Project Title- “**Sign Language Detection by using Hand Landmark-based Approach ”**

Abstract

Sign language detection is an essential step towards improving communication accessibility for the hearing-impaired community. This project aims to develop a vision-based system that detects and translates sign language gestures into text or speech in real time. Utilizing deep learning techniques, computer vision, and natural language processing, the model interprets hand movements and gestures, providing an interactive and effective means of communication. The system can be deployed in educational, social, and professional environments, bridging the gap between the hearing and speech-impaired individuals and the general public.

Sign language is an essential mode of communication for individuals with hearing and speech impairments. This research presents a **hand landmark-based sign language recognition system** that leverages **computer vision and video analytics** to interpret gestures in real-time. Using **Mediapipe Hands**, the system detects and tracks **21 key hand landmarks**, extracting crucial features such as **finger angles, distances, and orientations**. These features serve as input to a **machine learning or deep learning model**, enabling accurate classification of hand gestures representing **alphabets and numbers** . The system processes video streams using **OpenCV**, ensuring real-time gesture recognition and translation into text.

**Project Objectives**

* Implement a **computer vision-based model** to recognize and interpret sign language gestures for **alphabets and numbers** using **hand landmark detection**.
* Leverage **MediaPipe Hands** or **OpenCV-based detection** to extract **key hand landmarks**, such as **finger positions and joint angles**, for accurate gesture classification.
* Train a machine learning or deep learning model (e.g., **CNN, LSTM, or Random Forest**) to classify hand gestures into their respective sign language symbols.
* Optimize the system for **real-time** performance, ensuring smooth and accurate recognition of sign gestures **from live video input**.
* Develop an **interactive interface** that allows users to perform gestures and receive **text-based** or **speech-based** translations in real time.
* Implement **robust preprocessing techniques** to handle **varied lighting conditions, hand orientations, and occlusions**, ensuring reliable performance.
* Contribute to **bridging the communication gap** by providing an **AI-powered assistive technology** for those who rely on sign language.

**Methodology**  
The methodology involves the following steps:

**1. Data Acquisition**

* Collect sign language gesture datasets for **alphabets (A-Z) and numbers (0-9)**.
* Use **public datasets** (e.g., ASL datasets) or capture custom **video/image datasets** using a webcam.
* Store gesture images/videos with **labeled annotations** for supervised learning.

**2. Preprocessing & Hand Landmark Detection**

* Utilize **MediaPipe Hands** or **OpenCV with deep learning models** (e.g., **YOLO, Haar Cascade**) to detect and track hand movements.
* Extract **21 hand landmarks** (finger joints and palm) using **MediaPipe's Hand Tracking API**.
* Normalize hand coordinates to ensure robustness across different distances and angles.
* Apply **image processing techniques** (grayscale conversion, noise reduction, histogram equalization) to enhance gesture visibility.

**3. Feature Extraction**

* Extract **key features** from hand landmarks:
  + **Finger positions and angles** relative to the palm.
  + **Euclidean distances** between specific landmark points.
  + **Convex hull and defects** for gesture shape representation.
* Store these extracted features as **numerical vectors** for classification.

**4. Gesture Classification Using Machine Learning / Deep Learning**

**Approach 1: Machine Learning (ML) Model**

* Train an **ML model (e.g., SVM, Random Forest, k-NN, Decision Tree)** using extracted **landmark features**.
* Split the dataset into **training (80%) and testing (20%)** sets for evaluation.
* Use **cross-validation** to optimize model accuracy.

**Approach 2: Deep Learning (CNN/LSTM Model)**

* Train a **Convolutional Neural Network (CNN)** for gesture classification using raw images.
* Use **Long Short-Term Memory (LSTM)** networks for **sequential hand movement analysis**.
* Fine-tune the model with **data augmentation** to improve recognition across different lighting conditions.

**5. Real-Time Sign Language Recognition**

* Integrate the trained model with **a webcam feed** for real-time sign detection.
* Capture live video, extract hand landmarks, and classify gestures using the trained model.
* Display recognized **alphabet/number on-screen** or provide **text-to-speech** output.

**6. Performance Evaluation & Optimization**

* Assess the model using **accuracy, precision, recall, and F1-score**.
* Implement **error handling** for misclassified gestures.
* Optimize **frame rate** and **latency** for real-time performance.

**7. Deployment & User Interface**

* Develop a **GUI-based application** (using **Tkinter, PyQt, or Flask**) for user-friendly interaction.
* Provide options for **textual output** or **audio-based sign translation**.
* Ensure compatibility with **various devices (PC, mobile, embedded systems like Raspberry Pi)**.

**Key Findings**  
The sign language detection system demonstrated a high level of accuracy in recognizing predefined gestures. Using a deep learning-based approach significantly improved detection efficiency and real-time processing. Challenges such as varying lighting conditions, background noise, and hand occlusion were addressed by data augmentation and advanced preprocessing techniques. The findings suggest that AI-powered sign language recognition can be a crucial tool for communication accessibility.

**Step-wise Solution Approach**

* **Step 1:** Data Collection – Acquire a diverse dataset of sign language images and videos.
* **Step 2:** Data Preprocessing – Convert images to grayscale, resize, and apply augmentation.
* **Step 3:** Feature Extraction – Use CNNs to identify critical patterns in hand movements.
* **Step 4:** Model Training – Train CNN-based pretrained models such as VGG16, ResNet, or MobileNet using the MNIST dataset before fine-tuning for gesture classification.
* **Step 5:** Integration – Develop a user-friendly application that translates detected gestures into text or speech.
* **Step 6:** Testing & Optimization – Evaluate the model on unseen data and optimize it for real-world deployment.

Reference

1. Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv.
2. Zhang, J., et al. (2021). A Deep Learning Approach for Sign Language Recognition. IEEE Transactions on Multimedia.
3. American Sign Language Dataset (ASL) - Kaggle.
4. OpenCV and TensorFlow Documentation for Real-time Image Processing.

Team:

1. Kotha Bharath Reddy(E22CSEU1103)
2. Pratheek Reddy (E22CSEU1088)