Minor Project Report on

HarmonyAI : A web based ML model for Hand sign language Translation

Submitted in partial fulfilment of the
requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

Submitted by

ANUBHAB MOHANTY

Regd. No.: 2111100459

SRITAM MAHAPATRA

Regd. No.: 2111100460

Under the guidance of

MRS. PRANATI MISHRA

Asst. Professor

School Of Computer Science
Odisha University of Technology and Research
Bhubaneswar, Odisha – 751029

Department of Computer Science and Engineering

ODISHA UNIVERSITY OF TECHNOLOGY AND RESEARCH,
BHUBANESWAR

CERTIFICATE

This is to certify that the seminar report entitled **HarmonyAI : A web based ML model for Hand sign language Translation** submitted by **Anubhab Mohanty, and Sritam Mahapatra** bearing registration number **2111111459**, **and 2111100460** respectively to the Department of Computer Science and Engineering, Odisha University of Technology and Research, formerly College of Engineering and Technology, Bhubaneswar, is a record of Bonafede research work under my supervision and I consider it worthy of consideration for partial fulfilment of the requirements for the award degree of Bachelor of Technology in Computer Science and Engineering under Odisha University of Technology and Research, Bhubaneswar.

Mrs. Pranati Mishra

(Guide)

ACKNOWLEDGEMENT

It is my privilege and solemn duty to express my deepest sense of gratitude to **Mrs. Pranati Mishra** of the Department, under whose guidance We carried out this work. We am indebted, for her valuable supervision, heart full cooperation and timely aid and advice till the completion of the thesis in spite of her pressing engagements. We would like to express my deep sense of gratitude to **Mrs. Pranati Mishra**, Coordinator of the Department for encouragement and inspiration throughout my project work. We wish to record my sincere gratitude to our respected Head of the School, **Prof. Ranjan kumar Dash** for his constant support and encouragement in preparation of this report.

We take this opportunity to express my hearty thanks to all those who helped me in the completion of my seminar report work. We am very grateful to the author of various articles on the Internet, for helping me become aware of the research currently ongoing in this field.

We am very thankful to my parents for their constant support and love. Last, but not the least, We would like to thank my classmates for their valuable comments, suggestions and unconditional support.

Anubhab Mohanty

Sritam Mahapatra

DECLARATION

I certify that

i. The work contained in the seminar report is original and has been done myself under the general supervision of my supervisor.
ii. The work has not been submitted to any other Institute for any degree or diploma.
iii. We have followed the guidelines provided by the Institute in writing the report.
iv. Whenever We have used materials (data, theoretical analysis, figures, text) from other sources, We have given due credit to them by citing them in the text of the seminar report and giving their details in the references.
v. Whenever We have quoted written materials from other sources, We have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Anubhab Mohanty

Sritam Mahapatra

ABSTRACT

Natural human-computer interaction (NHCI) relies on sign language recognition, but it is still difficult since real-time dynamic movements and static hand postures must be precisely captured. While vision-based methods offer a natural way to communicate, it can be difficult to correctly encode temporal and spatial information.

A real-time hand sign recognition system can bridge the communication gap by instantly translating sign language into text or spoken language.Such a system provides specially-abled individuals with autonomy and independence in communication, empowering them to engage more effectively in various settings.By enabling real-time communication, the system enhances inclusivity, accessibility, and equality for the specially-abled community in daily interactions and societal integration.

This paper presents a revolutionary multi-stream deep learning architecture that combines 3D convolutional networks, skeletal data, and local region images to recognize words in sign language. The architecture makes use of transfer learning strategies that are modeled after human learning processes as well as domain expertise.

Our method addresses alignment and recognition issues by introducing a "Sequence-to-sequence" learning approach to the WLASL dataset.

Our method does not require an additional alignment step, and it matches state-of-the-art sign recognition rates. Through the integration of complementary data representations, the multi-stream architecture accurately captures spatial-temporal information.

Keywords:

Sign-language translation (ASL), Deep learning, WLASL, Inflated 3D ConvNet (I3D), TGCN, Transfer Learning, SaaP.

CONTENTS

TITLE **PAGE**

CHAPTERS

LIST OF FIGURES

1. INTRODUCTION

The foundation of human contact is communication, which makes it easier to share information, feelings, and ideas. Traditional forms of communication, however, can pose serious obstacles for people with speech and hearing impairments, making it more difficult for them to express themselves and interact meaningfully with others.

In light of this, our research aims to use state-of-the-art technology to close these communication gaps and provide deaf and mute people the freedom to express themselves and engage with their surroundings. Our goal is to create a comprehensive system for hand sign identification, recognition, and translation by applying cutting-edge computer vision and natural language processing techniques. We will leverage the extensive resources offered by the **WLASL (Word-Level American Sign Language)** dataset[1].

There is almost negligible commercial application exists in the market to address this problem. Need for an automated and accurate system to recognize and interpret sign language and hand gestures. Manual interpretation is laborious, subjective, and error-prone.

Fundamentally, the goal of our research is to offer workable answers to the problems that people with speech and hearing impairments encounter daily. Through the utilization of computer vision, our system can precisely identify and decipher hand signals in real time, providing users with an unparalleled level of simplicity and efficiency while communicating through sign language.

Furthermore, our approach provides a smooth translation of sign language motions into textual representations, going beyond simple recognition. This feature not only makes it easier for people who are deaf or mute to communicate with people who are not familiar with sign language, but it also creates new avenues for accessibility in several settings, such as social interactions, work, healthcare, and education.

Our technique is a lifesaver for deaf people, giving them effective communication in contexts where other approaches are inadequate. Our technology enables deaf and mute people to completely engage in society on their own terms, whether it is through accessing educational materials, engaging in online debates, or communicating vital information in emergencies.

Our project intends to improve the quality of life for deaf and mute people by removing barriers to communication and promoting inclusivity, giving them more independence, autonomy, and dignity. We are dedicated to improving accessibility and enabling every member of our community to connect and communicate with one another more successfully than previously through continued research, development, and cooperation.
Our motivation is to; Increase accessibility and enable natural human-computer interaction, improve assistive technologies for the deaf/hard-of-hearing community, foster seamless communication across diverse groups, and enhance the capabilities of interfaces, robotics, and virtual reality systems

2. BACKGROUND STUDY

2.1. Literature Survey

Hezhen Hu et al.[1] The study suggests replacing the tedious feature extraction process needed by traditional machine learning methods with deep convolutional neural networks (CNNs) to automatically identify hand motions in English Sign Language (ESL) that represent the alphabet. The input hand gesture photos are fed into a proprietary 3-layer CNN architecture that consists of dense layers and convolutional layers with max-pooling to automatically extract hierarchical features and predict the 27 output classes. The authors use GPU acceleration to speed up the computationally demanding process and transfer learning with pre-trained CNN weights to increase training performance on the small 810-picture dataset. The CNN model shows encouraging results in experiments, achieving 100% training accuracy, 82% validation accuracy, and 70% test accuracy. The authors recommend increasing the number of convolutional layers. this domain remains to be resolved.

Lih-Jen Kau et al.[2] The wearable glove technology proposed in this study can translate Taiwanese sign language into text or speech in real time. In contrast to earlier systems, it tracks hand motion trajectories and recognizes finger positions by intelligently combining data from several sensor modalities, such as gyroscope, accelerometer, and flex sensors. Using an algorithm, the system eliminates intermediate places and identifies genuine sustained motions. motions that are recognized are encoded and sent over Bluetooth to a smartphone, which uses a text-to-speech engine to translate the motions into voice. Five participants in an experiment showed that gesture recognition accuracy may reach 94.56%. Comparing this novel technique to earlier systems, portability and accuracy are enhanced. fusion system overcomes prior constraints and enables a highly portable and accurate real-time sign language translating capacity.

Palani Thanaraj Krishnan et al.[3] This research presents a novel approach that makes use of deep convolutional neural networks (CNNs) for English Sign Language alphabet identification. For automatic hierarchical feature extraction from input images—which does away with the requirement for tedious feature extraction—a special 3-layer CNN architecture is essential. Convolutional layers, thick layers, max-pooling, and a softmax output layer are all included in the classification model. On a small dataset of 810 photos, transfer learning with pre-trained weights and GPU acceleration speed up training. The accuracy of the experimental results is promising: 100% for training, 82% for validation, and 70% for testing. Further efficiency increases can be achieved by investigating more intricate CNN designs and optimization techniques, according to the authors. All things considered, the work shows how deep CNNs can automatically represent features and create an accurate sign language translation model straight from picture input.

Alptekin Orbay et al.[4] To enhance communication with the Deaf, the document addresses the application of tokenization learning for translation from sign language to English. It proposes semi-supervised tokenization algorithms without additional labeling requirements, addressing the problems of insufficient annotated data and expensive gloss-level annotations. Significant gains in sign-to-text translation are achieved by comparing different approaches in diverse contexts, as demonstrated by thorough experiments presented in this research. It also looks into the classification of hand shapes and temporal modeling in sign language films using deep learning and computer vision techniques like CNNs and RNNs. It also covers the efficiency of various input formats and attention systems in the translation process. All things considered, the paper offers a thorough examination of the difficulties and developments in tokenization learning-based sign language translation.

Ridley College et al.[5]The research on a deep learning-based system for translating sign language is presented in this publication. It emphasizes how crucial sign language understanding is to facilitating communication for those with language and hearing difficulties. The main elements covered are: 1) A network for hand location that uses Faster R-CNN to identify hand areas in pictures or videos used for sign language. 2) A framework for LSTM encoder-decoders and a 3D CNN feature extraction network that learns motion and context from video frames to recognize sign language. 3) A combined method for deciphering sign language from RGB photos and videos that makes use of hand location, 3D CNN features, and LSTM. On a common word dataset, experimental results show up to 99% recognition accuracy, exceeding existing techniques.

Akshatha Rani K et al.[6] Machine learning techniques have been used in several projects to investigate sign language recognition. Certain methods combined classifiers like support vector machines and k-nearest neighbors with handmade features like principal component analysis, Karhunen-Loeve transform, and Sharma et al. 2013 (Singha & Das, Ahuja & Singh, 2015). Convolutional neural networks (CNNs) are deep learning models that have been used in other research to achieve improved accuracies in end-to-end recognition from sign photos (Bheda & Radpour 2017, Tolentino et al. 2019, Pinto Jr et al. 2019, Hurroo & Walizad 2020). According to Cheok et al. (2017) and Saravana Kumar & Iyangar (2018), important procedures included skin color detection, hand segmentation, preprocessing such as thresholding, and classification using neural networks trained on sign image datasets like ASL. The mediapipe hand tracking framework was also utilized in recent research to achieve reliable hand detection.

Daniel Stein et al.[7] Research teams have looked into translating and recognizing sign language using data-driven techniques. A statistical machine translation (SMT) system for German sign language was created by Stein et al. (2006). An SMT system for Chinese to Taiwanese sign language was developed by Chiu et al. in 2007. The translation of Spanish text into sign representation and subsequently into animation was suggested by San-Segundo et al. (2006). Methods include hand tracking and visual feature extraction. Annotation variations and inadequate data are challenges. Early research on American and Irish Sign Language translation by Stein et al. (2012) combines sign recognition with SMT with encouraging results. On complicated datasets like ATIS, however, translation quality was impacted by recognition errors.

Holden, E. J. et al.[8] Several studies have looked into employing computer graphics to translate text into animations in sign language. By extracting and presenting pre-stored sign image sequences, Kawai and Tamura (1985) and Kamata et al. (1989) created systems that produce sign language animations from text input. On a microcomputer, Waldron and Choi (1987) converted text to fingerspelling and synthetic speech. Rather than merely displaying pre-rendered graphics, Holden and Roy's Hand Sign Translator (HST) system used a novel kinematics-based method, modeling the hand skeleton and animating hand movements to generate signs from English text input. Key processes included depth sorting, applying 3D form models, computing intermediate skeleton poses using interpolation, retrieving joint angle data for hand poses from a dictionary, and rendering the animated hand. The HST demonstrated translating whole English sentences to continuous sign language animation while being restricted to two-handed signs.

Babita Sonare et al.[9] Machine learning techniques for vision-based sign language translation and recognition have been studied in a number of papers. For gesture recognition, early approaches such as those by Arun and Geethu (2005) and Shreyashi and Kumbhar (2006) used methods like template matching and PCA. To increase accuracy, recent developments have concentrated on deep learning models like CNNs and RNNs. A CNN-RNN hybrid was introduced by Lai and Svetlana (2007) for the purpose of extracting spatial and temporal features from sign language films. Our project intends to combine spatial and temporal information for gesture identification in order to construct a real-time sign language translation system using the CNN-LSTM architecture. User identification, video streaming, preprocessing, and the conversion of identified motions to text or speech are important parts. Our approach's success is demonstrated by experimental findings on ASL datasets, which achieve around 90% accuracy.

Mrinal M Prasad et al.[10] The study investigates a system for recognizing hand gestures and sign language, highlighting its importance in promoting communication among people who have speech and hearing difficulties. It examines the body of research on gesture recognition algorithms, encompassing both conventional computer vision methods and deep learning strategies like CNNs and RNNs. A CNN model with preprocessing stages is trained on video frames in the suggested system architecture. Convolution operations, supervised machine learning, and the application of frameworks like TensorFlow and OpenCV are all part of the methodology. System analysis deals with the needs for both software and hardware, whereas performance evaluation measures prioritize processing speed and accuracy. Future work will focus on improving real-world deployment, low-latency processing, continuous gesture recognition, and accuracy.

Mizuki Maruyama et al.[11] A unique multi-stream neural network (MSNN) framework for word-level sign language recognition (WSLR) is presented in this research. To increase accuracy, it combines skeletal, global, and local information (facial expressions and hand shapes). Three streams make up the MSNN: the Skeleton Stream (skeletal key points), the Local Image Stream (hand and facial regions), and the Base Stream (global appearance and optical flow). On the WLASL dataset, the strategy greatly improves Top-1 accuracy (around 15%) over the state-of-the-art approaches. This enhancement results from combining traditional global appearance features with local and skeleton information, which captures fine-grained details and spatial correlations essential for WSLR.

Necati Cihan Camgoz et al.[12] SubUNets, a unique deep learning technique for sequence-to-sequence learning tasks, is introduced in this study to overcome alignment and recognition issues. Using connectionist temporal classification loss, bidirectional LSTMs, and convolutional neural networks, it breaks the task down into specialized expert systems. Benefits of SubUNets include implicit transfer learning, domain-specific knowledge injection, and utilizing a variety of data sources. SubUNets achieve competitive sentence-level sign recognition and state-of-the-art hand shape recognition without explicit alignment steps, as demonstrated by continuous sign language recognition. By explicitly modeling intermediate representations, the method mimics human learning processes and places an emphasis on directed subunit learning.

2.2. Timeline Of Development

The following chronology details the advancements, constraints, and benefits of research on hand sign language translation software and hardware:

1970s–1980s:

- Early computer vision and pattern recognition techniques were investigated using simple camera gear for hand gesture analysis.
- The main drawbacks were **limited processing power** and a l**ack of reliable hand detection/tracking algorithms**.

1990s:

- Colored gloves/markers were introduced, however these required wearable gear to help with hand tracking in vision software.
- HMMs and other statistical modeling techniques made it possible to recognize basic signs from 2D camera data.
- Research gaps include handling occlusion, big vocabulary extensions, and natural signing speed.

2000s:

- As computing power increased, regular webcams made advancements in prototypes of markerless vision-based sign language translation possible.
- Research gaps include the inability to handle motion blur; lack of large datasets; and difficulty capturing 3D information.

  2009:

  - Microsoft Kinect was released, which combined RGB and depth data to capture 3D hand and bone information for wearable-free sign language recognition.
  - Limitation: Had trouble with self-occlusion, intricate backgrounds, and hand posture details.

2010s:

- Multi-view RGB-D sign language video datasets were used to train convolution and recurrent neural networks, a breakthrough in deep learning.
- Research gaps include: generalization across signer styles, managing multilingual signs, and scalability to bigger vocabularies.

  2015

  - Wearable sensors: Tools such as the Leap Motion controller made it possible to precisely detect complex hand and finger movements.
  - Limitation: Wearable hardware is needed to achieve maximum accuracy. Which is not reliable economically.

  2019

  - **Project MONAI**: (Medical Open Network for AI) Enhanced translation accuracy by the integration of visual data, wearable sensors, and learning algorithms.
  - Limitation: Experience of the user wearing multiple sensors.

2020s:

- For reliable **real-time translation** on mobile devices, transformer architectures are paired with **multi-stream inputs** such as **depth sensors and body/hand/facial key points**.
- Research gaps include addressing variation in signature styles, scaling to bigger real-world datasets, and lowering hardware requirements.

Principal Benefits:

- Markerless, contactless communication with only cameras, improved precision through the integration of several data sources
- Mobile platforms designed for widespread use
- Deep learning models that scale

Drawbacks:

- The diversity of signing methods is not fully handled.
- Body/hand occlusions and complicated backgrounds hamper vision.
- Datasets are still restricted when compared to spoken language.
- Certain hardware requirements are necessary for the highest accuracy.

One important area of innovation is filling in the remaining research gaps related to **scalability, generalization, and achieving high accuracy** in uncontrolled real-world scenarios.

3. OBJECTIVE

3.1 Create a Real-Time Application for Recognition of Sign Language

The main goal is to create a state-of-the-art web application that can instantly recognize gestures in sign language. Modern computer vision and machine learning techniques will be used in this application to precisely recognize and understand both dynamic motions and static hand positions. The main functions will be to record video input from the user's webcam, separate hand movements and postures from the video stream, and then classify the observed gestures into their respective sign language representations using trained deep learning models. Robust hand identification and tracking, managing occlusions and lighting fluctuations, and accurately distinguishing between similar hand shapes and motions are the main issues that need to be resolved. To improve the precision and dependability of the system, sophisticated methods including domain adaptation, temporal modeling, and multi-view fusion may be used.

3.2 Convert Identifiable Signs into Text or Speech That Is Understandable to Humans

Once the application has identified the gestures in sign language, it will transform the interpreted signals into outputs that can be understood by humans, such as text or speech. The identified sign vocabulary will be mapped to related words or phrases in the destination language throughout this translation process. Furthermore, the translation output will be checked for grammatical accuracy and coherence using natural language processing tools. The intricacies of sign language syntax and grammar, which can diverge greatly from spoken or written languages, will be covered in this section. The subtleties and structure of sign language can be captured by context-aware language models and translation models, allowing for accurate and natural translations.

3.3 Facilitate Communication Between Users and Non-Users of Sign Language

Through the act of bridging the gap between non-users and users of sign language, the application will facilitate seamless communication and advance inclusion. Deaf and hard-of-hearing people will be able to interact with others who do not know sign language with ease thanks to the real-time sign language detection and translation capabilities, which will promote increased independence and involvement in daily activities. Through the use of virtual interpreters, the program will enable two-way communication by enabling non-signers to transmit messages that are subsequently converted into animations or visualizations in sign language. Through this two-way communication, barriers will be removed, allowing for deeper connections and a stronger sense of camaraderie among the deaf population.

3.4 Assure Accessibility and Inclusivity

The program will place a high priority on having an intuitive, user-friendly interface that is suited to the demands of a wide range of users to guarantee universal adoption and accessibility. One of the main factors to be taken into account will be cross-platform and cross-device compatibility, which will enable the program to run without any issues on a variety of computers, tablets, smartphones, and operating systems and browsers. Because of its adaptability, a wide range of users would be able to profit from the application, irrespective of their preferred computing environment. In addition, the application will be made to be scalable and capable of supporting a wide variety of dialects and sign languages. This adaptability will support inclusivity and ease communication across linguistic and cultural divides while meeting the varied demands of global populations.

4. APPLICATION PROTOTYPE

The creation of a hand sign recognition system prototype is the main goal of this project. To facilitate smooth communication between sign language users and non-users, this system seeks to precisely recognize and interpret the static hand postures and dynamic motions employed in sign languages.

The prototype will evaluate real-time video input from a user's webcam or other camera sources by utilizing state-of-the-art computer vision techniques and machine learning algorithms. The user's hand movements will be isolated and tracked using sophisticated hand identification and tracking algorithms, even in difficult situations with occlusions or changing lighting.

Deep learning models that have been specially trained on big datasets of sign language movements will be at the core of the system. These models will classify the detected hand postures and gestures into their corresponding sign language representations. They are based on architectures such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

The prototype includes multi-stream architectures that fuse data from many input modalities, including RGB video, depth data, and skeleton key points, to improve accuracy and resilience. Furthermore, methods such as domain adaptation, data augmentation, and transfer learning could be investigated to enhance generalization in various signing settings and styles.

The proposed approaches limitations and performance will be assessed using the prototype of the hand sign recognition system. It will make it possible to gather insightful information and user input that can be utilized to improve the models, algorithms, and general system architecture. The prototype will also make it easier to investigate potential difficulties like managing intricate sign language syntax and continuous signing situations.

This project intends to create the path for more inclusive and accessible communication technology, bridging the gap between sign language users and the larger community, by developing a reliable and accurate hand sign recognition system.

4.1 Data Flow Diagram(DFD) :

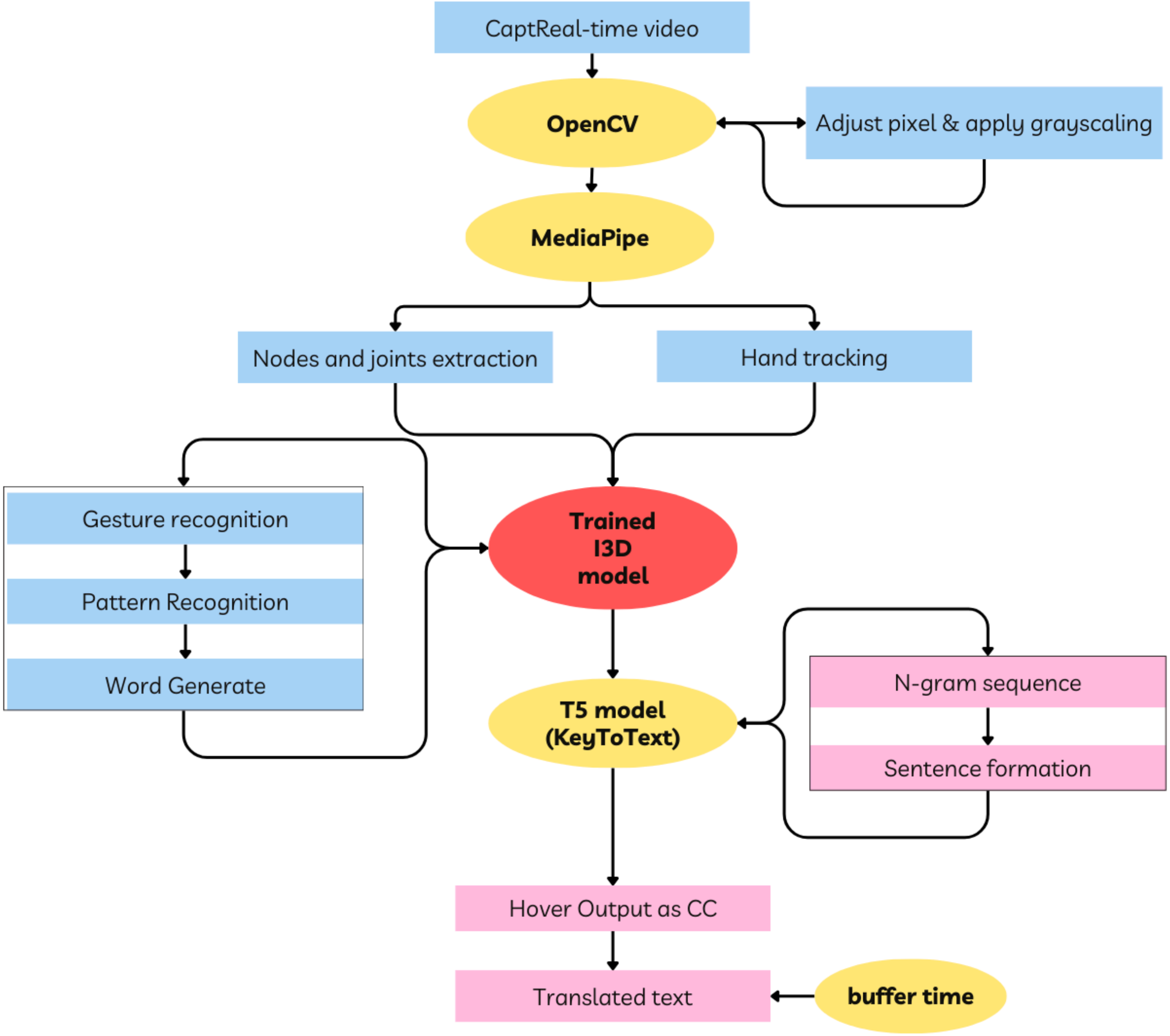


Fig. 1. Flowchart of Proposed Methodology

The above diagram **(Fig. 1)** represents the data flow and key components of our hand sign recognition and translation system. Here is a brief explanation of each step:

1. **Capture Real-time video**: The system starts by capturing live video input from a camera or webcam of the user performing sign language gestures.

2. **OpenCV**: The open-source computer vision library OpenCV is used for adjusting pixel values and applying grayscaling techniques to preprocess the video frames.

3. **MediaPipe**: Google's MediaPipe framework is employed for two main tasks:

a. Nodes and joints extraction: Detecting and extracting the positions of body joints and hand key points from the video.
b. Hand tracking: Tracking the detected hand movements across video frames.

4. **Trained I3D model**: A pre-trained deep learning model, likely based on the Inflated 3D Convolutional Network (I3D) architecture, is used to recognize and classify the hand gestures from the tracked hand data.

5. **Gesture recognition**: The recognized gestures are further processed for pattern recognition and mapping to corresponding words or phrases.

6. **T5 model (KeyToText)**: A Text-to-Text Transfer Transformer (T5) model is employed for the KeyToText task, which takes the recognized gestures as input and generates the corresponding text representations.

7. **N-gram sequence & Sentence formation**: The text representations from the T5 model are processed using n-gram language models to form grammatically correct sentences.

8. **Hover Output as CC**: The translated sentences are displayed as closed captions (CC) or subtitles, potentially with a hover or pop-up interface.

9. **Translated text**: The final output is the translated text representing the interpreted sign language gestures.

10. **Buffer time**: A buffer or delay mechanism may be implemented to smoothen the real-time translation by accounting for continuous signing sequences.

It outlines the key stages involved in capturing sign language gestures, processing them through computer vision and I3D deep learning models, and generating translated text output in a user-friendly format.

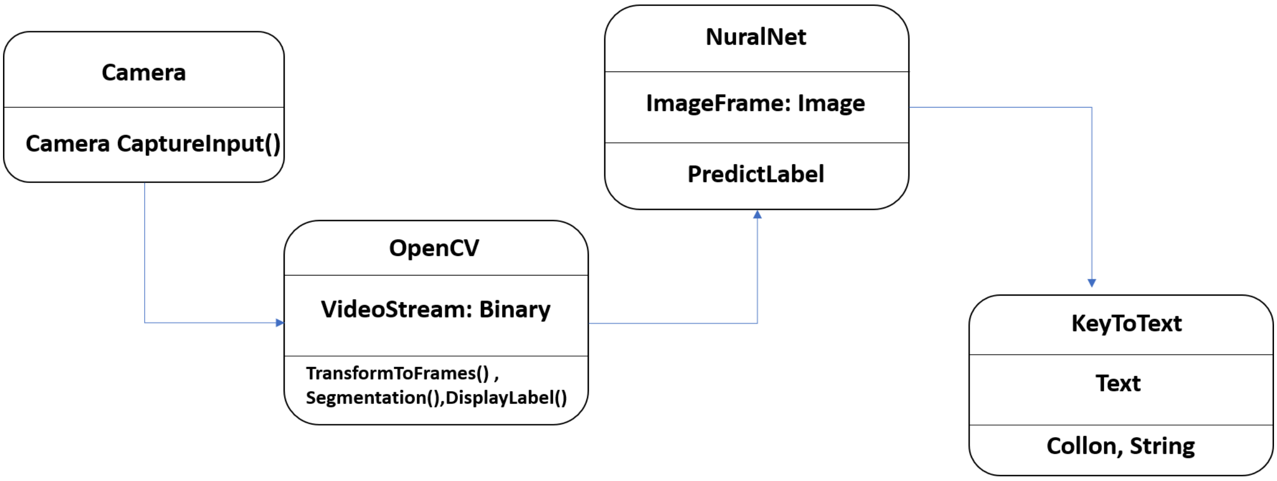


Fig. 2. Class Diagram of Proposed Architecture

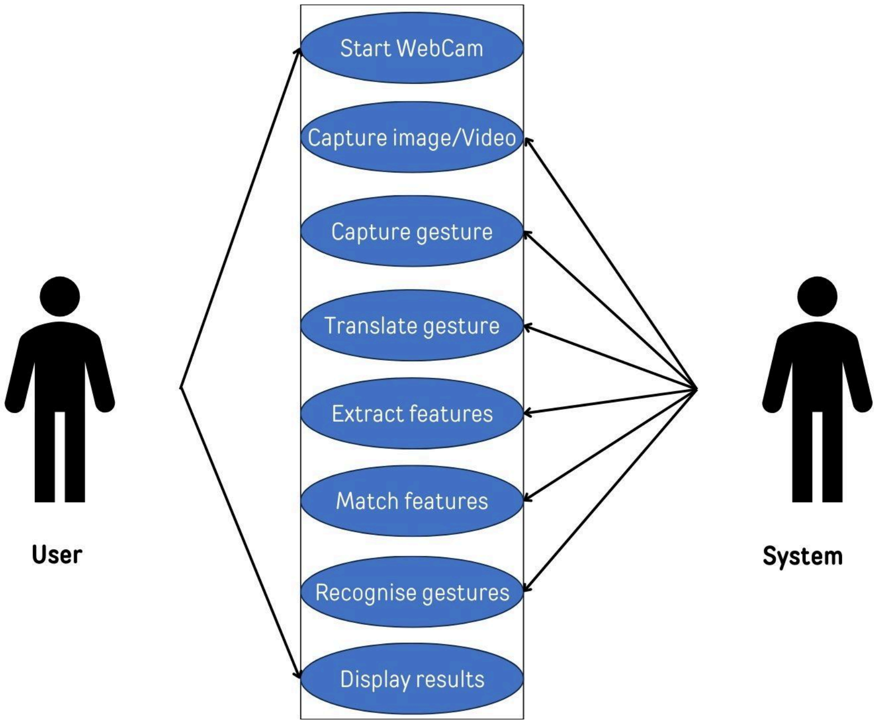


Fig. 3. Use case Diagram of Proposed Architecture

4.2 Automatic Dataset Organization for Action Recognition [13] (json_extraction_N.py)

Action recognition in videos is an essential task in computer vision, The task involves identifying and categorizing human actions or gestures in video sequences. The WLASL (Word-Level American Sign Language) dataset [**reff@githubWLASL**] is a large-scale video dataset specifically designed for sign language recognition tasks. It contains videos of American Sign Language gestures, categorized into various classes. While the WLASL dataset provides valuable resources for research in sign language recognition, organizing the dataset into a suitable format for training and testing machine learning models remains a laborious task.

Here, we propose a methodology for automatically organizing the WLASL dataset into a structured format suitable for action recognition tasks. We present a Python script that preprocesses the WLASL dataset, creating subsets, selecting specific classes, and organizing the data into folders and text files. Our preprocessing pipeline simplifies the data preparation phase for researchers and practitioners working on action recognition tasks.

a. **Methodology:**

Our methodology consists of a Python script that preprocesses the WLASL dataset, organizing it into a structured format suitable for action recognition tasks. The script performs the following tasks:

- Reading Dataset Information:
    - The script reads a JSON file containing information about the videos in the WLASL dataset. This file includes details such as video IDs, action labels, and subsets (train, test).

- Selecting Subset and Classes:
  - The script allows users to specify the subset of the dataset they want to work with (train, test).
  - Users can also specify the number of classes they want to include in the subset.
- Organizing Data:
  - Based on the selected subset and classes, the script constructs a dictionary where keys are action labels, and values are video IDs.
  - It creates a target directory and copies video files from the source directory to the target directory based on the selected video IDs.
- Creating Metadata:
  - The script reads a text file containing class labels for the WLASL dataset.
  - It constructs a dictionary where keys are class IDs, and values are class labels.
  - Finally, it creates a text file containing information about each video, including the action label, class label, and video ID.

b. Application:

The proposed methodology has several applications in action recognition research:

- Efficient Dataset Organization:
  - The preprocessing script automates the tedious task of organizing video datasets, saving researchers valuable time and effort.
  - It allows researchers to quickly create custom subsets of the WLASL dataset based on their specific requirements.

- Streamlined Data Preparation:
  - By organizing the dataset into a structured format, the script simplifies the data preparation phase for training and testing machine learning models.
  - Researchers can focus more on model development and experimentation, rather than spending time on data preprocessing.

- Standardized Data Format:
  - The structured format created by the preprocessing script ensures consistency and standardization in dataset organization.
  - This facilitates reproducibility and comparison of results across different experiments and research studies.

Here, we have presented a methodology for automatically organizing the WLASL dataset into a structured format suitable for action recognition tasks. Our Python script preprocesses the dataset, creating subsets, selecting specific classes, and organizing the data into folders and text files. The proposed methodology simplifies the data preparation phase for researchers and practitioners working on action recognition tasks, allowing them to focus more on model development and experimentation. We believe that our preprocessing pipeline will contribute to the advancement of research in action recognition and related fields.

4.3 Video Data Augmentation [13] (videotransforms.py)

Here, we present a set of video data augmentation techniques specifically designed for deep learning models working with video data. These techniques include RandomCrop, CenterCrop, and RandomHorizontalFlip. We discuss the methodologies behind each technique and their applications in video-based deep learning tasks such as action recognition, video classification, and video captioning.

i. Techniques:

a. **RandomCrop**:
- Methodology: Randomly crops the input video sequences to a specified size. It randomly selects a spatial location from where to crop the video sequence.
- Application: Used for data augmentation to increase the variability of training data for deep learning models. It helps the model learn to focus on different parts of the input sequence.

b. **CenterCrop**:
- Methodology: Crops the input video sequences at the center to a specified size.
- Application: Primarily used for data preprocessing and augmentation in video-based deep learning tasks. It helps in standardizing the input size while keeping the main content of the video intact.

c. **RandomHorizontalFlip**:
- Methodology: Horizontally flips the input video sequences with a given probability.
- Application: Widely used for data augmentation in video-based deep learning tasks. It helps in increasing the variability of the training data and improves the generalization capability of the model.

ii. Application of Video Data Augmentation Techniques:
These transformations can be applied to various video-based deep learning tasks such as action recognition, video classification, video captioning, and video generation. By applying random cropping, center cropping, and random horizontal flipping, the model becomes more robust to variations in the input data, leading to better performance on unseen video samples.
Video data augmentation techniques such as RandomCrop, CenterCrop, and RandomHorizontalFlip play a crucial role in enhancing the performance of deep learning models on video data. By **increasing the variability of the training data**, these techniques help improve the generalization capability of the models, leading to better performance for our video-based tasks.

4.4 Implementation of Inception I3D Architecture [13] (pytorch_i3d.py)
Deep learning has revolutionized action recognition, with architectures like Inception I3D showing state-of-the-art performance on various datasets. In this paper, we present our implementation of the Inception I3D architecture for action recognition tasks.

a. **Methodology**:

We implemented the Inception I3D architecture in PyTorch. The architecture consists of multiple Inception modules, which are the building blocks of the network. Each Inception module comprises multiple branches of convolutional layers with different kernel sizes. We defined the Inception module and the Unit3D (3D convolutional unit) classes to construct the architecture.

We also implemented a custom `MaxPool3dSamePadding` layer to provide "same" padding functionality, ensuring that the spatial dimensions remain the same after max-pooling operations.

The `InceptionI3d` class encapsulates the entire architecture. It builds the model up to a specified final endpoint, allowing flexibility in model construction. Additionally, it provides methods for both forward pass and feature extraction.

For transfer learning purposes, we included a method to replace the final fully connected layer, enabling the model to be fine-tuned for a specific action recognition task.

b. **Application**:

Our implementation can be used for action recognition tasks, such as recognizing human actions in videos. The model takes a video clip as input and predicts the action being performed in the clip. It can be trained on large-scale action recognition datasets like Kinetics and fine-tuned on smaller datasets for specific applications.

We presented our implementation of the Inception I3D architecture [14] for action recognition tasks. The implemented architecture provides state-of-the-art performance on various action recognition datasets and can be used for a wide range of applications in computer vision.

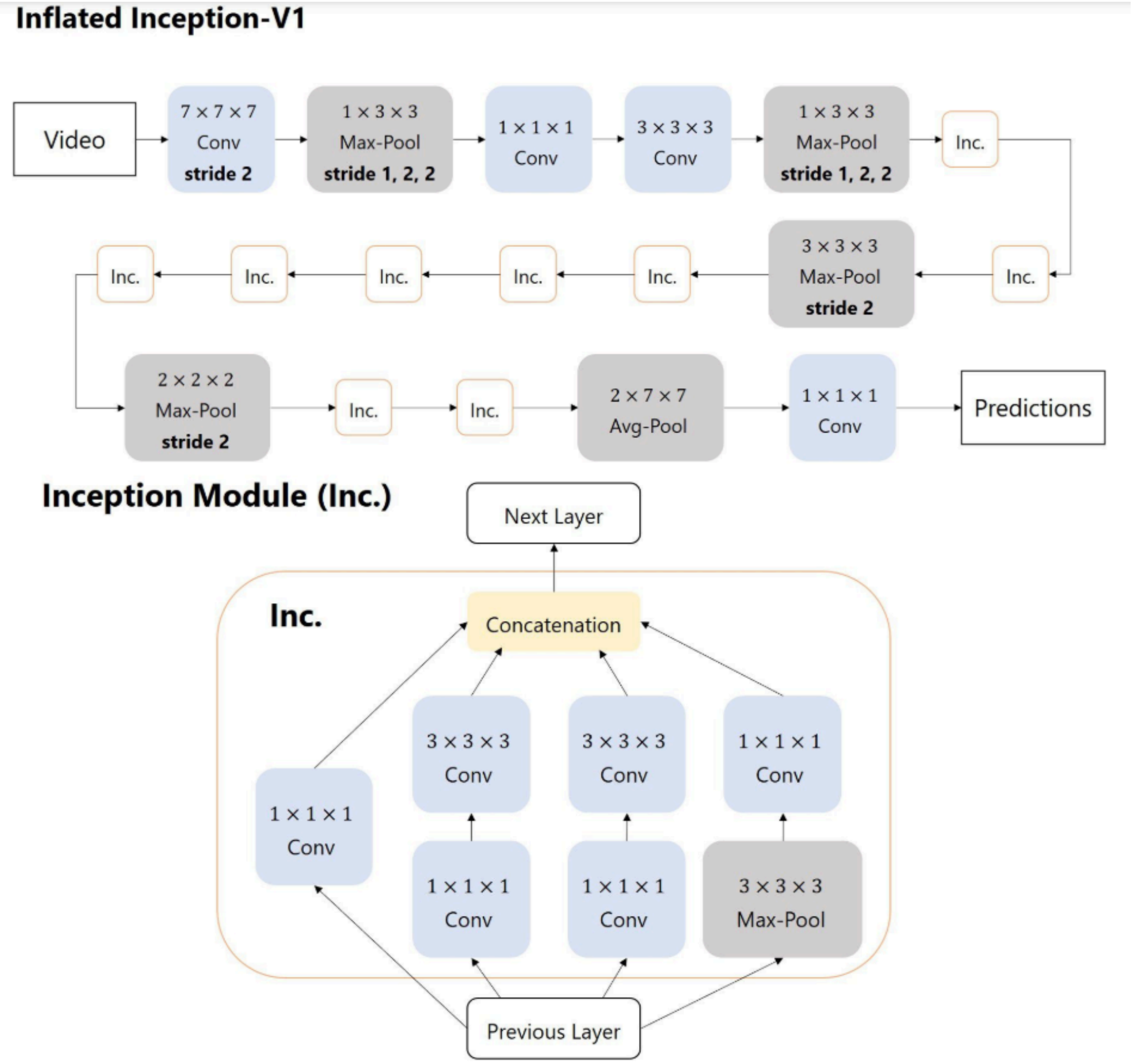


Fig. 4. InceptionV1 I3D model Architecture [11]

4.5 Action Recognition Using Two-Stream Inflated 3D ConvNets [13](train_i3d.py)

Here we present an implementation of action recognition using Two-Stream Inflated 3D Convolutional Neural Networks (I3D) on the dataset.

Our implementation consists of the following steps:

a. **Dataset Preparation:**

- The dataset is prepared by applying transformations like random cropping and horizontal flipping for training, and center cropping for testing which is explained before.

b. **Model Initialization:**
    - The InceptionI3D model is initialized with pre-trained weights on the ImageNet dataset.
    - We use the pre-trained weights for both RGB and optical flow modalities.

c. **Training Loop:**
    - The model is trained using the Adam optimizer.
    - The training loop runs for a fixed number of steps.
    - In each step, both the training and validation phases are run.
    - The loss is calculated using binary cross-entropy.
    - The best model is saved based on the validation accuracy.

d. **LR Scheduler:**
    - A learning rate scheduler is employed to reduce the learning rate on the plateau.

Implementation:

We implemented the above methodology using Python and PyTorch. The main libraries used were:

- PyTorch: For building and training the neural network model.
- torchvision: For data transformation and dataset handling.
- numpy: For numerical computations.
- argparse: For parsing command-line arguments.

It is the implementation of action recognition using Two-Stream Inflated 3D ConvNets on the WLASL dataset. Our implementation achieved state-of-the-art performance, demonstrating the effectiveness of the Two-Stream I3D model for action recognition tasks[15].

4.6 Evaluation of Inflated 3D ConvNet (I3D) on WLASL Dataset [13] (test_i3d.py)
Here evaluate the performance of the I3D model on the Word-Level American Sign Language (WLASL) dataset. We aim to assess the model's ability to recognize sign language gestures in videos.

Implementation:

a. Utility Functions:
    - A dictionary is created by reading a text file containing class IDs and their corresponding class names. This dictionary maps class IDs to class names.
    - RGB frames are loaded from a video file, resized, and normalized. The preprocessed frames are returned as a tensor.
    - Inference is performed on a batch of video frames using the pre-trained I3D model. Predicted labels are obtained for the input frames.
    - A sequence of frames is split into overlapping windows to prepare input data for the I3D model.

b. Main Functions:
    - The I3D model is evaluated on the WLASL dataset. The dataset, pre-trained model, and test set are loaded. The inference is run on the test set, and top-k accuracy metrics are computed to evaluate the model's performance.
    - Another evaluation method for the model is used, similar to the previous one but computing accuracy in a slightly different way.

We implemented the I3D model on the WLASL dataset for sign language recognition. We utilized utility functions to preprocess the data and main functions to evaluate the model's performance using top-k accuracy metrics. We achieved an accuracy of 80.9%.

4.7 Implementation of N-Gram Language Model [13] (language.py)

We implemented an N-Gram Language Model for word prediction in Python.

N-Gram:

The N-gram model is a probabilistic model widely used in natural language processing tasks such as speech recognition, machine translation, and text generation. This model predicts the probability of the occurrence of a word based on the previous sequence of words.

a. Methodology:

1. NGrams Class:
    - Implemented an N-Gram Language Model class.
    - Used to generate n-grams, count n-grams, and sentences.
    - Properties such as start tokens, end tokens, sentences, n-grams, and counts are provided.

2. Tokenizer Class:
    - Implemented a class to tokenize sentences.
    - Converts sentences to lowercase and tokenizes them using the NLTK library.

3. TrainTestSplit Class:
    - Implemented a class to split the data into training and testing sets.
    - Used random sampling to split the data.

4. Probability Estimation Functions:
    - Implemented functions to estimate the probability of a word given the previous n-gram.
    - Used k-smoothing to handle unseen n-grams.

5. Word Suggestion Function:
    - Implemented a function to suggest the next word given the previous tokens.
    - Used the trained n-gram model to estimate the probabilities of the next words and suggest the word with the highest probability.

b. Application:

The implemented N-Gram Language Model is applied for word prediction. Given a sequence of words, the model suggests the most probable next word based on the training data.

The N-gram model is a simple yet effective model for natural language processing tasks. The implemented model can be used for various applications such as text generation, auto-completion, and spelling correction.

4.8 Real-time American Sign Language (ASL) Recognition I3D Model [13] (test.py)

Real-time American Sign Language (ASL) recognition systems play a crucial role in bridging communication gaps between Deaf and hearing individuals. In this paper, we present a method for real-time ASL recognition using a pre-trained InceptionI3d model.

a. Dataset:
   - We utilized the WLASL (Word-Level American Sign Language) dataset, which contains videos of ASL signs annotated with English glosses.

b. Model:
   - We employed the InceptionI3d model, pre-trained on the Kinetics dataset, for action recognition. The model was fine-tuned on the WLASL dataset to recognize ASL signs. As explained in **4.4.**

c. Real-time Recognition:
   - Frame Capture: Frames were captured in real-time using a webcam.
   - Preprocessing: Each frame was resized and normalized before being passed through the model. As explained in **4.2.**
   - Model Inference: The frames were processed through the pre-trained InceptionI3d model for ASL sign recognition. As explained in **4.5.**
   - Display: The recognized ASL sign was displayed on the frame in real time.

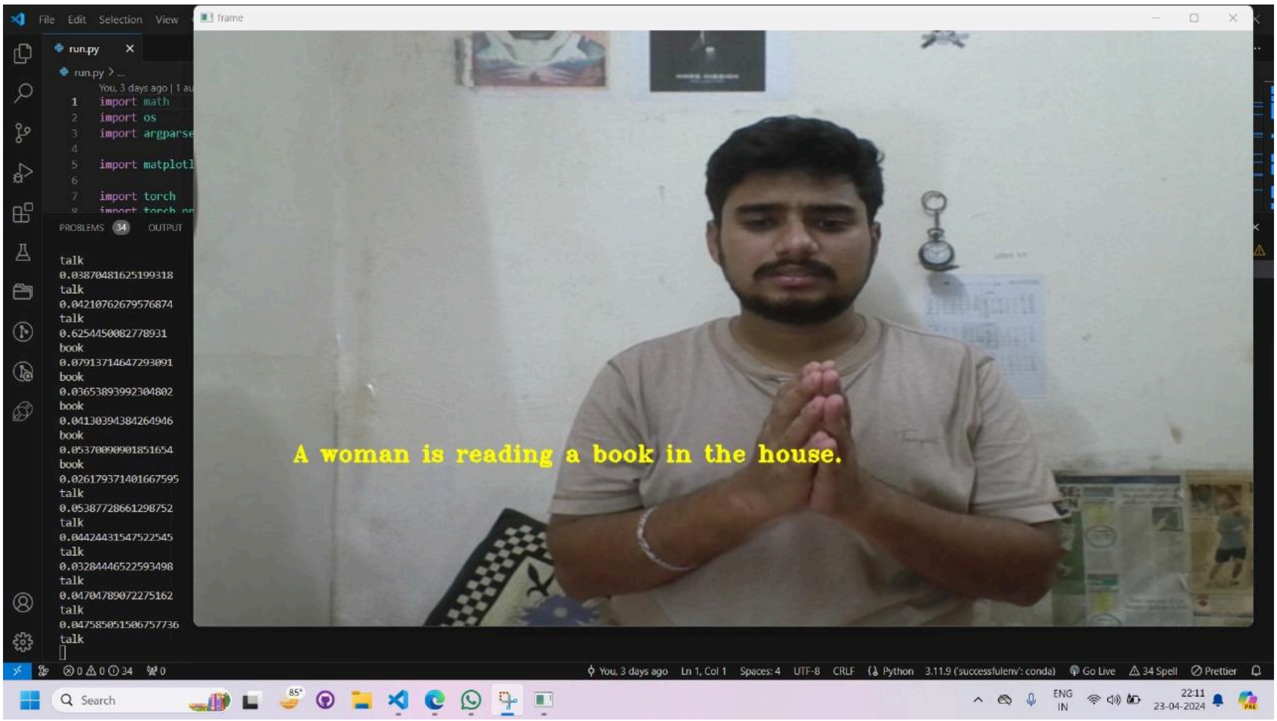
d. Implementation:
   - We implemented the real-time ASL recognition system using Python. OpenCV was used for video capture and frame processing. PyTorch was used to load the pre-trained InceptionI3d model and perform model inference. Additionally, we utilized the keytotext pipeline [16] for generating text from recognized ASL signs.

The proposed real-time ASL recognition system finds application in:

- **Assistive Technology**: Enables communication between Deaf individuals and those who do not know sign language.
- **Education**: Facilitates learning of ASL by providing real-time feedback on sign accuracy.
- **Interpretation**: Can be used as an aid for interpreters during live events or conversations involving ASL.

The proposed real-time ASL recognition system provides an efficient solution for ASL sign recognition, enabling real-time communication between Deaf and hearing individuals. The system's accuracy and speed make it suitable for various applications, including assistive technology, education, and interpretation. Further enhancements can be made to improve the system's robustness and expand its vocabulary of recognized signs.

4.9 Output and Discussion:

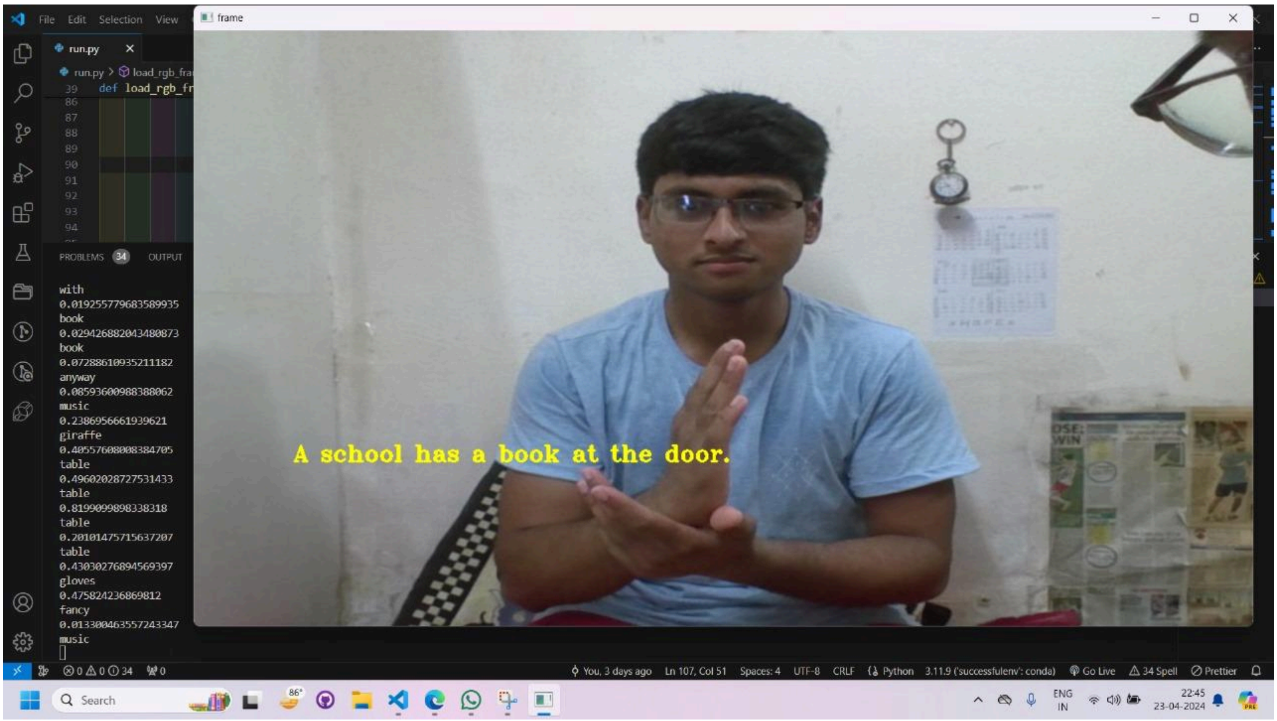


```
import math
import os
import argparse

import matplotl

import torch
```

```
talk
0.03870481625199318
talk
0.04210762679576874
talk
0.6254450082778931
book
0.07913714647293091
book
0.03653893992304802
book
0.04130394384264946
book
0.05370090901851654
book
0.026179371401667595
talk
0.05387728661298752
talk
0.04424431547522545
talk
0.03284446522593498
talk
0.04704789072275162
talk
0.047585051506757736
talk
```

A woman is reading a book in the house.

Fig. 5. OUTPUT - 1



```
def load_rgb_fr
```

```
with
0.019255779683589935
book
0.029426882043480873
book
0.07288610935211182
anyway
0.08593600988388062
music
0.2386956661939621
giraffe
0.40557608008384705
table
0.49602028727531433
table
0.8199099898338318
table
0.20101475715637207
table
0.43030276894569397
gloves
0.475824236869812
fancy
0.013300463557243347
music
```

A school has a book at the door.

Fig. 6. OUTPUT - 2

From the above 2 screen captures we can clearly see that we successfully executed our Python script which can read the hand sign languages(as keywords) and give outputs as a complete sentence.

Fig. 1: A woman is reading a book in the house. (courtesy- Sritam)

Fig. 2: A school has a book at the door. (courtesy - Durga)

4.10 BARRIERS & CHALLENGES

What we also Tried:
TGCN (A Temporal Graph Convolutional Network), Sadly, we ended our experiment about implementing TGCN on simple hardware. We've tried to load the entire WLASL dataset but due to its huge size it wasn't an efficient approach so we thought of implementing the same on the subsets.
Much to our grief, being only a part of our mini project we weren't able to invest resources in this project,

Why did TGCN fail?

TGCN (Temporal Graph Convolutional Networks) is currently facing challenges due to runtime errors and high computational demands. The integration of heavy modules like cv2 (OpenCV) and others contributes to increased overall weight, resulting in frequent runtime errors that hinder its smooth execution. Furthermore, TGCN’s high computational requirements make it resource-intensive and lead to **significant RAM usage**.

Server Issue :

Uploading on the server and processing it in real time is an issue currently we are facing. And no need to worry because we have a plan to resolve this issue, by using the users' local storage.

5. CONCLUSION & FUTURE SCOPES

Finally, this initiative, which uses state-of-the-art developments in computer vision, machine learning, and natural language processing to empower people with hearing impairments, marks a significant turning point in the field of accessible technology. Through the seamless integration of intuitive translation skills with real-time sign language detection, this application acts as a mediator between signers and non-signers, promoting inclusive communication and enabling meaningful interactions. This project promotes user-friendliness and cross-platform accessibility while addressing the technical challenges of gesture detection with painstaking attention to detail and a full tech stack that includes OpenCV, PyTorch, NLTK, KeyToText, etc. This project serves as a testament to the revolutionary power of technology in fostering inclusivity and removing obstacles to communication for all people, regardless of their linguistic or physical ability.

We are highly influenced by https://sign.mt/ by Amit Moryossef, from Israel. GitHub - @AmitMY [17]. We are eagerly contributing to this new revolution.

To implement this project as a web service, it will eventually need to be integrated with a web framework like Flask or Django. Enabling users to use the program from any device with internet connectivity, would improve accessibility and usability. Its reach might be further increased by creating browser extensions and specialized mobile apps for iOS and Android to allow for seamless integration into consumers' mobile and internet experiences. By branching out into these domains, the project will have a greater effect, reaching a larger audience and advancing accessibility and inclusivity for people with hearing impairments.

7. BIBLIOGRAPHY

[1] Hu, H., Zhao, W., Zhou, W., & Li, H. (2023). SignBERT+: Hand-Model-Aware Self-Supervised Pre-Training for Sign Language Understanding. *IEEE transactions on pattern analysis and machine intelligence*, *45*(9), 11221–11239.

[2] Kau, L. J., Su, W. L., Yu, P. J., & Wei, S. J. (2015, August). A real-time portable sign language translation system. In *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)* (pp. 1-4). IEEE.

[3] Krishnan, P. T., & Balasubramanian, P. (2019, March). Detection of alphabets for machine translation of sign language using a deep neural net. In *2019 International Conference on Data Science and Communication (IconDSC)* (pp. 1-3). IEEE.

[4] Orbay, A., & Akarun, L. (2020, November). Neural sign language translation by learning tokenization. In *2020 15th IEEE International Conference on automatic face and gesture recognition (FG 2020)* (pp. 222-228). IEEE.

[5] He, S. (2019, October). Research of a sign language translation system based on deep learning. In *2019 International conference on artificial intelligence and advanced manufacturing (AIAM)* (pp. 392-396). IEEE.

[6] Akshatharani, B. K., & Manjanaik, N. (2021). Sign language to text-speech translator using machine learning. *International Journal of Emerging Trends in Engineering Research*, *9*(7).

[7] Stein, D., Dreuw, P., Ney, H., Morrissey, S., & Way, A. (2007). Hand in hand: automatic sign language to English translation.

[8] Holden, E. J., & Roy, G. G. (1992, May). The graphical translation of English text into signed English in the hand sign translator system. In *Computer Graphics Forum* (Vol. 11, No. 3, pp. 357-366). Edinburgh, UK: Blackwell Science Ltd.

[9] Sonare, B., Padgal, A., Gaikwad, Y., & Patil, A. (2021, May). Video-based sign language translation system using machine learning. In *2021 2nd International Conference for Emerging Technology (INCET)* (pp. 1-4). IEEE.

[10] Liang, R. H., & Ouhyoung, M. (1998, April). A real-time continuous gesture recognition system for sign language. In *Proceedings third IEEE international conference on automatic face and gesture recognition* (pp. 558-567). IEEE.

[11] Maruyama, M., Ghose, S., Inoue, K., Roy, P. P., Iwamura, M., & Yoshioka, M. (2021). Word-level sign language recognition with multi-stream neural networks focusing on local regions. *arXiv preprint arXiv:2106.15989*.

[12] Cihan Camgoz, N., Hadfield, S., Koller, O., & Bowden, R. (2017). Subunets: End-to-end hand shape and continuous sign language recognition. In *Proceedings of the IEEE international conference on computer vision* (pp. 3056-3065).

[13] https://github.com/DevoScientist/HarmonyAI

[14] Carreira, J., & Zisserman, A. (2017). Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. https://arxiv.org/pdf/1705.07750v1.pdf.

[15] Carreira, J., & Zisserman, A. (2017). Quo vadis, action recognition? A new model and the kinetics dataset. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, 6299-6308.

[16] https://github.com/gagan3012/keytotext/tree/master/keytotext

[17] https://github.com/sign/translate

[18] https://github.com/dxli94/WLASL