REAL-TIME SIGN LANGUAGE RECOGNITION SYSTEM

Chappidi Harshitha

Internship Title: AI/ML Internship (1-Month)

Project Title: Real-Time Sign Language Recognition System

Intern Name: Chappidi Harshitha

Institute: 5th Year Integrated MSc Data Science Student, Amrita Vishwa Vidyapeetham, Coimbatore

Internship Organization: Elevate Labs

Table of Contents

1. Introduction
2. Objective of the Study
3. Problem Statement
4. Significance of the Project
5. Overview of the Project
6. Datasets
7. Data Preprocessing
8. Models & Methodologies
9. Results & Output Screens
10. Conclusion
11. References

Introduction

In today's digital world, communication accessibility is vital. For the deaf and hard-of-hearing, sign language is a primary mode of interaction. However, most of the global population is not trained in sign language, creating barriers to inclusive communication.
To address this, the **Real-Time Sign Language Recognition System** aims to harness the power of artificial intelligence and computer vision to recognize hand gestures representing American Sign Language (ASL) alphabets and translate them into text in real time. This project serves as a practical example of how AI can empower assistive technologies for broader societal impact.

Objective of the Study

The objectives of this project are:

- To develop a deep learning model using CNNs to classify ASL alphabet gestures.
- To implement real-time detection using webcam input and convert hand signs to text.
- To integrate MediaPipe and TensorFlow for accurate gesture detection and classification.
- To demonstrate AI's potential in enabling inclusive communication.

Problem Statement

Despite advancements in technology, communication between deaf individuals and the general population remains a challenge due to the lack of widespread sign language knowledge. A real-time sign language recognition system can act as a bridge, reducing this gap. The challenge lies in accurately recognizing static hand gestures of ASL letters (excluding dynamic ones like J and Z) from live video and converting them into readable text with minimal delay and high accuracy.

Significance of the Project

This project plays a crucial role in advancing inclusive technologies:

- **Accessibility**: Makes technology more accessible to hearing-impaired individuals.
- **Cost-Effectiveness**: Uses standard webcam and laptop setups—no special hardware required.
- **Innovation**: Applies state-of-the-art deep learning and hand detection (MediaPipe + CNN).
- **Social Impact**: Encourages society to adopt inclusive digital tools.

Overview of the Project

The system is structured into three core phases:

1. Model Training

- A Convolutional Neural Network (CNN) was designed and trained using the Sign Language MNIST dataset.
- The dataset contains **27,455 training** and **7,172 testing** grayscale images of hand gestures (28x28 pixels) representing 24 ASL alphabets (excluding J and Z).
- Key preprocessing steps included **normalization** and **one-hot encoding** of class labels.

2. Model Evaluation

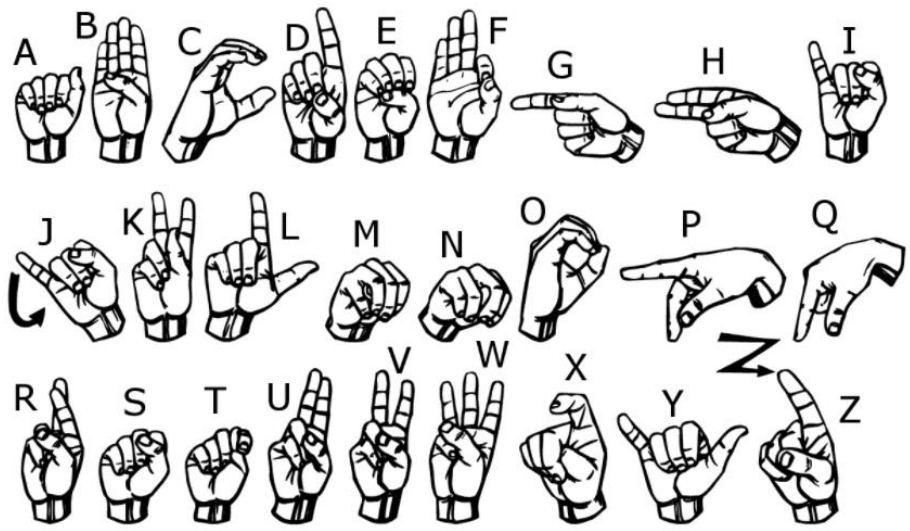
- The model's performance was evaluated based on **accuracy**, **loss trends**, and **classification reports**.
- Output visualization includes:
    - training_history.png – Shows model accuracy and loss across epochs.

3. Real-Time Recognition

- Real-time detection is implemented using a **webcam** and **MediaPipe** for hand tracking.
- The hand region is processed, and the trained CNN model classifies the gesture.
- Recognized letters are formed into words, and the live output is displayed on screen.
- Users can interact using keyboard inputs to manage output text.

Datasets

The dataset used for this project is the **Sign Language MNIST**, sourced from Kaggle. It consists of grayscale images representing 24 static American Sign Language (ASL) alphabet gestures, excluding the dynamic gestures “J” and “Z”. The data is organized in CSV format, where each row contains a label and 784 pixel values representing a 28x28 image. Specifically, the dataset includes **27,455 training images** and **7,172 test images**, making it suitable for building and evaluating deep learning models.



The images were originally captured in color and then pre-processed by cropping, converting to grayscale, resizing to 28x28 resolution, and applying various augmentation techniques like brightness and contrast adjustments, pixelation, and rotation. This processing ensures the dataset reflects realistic conditions, providing a challenging yet practical benchmark for real-world sign language recognition tasks.

Data Preprocessing

Although the Sign Language MNIST dataset is already formatted for image-based classification, additional preprocessing was necessary to make it suitable for training a Convolutional Neural Network (CNN). First, all pixel values were **normalized** to fall within the range [0, 1] to improve model convergence. The labels, initially represented as integers from 0 to 25, were **mapped to corresponding English alphabets**, omitting the dynamic gestures “J” and “Z” due to their motion-based nature. To enable effective multi-class classification, **one-hot encoding** was applied to the labels.

Furthermore, each image was **reshaped** from a flat 1D array (784 pixels) to a 2D format (28x28) with a single channel, matching the input requirements of the CNN. During real-time recognition, additional noise and variability in hand positions were handled using **MediaPipe**, which dynamically detects and isolates the hand region to maintain consistent input quality.

Models & Methodologies

This project utilizes a conservative and realistic approach to building a **Convolutional Neural Network (CNN)** model for static sign language recognition. The methodology focuses on achieving generalization while avoiding overfitting, and includes both model training and real-time implementation.

CNN Architecture and Training Strategy

The model was built using a compact **CNN architecture**, consisting of stacked **Conv2D** layers with **ReLU** activation followed by **MaxPooling**, **Dropout**, and **Flatten** layers. It ends with a **Dense softmax layer** for multi-class classification over 24 ASL alphabets. The model uses **categorical crossentropy** as the loss function and is optimized using **Adam**.

To ensure realistic and generalizable performance, a number of aggressive anti-overfitting techniques were applied:

- **Cross-validation** (5-fold) was used to assess the model's performance on multiple data splits, resulting in a **mean accuracy of 76.6% ± 1.04%** across folds. Each fold loaded fresh data, trained a separate model, and restored the best weights using early stopping.

- A **large validation set (30%)** was split from the original training data to improve the model's ability to generalize, leaving ~19,218 training and ~8,237 validation samples.
- Regularization techniques included **high dropout rates**, **noise injection**, and **minimal CNN depth** to reduce complexity and prevent overfitting.
- **Early stopping** monitored validation accuracy, ensuring training halted once the model's performance plateaued, thus preventing unnecessary fitting.

Final model performance was consistent with the cross-validation results:

- **Validation accuracy:** 75.7%
- **Test accuracy:** 71.8%
- **Overfitting gap (val-test):** 3.96% – within safe limits

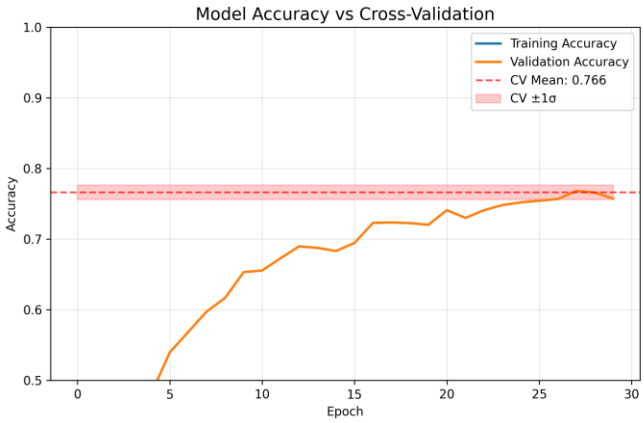
The trained model was saved as conservative_sign_model.keras and visualized using conservative_training_history.png, which plots accuracy and loss over training epochs.

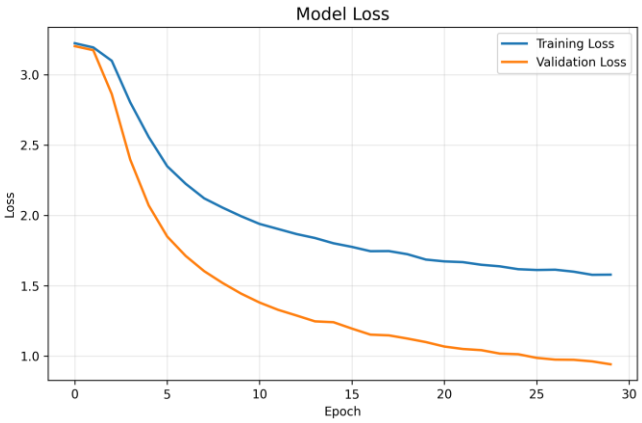
Evaluation Metrics

The model's performance was analyzed using several evaluation metrics:

- **Accuracy:** Overall classification accuracy across test samples.
- **Classification Report:** Precision, recall, and F1-score for each of the 24 gesture classes.
- **Per-Class Accuracy:** Indicates how well individual letters are recognized.

- **Sample Predictions:** Visual comparison of actual vs predicted gestures





Real-Time Recognition Pipeline

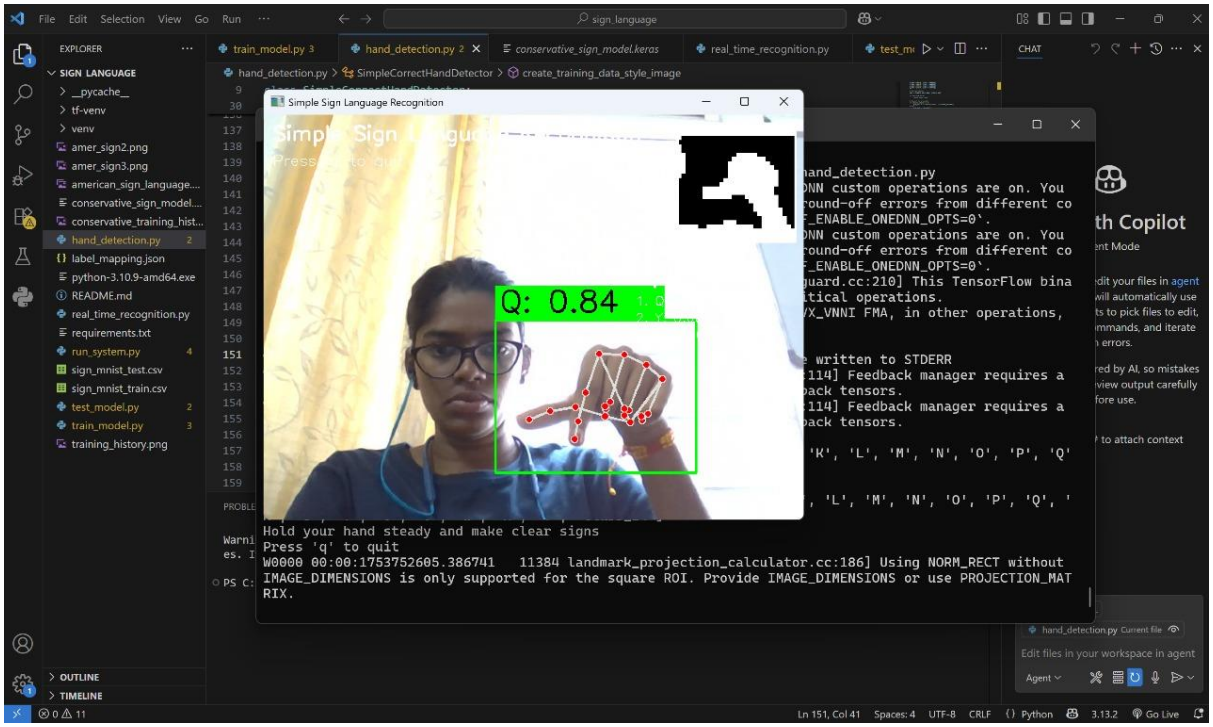
To demonstrate real-world applicability, the project implements a real-time recognition system using:

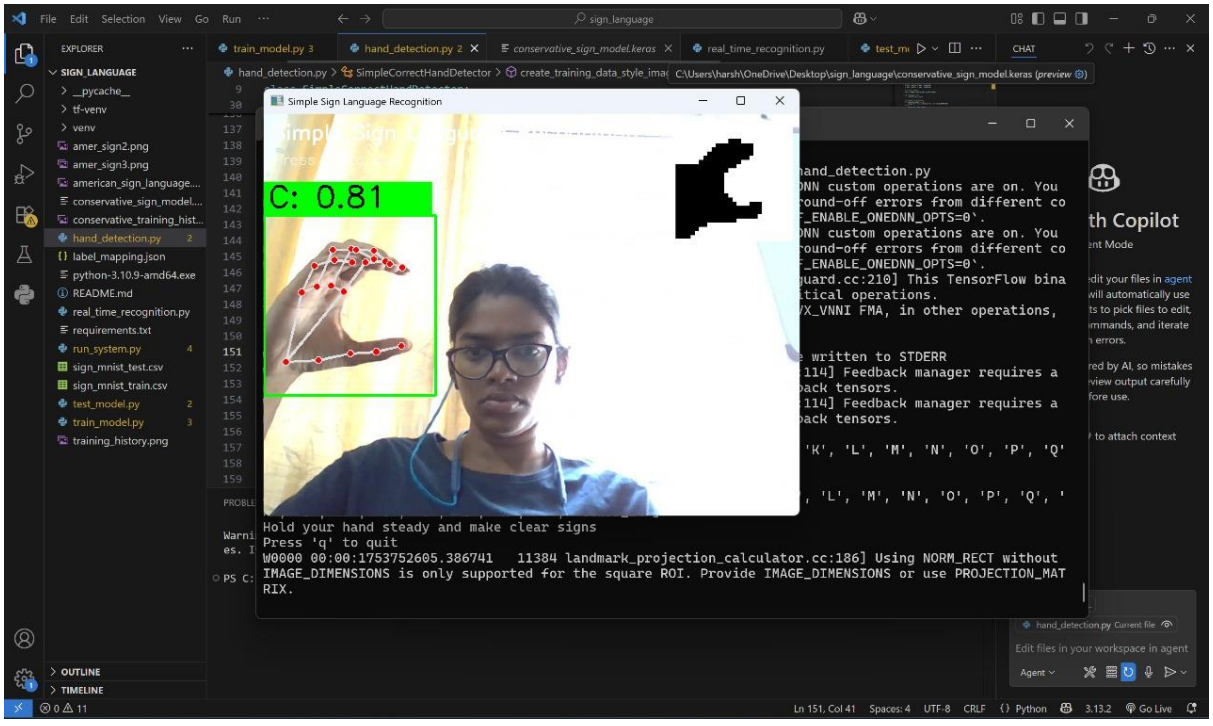
- **MediaPipe** for hand tracking and landmark extraction, providing robust detection across lighting and motion variance.
- **OpenCV** to capture live video feed and extract hand regions from each frame.
- **TensorFlow** model for gesture classification based on real-time input.
- **Threading** ensures smooth frame processing without UI lag.

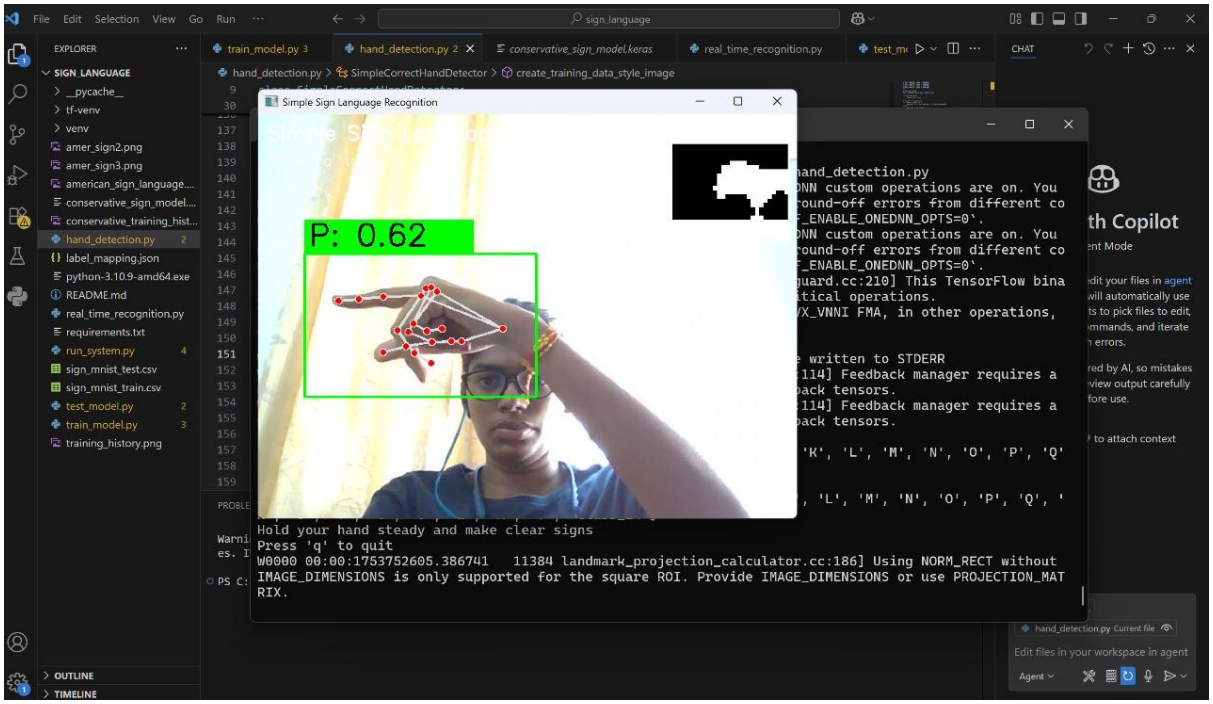
The system includes a text-building mechanism where recognized characters are held, appended into words, and displayed interactively. The user can **reset**, **add spaces**, or **clear the output** using key commands (A, S, C, etc.), providing an intuitive interface.

9. Results & Output Screens

The real-time sign language recognition system successfully identifies hand gestures using a webcam feed and overlays the predicted letter along with a confidence score. As shown in the output screens:







- The system detects hand landmarks using MediaPipe and highlights them with bounding boxes and keypoints.
- A prediction score (e.g., "Q: 0.84") is displayed on the screen, showing model confidence for each sign.
- Recognized signs are converted to text in real-time, enabling continuous word formation from gesture sequences.

This demonstrates the effectiveness and responsiveness of the system in practical usage scenarios.

Conclusion

This project successfully showcases the potential of **deep learning** and **computer vision** in developing a **real-time assistive communication system** for sign language recognition. By combining a minimalist yet effective **CNN architecture** with real-time **MediaPipe-based hand detection**, the system is capable of recognizing 24 static ASL gestures with considerable accuracy.

Key outcomes include:

- Reliable classification with a validation accuracy of over **75%** using cross-validation and dropout regularization.
- Real-time hand gesture recognition using a **standard webcam**, with interactive controls for text construction.
- Robust and smooth live performance in practical scenarios without the need for expensive hardware

This solution forms a **strong foundation** for more advanced assistive technologies. Future enhancements can focus on recognizing dynamic gestures (e.g., "J" and "Z"), supporting **multiple sign languages**, adding **text-to-speech conversion**, and improving model accuracy with larger and more diverse datasets.

References

1. Datamunge. *Sign Language MNIST Dataset*. Kaggle.
2. Xiao, H. et al. *Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms*. Zalando Research.
3. National Institute on Deafness and Other Communication Disorders (NIDCD). *Statistics and Resources on Hearing Loss*.
4. Abadi, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. TensorFlow & Keras Documentation.
5. Lugaresi, C. et al. *MediaPipe: A Framework for Building Perception Pipelines*. Google Research.