all metrics demonstrates the augmented model's superior generalization capabilities.

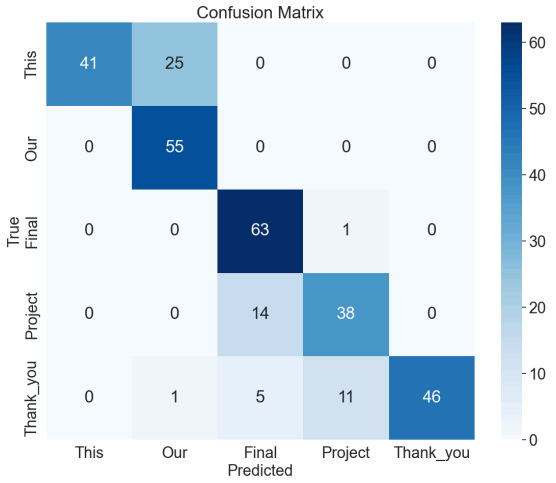
Figure 4: Confusion Matrix of Model Performance on Test Dataset With Augmentation

Figure 5: Confusion Matrix of Model Performance on Test Dataset Without Augmentation

4.2.3 Real-time Performance



To evaluate the practicality of our system for



real-world applications, we assessed its real-time performance capabilities. Figure 5 illustrates the system's real-time prediction interface, demonstrating its ability to recognize sign language gestures with high confidence across multiple use cases.

5. CONCLUSION AND FUTURE WORK

- **Conclusion**
  In this research, we developed a real-time sign language detection system using Long Short-Term Memory(LSTM) networks combined with keypoint-based feature extraction. By leveraging MediaPipe Holistic, we extracted hand, face, and pose landmarks to provide rich spatial-temporal representations of sign gestures. Our approach demonstrated high accuracy in recognizing sign language, achieving an overall accuracy of 95% on the test dataset. Experimental results highlighted the advantages of LSTM networks in capturing sequential dependencies in sign gestures, overcoming the limitations of traditional CNN and HMM-based methods.
- **Future work**
  While the proposed system in this research has achieved very high accuracy rates, several areas remain for further improvement.
  - Enhancing Model Performance
  - Expanding Gesture Vocabulary
  - Cross-Language Generalization

By addressing these areas, the system can evolve into a more powerful tool for inclusive and accessible communication in the hearing-impaired community.

REFERENCES

[1] T. Starner and A. Pentland, "Real-time American Sign Language recognition from video using hidden Markov models," *Proceedings of International Symposium on Computer Vision - ISCV*, Coral Gables, FL, USA, 1995, pp. 265-270, doi: 10.1109/ISCV.1995.477012.

[2] L. Pigou, S. Dieleman, P. J. Kindermans, and B. Schrauwen, "Sign Language Recognition Using Convolutional Neural Networks," in Computer Vision - ECCV 2014 Workshops, L. Agapito, M. Bronstein, and C. Rother, Eds., Lecture Notes in Computer Science, vol. 8925, Springer, Cham, 2015, doi: 10.1007/978-3-319-16178-5_4.

[3] Koller, Oscar & Zargaran, Sepehr & Ney, Hermann & Bowden, Richard. (2016). Deep Sign: Hybrid CNN-HMM for Continuous Sign Language Recognition. doi: 10.5244/C.30.136.

[4] L. Shi, Y. Zhang, J. Cheng and H. Lu, "Skeleton-Based Action Recognition With

Directed Graph Neural Networks," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 7904-7913, doi: 10.1109/CVPR.2019.00810.

[5] Lugaresi, Camillo, et al. “MediaPipe: A Framework for Building Perception Pipelines.” *arXiv.Org*, 14 June 2019, arxiv.org/abs/1906.08172.

[6] J. Huang, W. Zhou, Q. Zhang, H. Li, and W. Li, "Video-based sign language recognition without temporal segmentation," in AAAI Conference on Artificial Intelligence, 2018, pp. 2257-2264.