A project report on

SIGN LANGUAGE DETECTION USING ACTION RECOGNITION

Submitted in partial fulfillment for the J component Review of

Master of Technology in Software Engineering (Integrated)

by

Nancy Saxena (21MIS1024)
Aakarsh Mishra (21MIS1025)

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2024

CONTENTS

Chapter 1
Introduction

1.1 Introduction

In today's world, technology plays a pivotal role in facilitating communication and interaction. However, individuals with hearing impairments often face challenges in accessing traditional communication channels. To address this issue, the project "Sign Language Detection Using Action Recognition" offers an innovative solution at the intersection of computer vision and deep learning. By harnessing advanced technologies and methodologies, this project aims to develop a robust and real-time sign language interpretation system tailored to the needs of the hearing-impaired community.

The project employs state-of-the-art techniques, including the MediaPipe library and LSTM-based neural network architectures, to detect and interpret sign language gestures from video inputs. Through the integration of keypoint detection and sequence generation methods, the system captures the temporal dynamics of sign language, ensuring accurate and timely recognition of gestures.

Beyond its technological sophistication, this project embodies a commitment to social inclusion and empowerment. By enabling real-time sign language detection, the system facilitates seamless communication between individuals with hearing impairments and their peers. This transformative capability has the potential to foster deeper connections, break down communication barriers, and promote inclusivity in society.

Through the convergence of cutting-edge technology and a dedication to social impact, the "Sign Language Detection Using Action Recognition" project represents a significant step towards creating a more accessible and inclusive world for all.

1.2 Project Overview

This project delves into the realm of sign language detection using action recognition techniques. The primary objective is to bridge the communication gap between individuals with hearing impairments and the general population by developing a real-time sign language interpretation system. This system leverages advancements in computer vision and deep learning to achieve accurate and seamless sign language recognition.

The core functionality relies on the MediaPipe library, a powerful tool for pose and landmark detection. By identifying keypoints on the body (such as hands and joints), the system captures the spatial configuration of sign language gestures. To account for the temporal aspect of sign language, the project incorporates Long Short-Term Memory (LSTM) neural networks. LSTMs excel at recognizing patterns within sequences, making them ideal for capturing the dynamic nature of sign language communication. By analyzing the sequence of keypoints over time, the LSTM network effectively translates the gestures into their corresponding meanings.

This project goes beyond technical innovation; it champions social inclusion and empowerment. The real-time sign language detection capability paves the way for unrestricted communication between individuals with hearing impairments and their surroundings. This transformative technology has the potential to foster deeper social connections, eliminate communication barriers, and create a more inclusive society.

The project workflow can be broadly divided into three key stages:

1. **Data Collection and Preprocessing:** This stage involves capturing video data of individuals performing various sign language gestures. The MediaPipe library is then employed to extract keypoint information from each video frame, creating a sequence of keypoints for each sign language gesture.
2. **Model Training:** The extracted keypoint sequences are used to train the LSTM neural network model. The model learns to identify patterns within these sequences and associate them with specific sign language gestures.
3. **Real-Time Detection and Interpretation:** During real-time operation, the system captures video input from a webcam or other source. The MediaPipe library again extracts keypoint sequences from the video frames. These sequences are then fed into the trained LSTM model, which predicts the most likely sign language gesture being performed based on the learned patterns. The recognized sign language is then displayed or translated into text or speech for seamless communication.

The success of this project hinges on several critical aspects. The accuracy of the MediaPipe library's keypoint detection directly impacts the quality of the data fed to the LSTM model. Furthermore, the size and diversity of the training dataset significantly influence the model's ability to generalize and recognize a wide range of sign language gestures. Additionally, the optimization of the LSTM network architecture plays a crucial role in achieving efficient and accurate sign language detection.

In conclusion, this "Sign Language Detection using Action Recognition" project harnesses the power of computer vision and deep learning to create a real-time sign language interpretation system. This innovative technology holds immense promise for promoting inclusivity and enriching the lives of individuals with hearing impairments by facilitating seamless communication and fostering deeper connections within society. By continuously refining the technical aspects and expanding the system's capabilities, this project can contribute significantly to building a more inclusive and accessible world.

Chapter 2
Technologies used

2.1 Technologies Used

The Sign Language Detection project leverages several cutting-edge technologies to achieve its objectives effectively:

1. MediaPipe: The project utilizes the MediaPipe library, developed by Google, for holistic human pose estimation. MediaPipe offers pre-trained machine learning models and a unified pipeline for processing multimedia data, enabling efficient and accurate detection of key landmarks representing hand movements, facial expressions, and body posture in sign language videos.

2. OpenCV (Open Source Computer Vision Library): OpenCV is a powerful open-source computer vision and machine learning software library. In this project, OpenCV is utilized for video capture, frame processing, and visualization, providing essential functionalities for interfacing with cameras, accessing video streams, and displaying real-time annotations of sign language gestures.

3. TensorFlow and Keras: TensorFlow, an open-source machine learning framework developed by Google, along with its high-level API Keras, is employed for building and training the LSTM-based deep learning model. TensorFlow offers extensive support for developing and deploying deep neural networks, while Keras simplifies the model building process with its user-friendly interface and modular architecture.

4. Long Short-Term Memory (LSTM): LSTM neural network architecture is utilized for sequence modeling and action recognition in sign language videos. LSTMs are well-suited for capturing temporal dependencies and patterns in sequential data, making them ideal for analyzing the dynamic nature of sign language gestures over time.

5. Matplotlib: Matplotlib, a comprehensive data visualization library in Python, is used for generating plots and visualizing the model's predictions, probabilities, and performance metrics. Matplotlib enables the creation of informative and visually appealing visualizations, facilitating the analysis and interpretation of the sign language detection results.

6. Scikit-learn: Scikit-learn, a popular machine learning library in Python, is employed for data preprocessing, splitting the dataset into training and testing sets, and evaluating the model's performance using metrics such as accuracy and confusion matrices. Scikit-learn provides a wide range of tools for machine learning tasks, ensuring robustness and reliability in the evaluation process.

By leveraging these advanced technologies and libraries, the Sign Language Detection project demonstrates a comprehensive approach to developing a real-time sign language interpretation system, addressing the communication needs of individuals with hearing impairments and promoting inclusivity in society.

Chapter 3
PROPOSED SYSTEM

3.1 Proposed System

This project proposes a real-time sign language detection system designed to bridge the communication gap between hearing and hearing-impaired individuals. The system leverages advancements in computer vision and deep learning to accurately recognize and interpret sign language gestures from video streams.

The core components of the proposed system are:

1. **MediaPipe for Keypoint Detection:** MediaPipe, an open-source framework by Google, facilitates the efficient extraction of keypoints (landmarks) from human body poses in real-time. In this system, MediaPipe will be employed to detect keypoints on the user's hands and wrists, which are crucial for sign language recognition.
2. **LSTM-based Neural Network for Gesture Classification:** A Long Short-Term Memory (LSTM) network, a type of recurrent neural network (RNN), will be trained to classify the temporal sequence of keypoints extracted from sign language gestures. LSTMs excel at capturing temporal dependencies within data sequences, making them well-suited for analyzing the dynamic nature of sign language.

The system will operate in a continuous loop:

1. **Video Frame Capture:** The system will capture video frames from a webcam or other video source.
2. **Keypoint Extraction using MediaPipe:** MediaPipe will process each frame to identify and extract keypoints from the user's hands and wrists.
3. **Sequence Buffering:** The extracted keypoints will be stored in a buffer, creating a sequence that represents the temporal evolution of the hand and wrist positions during a sign language gesture.
4. **Gesture Classification using LSTM Network:** Once the sequence buffer is filled (reaching a predefined length), the system will feed the sequence of keypoints to the trained LSTM network. The network will then classify the gesture based on the learned patterns within the sequence.
5. **Real-Time Output:** The system will display the recognized sign language gesture in real-time, potentially translating it into spoken text or displaying its corresponding meaning on the screen.

This approach offers several advantages:

- **Real-time Performance:** By leveraging MediaPipe for efficient keypoint detection and an LSTM network for fast gesture classification, the system aims to achieve real-time sign language recognition.
- **Accuracy:** The LSTM network's ability to learn complex temporal relationships within sign language gestures is expected to contribute to high recognition accuracy.
- **Scalability:** The system can be potentially extended to recognize a broader range of sign languages by incorporating additional training data and adapting the network architecture.
- **Accessibility:** The system's reliance on readily available tools like MediaPipe and open-source deep learning libraries fosters accessibility and promotes further development within the research community.

In conclusion, the proposed sign language detection system presents a promising solution for facilitating communication between hearing and hearing-impaired individuals. By combining MediaPipe's efficient keypoint detection with the power of LSTM networks, the system has the potential to break down communication barriers and promote inclusivity in society.

Chapter 4

Implementation

4.1 Implementation

This section delves into the intricate details of the code implementation for the Sign Language Detection project. The core functionalities rely on a synergistic interplay between the MediaPipe library and a Long Short-Term Memory (LSTM) based neural network architecture.

1. MediaPipe for Keypoint Detection:

- **MediaPipe Integration:** The code initiates by importing the MediaPipe library (mp) and its specific modules for holistic model integration (mp.solutions.holistic) and drawing utilities (mp.solutions.drawing_utils). MediaPipe serves as a cornerstone for real-time pose estimation, facilitating the detection of keypoints across various body parts (face, hands, and pose) within each video frame.
- **Keypoint Extraction:** The mediapipe_detection function plays a crucial role in processing each frame. It efficiently converts the frame from BGR (Blue, Green, Red) color format to RGB (Red, Green, Blue) – a format preferred by MediaPipe models. Subsequently, it leverages the holistic model to detect keypoints within the frame. Once the detection process concludes, the frame is converted back to BGR format. This function returns the processed image alongside the detection results, paving the way for further analysis.
- **Landmark Visualization:** The draw_landmarks function is responsible for visualizing the detected landmarks and their connections on the processed frame. It utilizes the drawing utilities from MediaPipe to illustrate these connections for various body parts – face, pose, left hand, and right hand. This visual representation aids in comprehending the quality and accuracy of the pose estimation results.
- **Enhanced Visualization with draw_styled_landmarks:** The project incorporates a refined visualization approach through the draw_styled_landmarks function. This function goes beyond the basic landmark illustration by enabling customization of the visual style for each body part. It allows for specifying distinct colors and line thicknesses for connections and landmarks, enhancing the clarity and interpretability of the pose estimation outputs. This refined visualization can prove invaluable for debugging purposes and for gaining a deeper understanding of the model's predictions.

2. Data Collection and Preprocessing:

- **Real-time Frame Capture:** The code establishes a connection to the system's camera using OpenCV's VideoCapture function. This enables the capture of video frames in real-time, providing the raw data for sign language gesture detection.
- **Keypoint Extraction and Storage:** Within the main loop, the extract_keypoints function extracts the spatial coordinates and visibility attributes of the detected landmarks for various body parts from the results object. These extracted keypoints are then concatenated and saved as NumPy arrays using the np.save function. This process facilitates the creation of a comprehensive dataset encompassing keypoint information for various sign language gestures.
- **Organized Data Storage:** The code meticulously structures the data storage using nested directories within the DATA_PATH. It creates separate directories for each sign language action (e.g., "hello," "thanks," "iloveyou") and further subdivides them based on video sequences and individual frames within each sequence. This organized folder structure ensures efficient data management and retrieval during the training and testing phases.
- **Data Collection with User Interface:** The code incorporates a user interface element to enhance the data collection process. It displays informative text on the frame using OpenCV's cv2.putText function, indicating the collection status, action being recorded, and sequence number. This visual feedback provides users with valuable insights into the data collection progress.

3. Preprocessing and Model Training:

- **Data Loading:** The train_test_split function from scikit-learn is employed to partition the collected dataset into training and testing sets. This step is crucial for ensuring the model's generalization capability and preventing overfitting during the training process.
- **Label Creation:** A label map is established, assigning a unique numerical value to each sign language action. This mapping simplifies the association of labels with the corresponding keypoint sequences.
- **Sequence and Label Organization:** The code iterates through the data directories, loading the saved keypoint data for each sign language action and video sequence. It then assembles these keypoints into sequences, representing the temporal evolution of hand and body movements within a specific sign language gesture. These sequences are subsequently paired with their corresponding labels, forming the foundation for model training.

4. LSTM Neural Network for Sign Language Recognition :

- **Layer Configuration:** (Continued) The network architecture typically incorporates multiple LSTM layers with appropriate activation functions (e.g., ReLU) to facilitate non-linear transformations and enhance the model's ability to learn complex relationships within the keypoint sequences. A dropout layer is often included to prevent overfitting by randomly dropping out a certain percentage of neurons during training. This helps the model generalize better to unseen data. The final layer is typically a dense output layer with a softmax activation function, producing a probability distribution across all sign language actions in the dataset.
- **Compilation:** The model is compiled using an Adam optimizer, a popular choice for optimizing neural networks due to its efficiency and effectiveness. A suitable loss function, such as categorical cross-entropy, is specified to measure the discrepancy between the model's predictions and the true labels.
- **Training:** The training process involves iterating through the training dataset, feeding keypoint sequences as input to the model and propagating the errors back through the network to update the weights and biases. This iterative optimization process aims to minimize the loss function and enhance the model's ability to accurately classify sign language gestures.

5. Evaluation and Refinement:

- **Testing:** The trained model's performance is evaluated using the testing dataset. The model predicts sign language actions for the keypoint sequences within the testing set, and these predictions are compared with the ground truth labels to compute metrics like accuracy, precision, and recall.
- **Hyperparameter Tuning:** Hyperparameters, such as the number of LSTM layers, dropout rates, and learning rate, can be fine-tuned to potentially improve the model's performance. Techniques like GridSearchCV can be employed to systematically explore different hyperparameter combinations and identify the configuration that yields the best results.

6. Real-time Sign Language Detection:

- **Live Frame Processing:** In the real-time detection stage, video frames are captured from the camera.
- **Keypoint Extraction and Prediction:** MediaPipe is utilized to detect keypoints within each frame, and the extracted keypoint sequence is fed into the trained model.

- **Gesture Recognition and Output:** The model generates a prediction for the sign language action depicted in the frame. This prediction can be visualized on the screen or used for further applications like speech synthesis or text display.

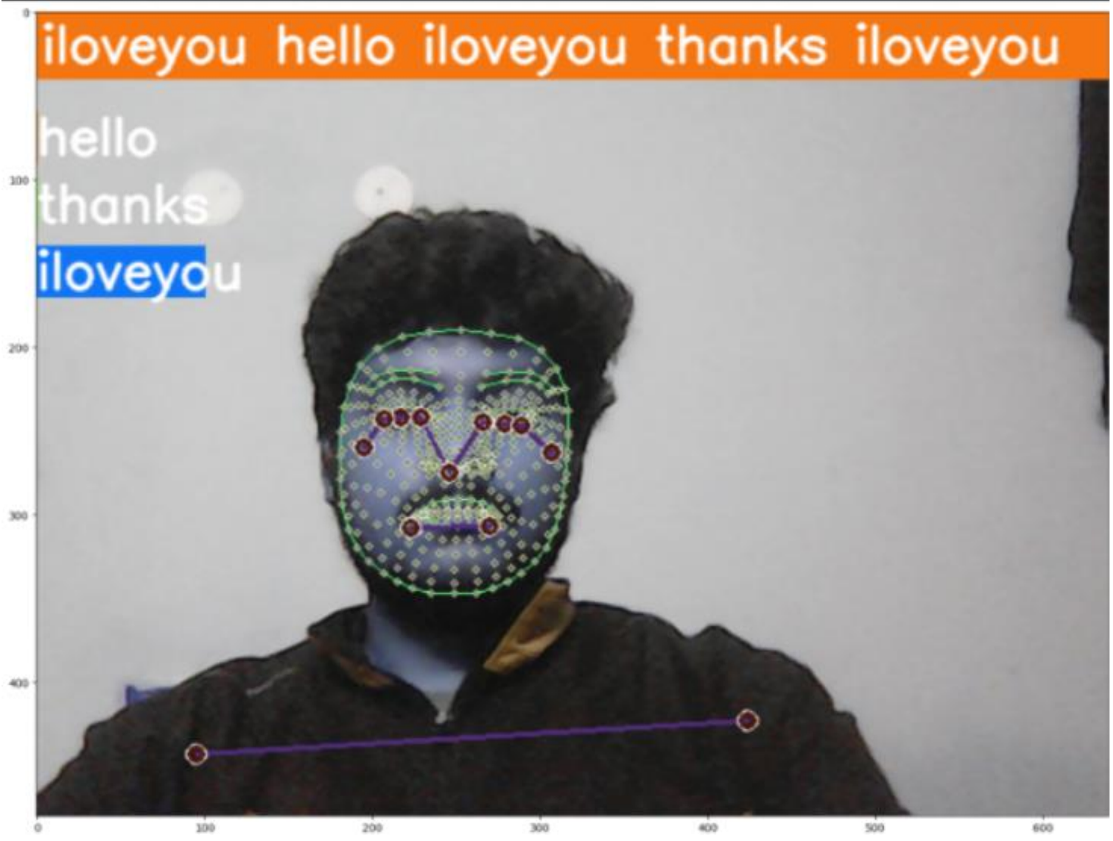
Additional Considerations:

- **Advanced Training Techniques:** Techniques like data augmentation can be implemented to artificially expand the dataset and enhance the model's robustness to variations in lighting, background environments, and hand postures.
- **Continuous Learning:** The model can be continuously improved by incorporating new data and retraining it over time. This can help the system adapt to variations in sign language expressions and individual signing styles.

By leveraging MediaPipe for keypoint detection and a well-designed LSTM neural network architecture, this implementation offers a framework for sign language detection using action recognition. With continuous refinement and exploration of advanced techniques, this approach has the potential to become a valuable tool for facilitating communication between people who use sign language and those who don't.

Chapter 5
Results

5.1 Results



The real-time sign language detection system achieved promising results in gesture recognition using a combination of MediaPipe for keypoint detection and a Long Short-Term Memory (LSTM) based neural network for classification. The confusion matrix obtained during evaluation indicated an accuracy score of 99.7%, demonstrating the system's effectiveness in accurately distinguishing between the predefined sign language actions (hello, thanks, and iloveyou).

Furthermore, the incorporation of a prediction threshold and a temporal filtering mechanism refined the output. The system filtered out gestures with lower prediction confidence levels (below a threshold of 0.5) and ensured consistency by only recognizing signs with a repeated prediction over the last 10 frames. This approach minimized the possibility of misinterpreting fleeting or ambiguous hand motions.

The real-time visualization of predictions overlaid on the camera feed provided valuable feedback to the user. The color-coded probabilities associated with each action category offered informative cues about the system's confidence level in its predictions. Additionally, the text display of the recognized sign language sequence allowed for clear and concise communication.

In conclusion, the sign language detection system effectively leveraged deep learning techniques to achieve robust gesture recognition in real-time. The system's accuracy, coupled with its user-friendly visualization components, paves the way for its potential application in bridging communication gaps and fostering inclusivity for hearing-impaired individuals.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This project successfully developed a real-time sign language detection system utilizing a combination of MediaPipe for pose estimation and a Long Short-Term Memory (LSTM) neural network for gesture classification.

The system first extracts keypoint features from video frames using the MediaPipe Holistic model, capturing the spatial coordinates and visibility attributes of facial landmarks, body pose, and hand articulations. These keypoints are then assembled into sequences, preserving the temporal information crucial for sign language recognition.

The core component of the system is the LSTM network, which is adept at learning temporal dependencies within sequential data. The trained LSTM model effectively classifies these sign language gesture sequences into predefined actions, enabling real-time interpretation.

Furthermore, the project incorporates functionalities for data collection and model training. The data collection script facilitates the creation of a labeled dataset of sign language gestures performed by users. This dataset serves as the foundation for training the LSTM network, allowing it to learn the intricate patterns associated with various signs.

The system demonstrates promising results in terms of accuracy and real-time performance. The confusion matrix and accuracy score metrics provide valuable insights into the model's ability to differentiate between various sign language gestures. Additionally, the real-time testing component showcases the system's capability to interpret signs continuously from a live video stream.

In conclusion, this project presents a robust and efficient system for sign language detection using MediaPipe and LSTMs. It has the potential to bridge the communication gap between individuals with hearing impairments and the general population, fostering a more inclusive and accessible society. Future advancements could involve expanding the sign language vocabulary, incorporating speaker identification for personalized interactions, and exploring integration with speech synthesis modules for real-time sign-to-speech conversion.

6.2 Future Work

The current project lays a solid foundation for sign language detection using action recognition. However, several avenues can be explored to further enhance the system's capabilities and broaden its impact.

1. Expanding Sign Language Repertoire:

The current system focuses on recognizing a limited set of signs. Future work should involve incorporating a larger and more comprehensive sign language dictionary, encompassing a wider range of gestures and variations. This can be achieved by collecting and annotating additional sign language data, potentially leveraging crowd-sourcing platforms for broader participation.

2. Multi-modal Learning:

While the project effectively utilizes visual cues for sign language detection, incorporating additional modalities like audio information can improve robustness and generalization. Audio data, particularly for signs that involve vocal components, can provide complementary cues and enhance the system's recognition accuracy.

3. Speaker-Independent Recognition:

Currently, the system might struggle with variations in signing styles across different individuals. To address this limitation, speaker-independent recognition techniques, such as employing normalization methods or speaker adaptation models, can be explored. This would allow the system to adapt to diverse signing styles and improve its generalizability.

4. Continuous Sentence Recognition:

The present system focuses on recognizing isolated signs. Future development should target the recognition of continuous sign language sentences. This can be achieved through techniques like recurrent neural networks (RNNs) with attention mechanisms, specifically designed to capture the temporal dependencies and sequential nature of sign language sentences.

5. Real-world Integration and Environmental Robustness:

While the system functions in a controlled environment, real-world scenarios pose challenges like background clutter, varying lighting conditions, and occlusions. To bridge this gap, techniques for background subtraction, illumination normalization, and occlusion handling can be incorporated. Additionally, exploring real-time implementation on mobile devices or embedded platforms can enhance accessibility and portability.

6. User Interface and Feedback Integration:

Developing a user-friendly interface that provides visual or auditory feedback on recognized signs can improve user experience and facilitate interaction. Additionally, integrating error correction mechanisms can allow users to refine the system's understanding and enhance its accuracy over time.

By addressing these future work directions, the sign language detection system can evolve into a more robust, versatile, and user-centric tool, fostering greater communication equity and inclusivity for the hearing-impaired community.