AOL REPORT

ASL HAND GESTURE RECOGNITION

Deep Learning - LD01
Lecturer:
Wawan Cengoro

Anggota Kelompok:
Supandi - 2702315223
Evan - 2702322853

UNIVERSITAS BINA NUSANTARA JAKARTA
2025/2026

Abstract

An experimental examination of a real-time American Sign Language (ASL) letter recognition system using a Multilayer Perceptron (MLP) classifier with MediaPipe hand landmarks is presented in this research. Using a publicly accessible dataset of ASL alphabet images, each image is processed by MediaPipe Hands to extract 21 three-dimensional hand landmarks. The resulting 63-dimensional feature vectors are then standardized and fed into the MLP to predict one of the ASL alphabet classes. The system is implemented using FastAPI and OpenCV with a threaded architecture that decouples high-frame-rate video streaming from gesture prediction: one thread continuously captures and streams webcam frames to the client, while another thread periodically consumes frames from a queue, performs landmark extraction and MLP inference, and updates a shared prediction state containing the current gesture, confidence score, and top-3 candidates. Temporal smoothing over a short history of predictions is applied to reduce flicker and stabilize the output. Experimental evaluation consists of offline tests on a held-out subset of the dataset to quantify classification performance, as well as online tests in a live webcam setting to assess robustness, latency, and user experience. The results indicate that a lightweight, landmark-based MLP model can provide accurate and responsive ASL alphabet recognition suitable for interactive sign language interfaces on standard consumer hardware.

1. Introduction

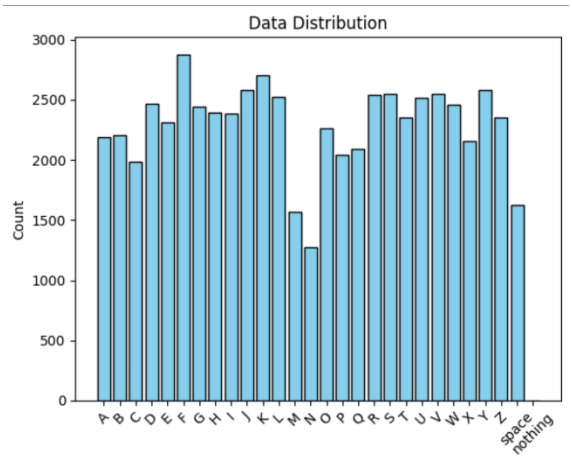
The goal of automatic sign language recognition is to facilitate more inclusive communication between the hearing community and those who are deaf or hard of hearing. Deaf users continue to have major communication obstacles since hearing individuals frequently do not understand sign language. A useful first step toward sign-to-text interfaces that can operate on everyday devices like a laptop with a webcam is real-time recognition of the American Sign Language (ASL) manual alphabet.

In this study, we create and test a real-time ASL alphabet recognition system using a Multilayer Perceptron (MLP) classifier with MediaPipe hand landmarks. The system uses MediaPipe Hands to identify a single hand and extract 21 three-dimensional landmarks instead of learning directly from raw images. These landmarks are then concatenated into a 63-dimensional feature vector, normalized, and sent to the MLP. To provide consistency between training and deployment, the same landmark-based pipeline is employed for inference on live webcam frames once the model has been trained on an ASL alphabet picture dataset.

FastAPI and OpenCV are used in the implementation of the multi-threaded system. While a different prediction thread reads chosen frames from a queue, executes the MediaPipe to MLP pipeline, and updates a shared prediction state that contains the current gesture, its confidence, and the top three candidates, another thread continuously streams webcam frames to the browser to maintain a smooth live feed. Temporal smoothing reduces flicker and increases output stability in real time by using a brief history of predictions.

This research aims to evaluate this lightweight, landmark-based approach's classification performance as well as its useful real-time behavior. We examine how effectively the MLP generalizes to live webcam circumstances, how well it can recognize ASL alphabet movements from the dataset, and how design decisions like frame skipping and prediction smoothing impact perceived stability and responsiveness. The primary contributions are: (1) a real-time ASL alphabet recognition system based on an MLP classifier and MediaPipe landmarks; and (2) an experimental assessment of its online performance and offline accuracy in a realistic webcam-based scenario.

2. Dataset



The dataset comprises multiple gesture classes corresponding to the ASL manual alphabet, letters A–Z plus functional classes such as SPACE, DELETE, and NOTHING. As illustrated in the data distribution plot, each class contains roughly 2,000–2,800 samples, so the dataset is approximately balanced and no single class dominates the data. For model development, the samples are partitioned into three disjoint subsets: 80% of the data for training, 10% for validation, and 10% for final testing, with stratified splitting used to preserve the class distribution across all subsets.

3.Modeling

3.2 Hand Landmark Extraction with MediaPipe

The key idea of this approach is to replace raw image pixels with a compact description of the hand pose by using MediaPipe Hands in static image mode. For each image in the dataset, the image is first passed to the MediaPipe pipeline, which detects the presence of a hand and estimates its pose. If a hand is detected, MediaPipe returns 21 landmarks, each represented by normalized coordinates $(\boldsymbol{x_i}, \boldsymbol{y_i}, \boldsymbol{z_i})$ relative to the image frame. The coordinates of these 21 landmarks are then concatenated into a single feature vector

$$\mathbf{x} = [\boldsymbol{x_1}, \boldsymbol{y_1}, \boldsymbol{z_1}, \dots, \boldsymbol{x_{21}}, \boldsymbol{y_{21}}, \boldsymbol{z_{21}}] \in \mathbb{R}^{\boldsymbol{63}}$$

which serves as the numerical representation of the corresponding image. If no hand is detected in an image, that sample can either be discarded or assigned to a special "no gesture" class, depending on the chosen labeling scheme. After all images have been processed in this manner, the dataset is effectively converted into a collection of labeled examples $(\mathbf{x_i}, \boldsymbol{y_i})$, where $\mathbf{x_i}$is a 63-dimensional feature vector and $\boldsymbol{y_i}$is the associated gesture label.

3.3 Feature Normalization

All feature vectors are stacked into a matrix

$$\mathbf{X} \in \mathbb{R}^{N \times 63},$$

Where N is the number of samples. A feature scaler (for example, standardization) is fitted on the training set:

$$\tilde{\mathrm{x}} = \frac{\mathrm{x} - \mu}{\sigma},$$

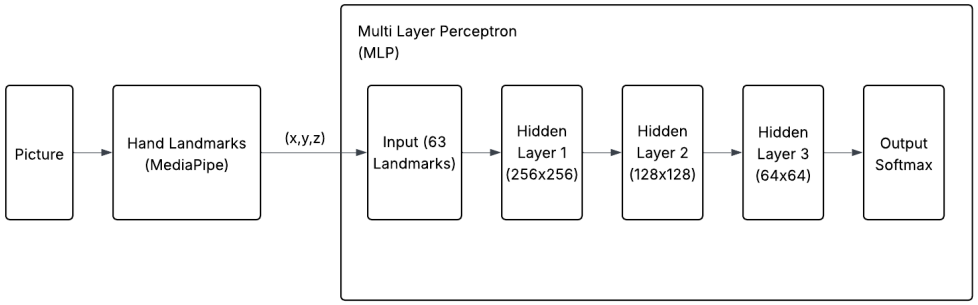
Where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the mean and standard deviation computed per feature dimension over the training data. The same scaling transformation is applied to all validation samples and must also be applied to any new data in deployment. In implementation, this scaler is stored as an object and later

saved together with the trained model.

3.4 Label Encoding

The original gesture labels are strings ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'nothing', 'space']. These labels are converted into integer indices using a label encoder. These integer labels are used as targets during supervised training of the MLP. The label encoder is also stored so that predicted integers can later be mapped back to their corresponding gesture names.

3.5 Model Architecture



The classifier is implemented as a Multilayer Perceptron, which consists of an input layer, one or more hidden layers, and an output layer. The input layer has 63 units corresponding to the 63 normalized landmark features obtained from the MediaPipe Hands module, where each feature vector is formed by concatenating the (x, y, z)coordinates of the 21 detected hand landmarks. The hidden layers are fully connected and use non-linear activation functions (such as Rectified Linear Units, ReLU) to model the complex, non-linear relationships between these landmark-based hand pose representations and the corresponding gesture classes.

3.6 Training Procedure

For a maximum of 100 epochs, the model is trained using mini-batch gradient-based optimization. Each update is calculated on batches of 32 samples taken from the training subset, and performance is tracked on a different validation subset. The network learns to transfer the normalized landmark data to the associated gesture classes by using a contemporary adaptive optimizer in conjunction with a typical multi-class classification loss function. An early stopping method is used to prevent overfitting and avoid needless training by keeping an eye on the validation loss and stopping training if no improvement is seen for 15 consecutive epochs, while retaining the set of weights that achieved the best validation performance. In addition, a learning-rate scheduling mechanism is employed that automatically reduces the learning rate by a factor of 0.5 whenever the validation loss stagnates for 10 epochs, with a minimum allowable learning rate of 1×10^{-7}. This combination of mini-batch training, early stopping, and adaptive learning-rate reduction helps the model to converge efficiently and improves its generalization to unseen data.

4. Evaluation

The proposed ASL alphabet recognition system was evaluated through both offline experiments on a held-out test set and online tests using a live webcam stream. Together, these evaluations assess the classifier's accuracy, generalization ability, and real-time performance under practical deployment conditions.

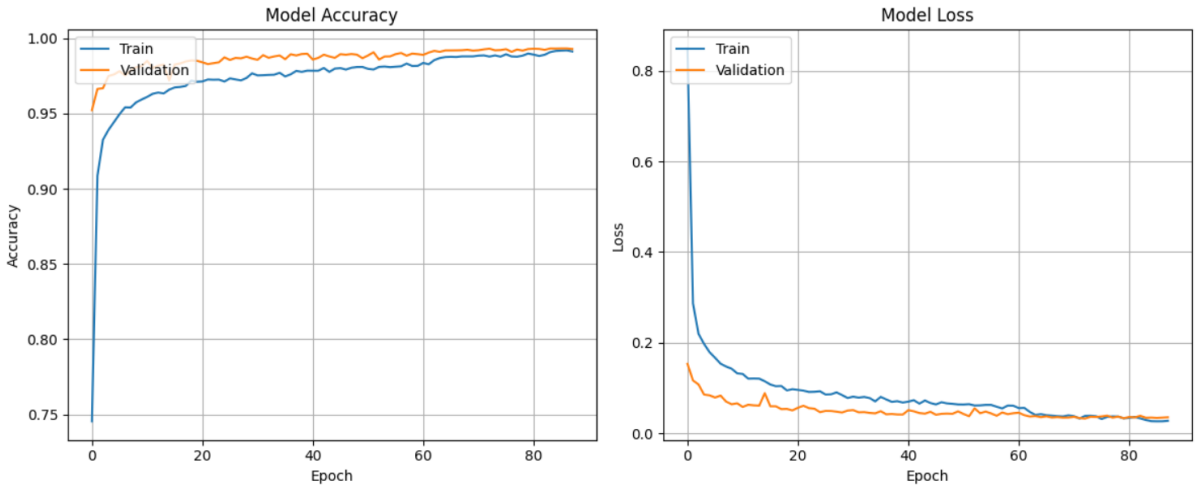
4.1. Classification Performance

Training and validation learning curves show rapid convergence within the first 20 epochs, with both accuracy and loss gradually stabilizing afterward. Early stopping halted training at epoch 88, where the model achieved a final test accuracy of 99.23% and a test loss of 0.033. This indicates

that the landmark-based MLP architecture is capable of learning highly discriminative representations for the ASL alphabet under controlled dataset conditions.

The confusion matrix demonstrates that almost all classes are classified with high precision, with diagonal counts consistently above 400 across all gesture categories. Misclassifications are sparse and primarily occur between visually similar gestures, particularly those involving subtle finger placements (e.g., M vs N, S vs T, U vs V). These errors are expected because MediaPipe's 21-point hand pose model compresses fine-grained finger distinctions into coarse spatial relationships, which limits separability for tightly clustered gestures. Nevertheless, error rates remain extremely low, confirming that landmark normalization, MLP modeling, and stratified dataset splitting effectively support strong offline generalization.

Preprocessing techniques such as feature scaling, early stopping, and learning-rate scheduling contributed to stable optimization but did not materially change the final accuracy, suggesting that the dataset is clean and well-posed for this modeling approach.

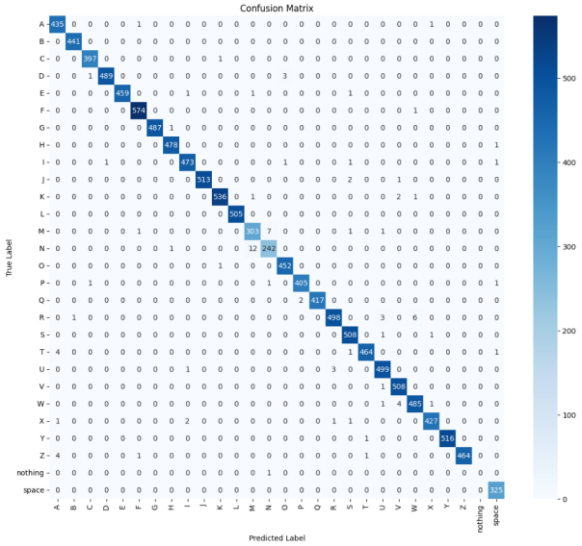


4.2. Real-Time Performance

When deployed in a live FastAPI + OpenCV pipeline, performance differs substantially from offline results due to environmental variability and system-level constraints. The raw video stream maintains high frame rates, but prediction throughput drops when MediaPipe inference and MLP classification run on the same thread. Introducing a dedicated prediction thread reduces UI lag, though per-frame inference remains the primary bottleneck. Attempting to perform prediction at 30 FPS noticeably degrades responsiveness, confirming that MediaPipe's landmark extraction dominates CPU usage on standard hardware.

Hand detection is generally stable across distances and backgrounds, but real-time accuracy declines in two key scenarios: (1) inconsistent lighting, which distorts landmark placement, and (2) right-hand gestures, since the dataset contains only left-hand samples. The left–right mirroring mismatch produces systematic prediction errors even for otherwise trivial gestures. Despite this, the system typically stabilizes to a confident prediction within approximately one second after the user forms a gesture.

Temporal smoothing has only modest effect, as landmark variability in poor lighting conditions creates unstable prediction sequences that smoothing alone cannot fully correct. Even so, smoothing helps suppress short-lived spikes and improves perceived stability during static gestures.



4.3. Generalization Assessment

While offline accuracy exceeds 99%, real-time accuracy is visibly lower, largely due to factors not represented in the training data: hand orientation, lighting variations, camera angle differences, and inconsistencies in MediaPipe's landmark extraction under non-studio conditions. These gaps indicate that the system generalizes well only when runtime conditions closely mimic dataset conditions. Despite this limitation, the model remains responsive and usable for clear, well-lit gestures performed with the same orientation as the training data.

4.4. Generalization Assessment

Evaluation revealed several constraints inherent to a lightweight landmark-based MLP architecture:

- **Left-hand dataset bias** reduces robustness for right-hand users.
- **Landmark instability** under variable lighting degrades prediction confidence.
- **MLP capacity limits** its ability to separate fine-grained gestures with similar poses.
- **MediaPipe inference cost** becomes the dominant bottleneck in real-time systems.
- **Environmental sensitivity** (camera angle, shadows) introduces errors absent in offline testing.

The evaluation demonstrates that the system performs exceptionally well in controlled offline settings and achieves responsive, real-time inference with a simple model on consumer hardware. However, real-world accuracy depends heavily on environmental conditions and the consistency of landmark extraction, suggesting clear opportunities for improvement via data augmentation, mirrored-hand training, or hybrid landmark and image models.

5. Deployment

The system is deployed as a lightweight web application using **FastAPI** as the backend and **OpenCV** for camera handling. A multithreaded design is used to prevent UI blocking:

- **Thread 1** streams webcam frames at high FPS to the client via a FastAPI video endpoint.
- **Thread 2** performs the full inference pipeline — MediaPipe landmark extraction, feature normalization, MLP prediction, and top-3 probability computation.

A shared state object stores the most recent prediction, updated asynchronously and accessed by the client without interrupting inference. Temporal smoothing operates on the server side to stabilize the output before transmission.

The deployment requires only CPU and runs on consumer laptops without GPU acceleration. All preprocessing artifacts — scaler, label encoder, and trained MLP weights — are loaded at startup. MediaPipe is executed in **real-time mode**, allowing frame skipping when the prediction thread is

overloaded to maintain system responsiveness.

6. Reflection

The system successfully demonstrates that **landmark-based models can achieve high accuracy with minimal compute cost**, but several practical limitations emerged during real-time testing. Offline accuracy above 99% does not reflect real-world robustness: prediction quality degrades under poor lighting, hand rotation, and right-hand usage due to dataset bias. MediaPipe itself becomes the computational bottleneck, constraining prediction throughput and limiting scalability for higher FPS or multi-user scenarios.

The MLP classifier performs well for static gestures but lacks capacity for highly similar finger configurations and cannot handle dynamic sequences. The reliance on MediaPipe's 21 landmarks restricts the model's ability to capture fine-grained variations needed for ambiguous letters.

Overall, the system is viable for controlled environments but not yet reliable for unconstrained, real-world use. Improvements should focus on dataset expansion (mirrored right-hand data, lighting and angle augmentation), more efficient runtime inference, and potentially incorporating hybrid image–landmark models to increase robustness without sacrificing real-time performance.

7. References

[1] M. L. Amit, A. C. Fajardo, and R. P. Medina, "Recognition of Real-Time Hand Gestures using Mediapipe Holistic Model and LSTM with MLP Architecture," *2022 IEEE 10th Conference on Systems, Process and Control, ICSPC 2022 - Proceedings*, pp. 292–295, 2022, doi: 10.1109/ICSPC55597.2022.10001800.

[2] S. Barman and S. Majumdar, "Person-Independent Hand Gesture Recognition Using MediaPipe and Multi-layer Perceptron," *Smart Innovation, Systems and Technologies*, vol. 433, pp. 3–16, 2025, doi: 10.1007/978-981-96-1348-9_1.

[3] Y. Zhang and G. Notni, "3D geometric features based real-time American sign language recognition using PointNet and MLP with MediaPipe hand skeleton detection," *Measurement: Sensors*, vol. 38, p. 101697, May 2025, doi: 10.1016/J.MEASEN.2024.101697.

[4] "kinivi/hand-gesture-recognition-mediapipe: This is a sample program that recognizes hand signs and finger gestures with a simple MLP using the detected key points. Handpose is estimated using MediaPipe." Accessed: Dec. 12, 2025. [Online]. Available: https://github.com/kinivi/hand-gesture-recognition-mediapipe?tab=readme-ov-file

[5] B. Zhang and R. Yun, "Robust gesture recognition based on distance distribution feature and skin-color segmentation," *ICALIP 2010 - 2010 International Conference on Audio, Language and Image Processing, Proceedings*, pp. 886–891, 2010, doi: 10.1109/ICALIP.2010.5685201.

[6] M. E. Benalcázar, A. G. Jaramillo, J. A. Zea, A. Paéz, and V. H. Andaluz, "Hand gesture recognition using machine learning and the myo armband," *25th European Signal Processing Conference, EUSIPCO 2017*, vol. 2017-January, pp. 1040–1044, Oct. 2017, doi: 10.23919/EUSIPCO.2017.8081366.

[7] M. Shivani and R. Nawale, "A FRAMEWORK DESIGN FOR SIGN LANGUAGE GESTURE RECOGNITION WITH EFFICIENT HAND GESTURE REPRESENTATION," *www.irjmets.com @International Research Journal of Modernization in Engineering*, vol. 7066, doi: 10.56726/IRJMETS37682.

[8] D. S. JAGLI, N. Kumari, and L. Nakirekanti, "Hand Gestures Recognition System," Jun. 2024, doi: 10.20944/PREPRINTS202406.1837.V1.

[9] C. Lugaresi *et al.*, "MediaPipe: A Framework for Building Perception Pipelines," Jun. 2019, Accessed: Dec. 12, 2025. [Online]. Available: https://arxiv.org/pdf/1906.08172

[10] K. V. Sharma A., "Action Detection for Sign Language Gestures," Department of Computer Science & Engineering and Information Technology Jaypee University of Information Technology, Waknaghat, Solan - 173234 (India) . Accessed: Dec. 12, 2025. [Online]. Available: http://www.ir.juit.ac.in:8080/jspui/bitstream/123456789/11405/1/Action%20Detection%20for%20Sign%20Language%20Gestures.pdf