Sign Language Recognition System - Project Report

1. Introduction

Communication is essential to human interaction. But for individuals who are deaf or hard of hearing, expressing themselves can become a challenge when others don’t understand sign language. The motivation behind this project is to create a real-time solution that interprets sign language and bridges that communication gap using computer vision and deep learning technologies. Our goal was to design a system that captures hand gestures and instantly converts them into readable text.

2. Abstract

This project introduces a real-time American Sign Language (ASL) recognition system. It uses a webcam to capture hand gestures, identifies key hand landmarks using **MediaPipe**, and classifies the signs with a **MobileNetV2-based Convolutional Neural Network (CNN)**. The system was trained on the publicly available ASL Alphabet dataset from Kaggle.

To ensure the best results, we developed and compared **two different model training methods**:

* A basic custom CNN
* A transfer learning approach using MobileNetV2

Model development, training, and evaluation were performed in **Jupyter Notebook**, allowing for interactive exploration and fine-tuning. Once trained, the model was deployed in a **VS Code environment** to run real-time predictions using live webcam input. This approach allowed for a seamless blend of experimentation and production-readiness.

3. Tools Used

* **Programming Language**: Python 3.10
* **Libraries**: TensorFlow, MediaPipe, OpenCV, NumPy, TQDM, scikit-learn
* **Development Environments**: Jupyter Notebook (for training/testing), VS Code (for real-time prediction)
* **Hardware**: PC/Laptop with webcam
* **Dataset Source**: ASL Alphabet dataset (Kaggle)

4. Steps Involved in Building the Project

1. Dataset Preparation

    * Used the Kaggle ASL Alphabet dataset organized into folders A-Z.
    * Ensured balanced classes and consistent image format for training.

2. Data Preprocessing

    * Resized images to 96x96 pixels.
    * Normalized and augmented the dataset to improve model generalization.

3. Model Building & Training (Jupyter)

    * Trained two models: a custom CNN and a MobileNetV2-based transfer learning model.
    * Evaluated performance using accuracy and loss metrics.
    * Saved models in `.h5` format.

4. Real-Time Prediction (VS Code)

* Captured video using a webcam.
* Used MediaPipe for hand landmark detection.
* Passed processed frames through the trained model to classify gestures.
* Displayed the predicted letter in real-time on the video feed.

5. Conclusion

This project demonstrates how deep learning and computer vision can create impactful solutions to real-world communication challenges. The real-time Sign Language Recognition System accurately interprets static ASL gestures and converts them into text, making it easier for hearing-impaired individuals to communicate with those unfamiliar with sign language.

While this version focuses on the ASL alphabet, it sets the stage for future enhancements—such as dynamic gesture recognition, full sentence interpretation, and even speech integration. The combined use of Jupyter Notebook for development and VS Code for deployment made the workflow efficient and production-ready.

Prepared by: CHOPPARI HARSHAVARDHAN