Middlesex University London

CST4090 Individual Project

Real-Time Hand Gesture Translator using Machine Learning

Student Name: ANISH MOHANAN

Student ID: M00879683

Supervisor Name: GIACOMO NALLI

October 2023

This thesis is submitted in part of fulfilling the requirements for the MSc.
Data Science at Middlesex University, 2023.

ACKNOWLEDGEMENTS

I want to express my deep and sincere gratitude to my supervisor, Lect. Giacomo Nalli's continuous support for my master's study final project research paper. His enthusiasm, patience, immense knowledge, and guidance have supported me in all the processes of experiments and writing of this final project. I could not have imagined it would have been possible to have a better supervisor and mentor for this dissertation.

Apart from my supervisor, I thank Prof. Gao XiaoHong for her immediate responses to difficult questions and assistance with insightful comments and encouragement, especially during the COVID-19 pandemic.

I am incredibly grateful to my parents and siblings for their love, prayers, and care about always motivating me. I am also thankful to my boyfriend for his love, understanding and continuing genuine support to complete this final project.

I thank all the staff of Middlesex University's Unihub, Welfare Department, Progression, and Support team for their kindness in resolving all my problems and questions.

Finally, my gratitude to all the people who have helped me, directly or indirectly, to finish this dissertation successfully.

Table of Contents

ACKNOWLEDGEMENTS ii

ABSTRACT v

1. Introduction 1

1.1 Background 1

1.2 Scope and Objectives 2

2. Literature Review 3

2.1 Existing methods other than Sign Languages. 3

2.2 Sign Language 3

2.3 Sign Language Recognition 4

2.4 Challenges in Sign Language Recognition 4

2.5 Machine Learning and Deep Learning 5

2.6 American Sign Language 6

2.7 Previous approaches in Sign Language Recognition 6

2.8 Recent Advances in Deep Learning for Gesture Recognition 8

2.9 Research dedicated to the recognition of American Sign Language(ASL). 9

2.10 Comparing CNN and Random Forest Models for Gesture Recognition: Discussion and Analysis 10

2.11 Discussion on Dataset Preparation 10

2.12 Evaluation Metrics 10

3. Methodology 11

3.1 System Design 11

3.2 Data Collection 13

3.3 Data Preprocessing 15

3.4 Model Selection 16

3.5 Training Model 16

3.5.1 Random Forest 16

3.5.2 Convolutional Neural Network (CNN) training. 18

3.6 Development and Deployment of the Application 19

3.7 Summary 20

4. Results and Evaluation 20

4.1 Experimental Setup and Data Splitting 20

4.2 Model Performance Metrics 20

4.2.1 Random Forest 20

4.2.2 Convolutional Neural Network Model 21
4.3 Application User interface (UI) results 22
4.4 Performance Metrics and Model Evaluation 23
4.4.1 Random Forest Model 23
4.4.2 CNN Model 25
4.5 Real-time Recognition Performance 28
4.5.1 Experiment 28
4.5.2 Throughput Analysis 28
4.5.3 Process Rate Analysis 28
4.5.4 Comparative Analysis 28
4.5.5 Limitations in Evaluation 29
4.6 User Interface and User Experience (UX) Assessment 29
4.7 Comparison of different Models 29
5. Discussion and limitations 30
5.1 Discussion 30
5.2 Limitations 31
6. User Guide 31
6.1 Visual Studio Code 31
6.2 Jupiter Notebook 32
7. Conclusion 33
7.1 Summary 33
7.2 Evaluation and Project Management 33
7.3 Impact and Innovation in the Project 34
7.4 Future Scope 35
8. REFERENCES: 37

ABSTRACT

More than 72 million individuals worldwide who are either deaf or hard of hearing rely on over 300 distinct sign languages for their means of communication. These statistics underscore the pressing need to facilitate communication between these individuals and those who do not possess knowledge of sign languages by automating the translation of sign language into text. However, the diversity of sign languages poses a significant challenge in achieving this automation.

The project titled "Real-Time Hand Gesture Translator using Machine Learning" seeks to develop a system that can promptly recognise and translate gestures from American Sign Language (ASL), the chosen sign language for this endeavour, into text in real time. This system employs machine learning algorithms and computer vision techniques to detect hand movements and identify specific hand gestures. To accomplish this, a dataset is constructed using frames extracted from live video feeds captured by a webcam. This dataset encompasses the 26 alphabets of American Sign Language, as well as a custom gesture denoting "space." Deep learning techniques are employed in the subsequent stages of the process.

The Mediapipe library is utilised to extract features from the dataset, which are then harnessed to train machine-learning models, specifically classification models. These trained models are subsequently employed to predict the detected hand gesture in real time. The proposed system incorporates three machine learning models: the Convolutional Neural Network (CNN) classifier and the Random Forest classifier. Notably, the Random Forest model attains the highest accuracy, achieving a training accuracy of 99.42% and a validation accuracy of 98.44% among all models utilised in this project.

At the user interface level, a user-friendly interface streamlines communication between sign language users and those unfamiliar with sign languages. This interface empowers users to select the model of their choice for translation and view the translated text within the interface, thus facilitating more effortless communication.

1. Introduction

In the modern world, good communication with others is essential. Highly skilled people sometimes need help to get a job because they cannot convey what they intend to make clear. People are not able to express their love or affection to others. This problem even happens to people who don't have any disability in communication. Now, thinking about people who are deaf or hard of hearing, they struggle to communicate with others who are not disabled. Disabled people use sign language to communicate with each other. When speaking between ordinary and disabled people, communication is difficult if the typical person doesn't know sign language.

1.1 Background

There are around 300 sign languages in the world [1]. A regular person will struggle to learn all these languages. Regarding the spoken language, British political economist John Bowring claimed that he knew 200 languages and could speak 100 [2]. Some others can speak ten languages, twenty languages, and more. These facts indicate the complexity of learning and using sign languages when a lot is available.

This issue is significant because efficient communication is essential to human connection. People with disabilities need to live among everyone quickly by communicating with others who do not have a disability. The others must understand what the dumb and deaf people are trying to communicate. In the present world, everyone is too busy, and no one wants to waste time. Time is money in this era. No one likes to waste time understanding what disabled people are trying to communicate. So, making a Real-Time Hand Gesture Translator to understand what disabled people are trying to convey is worth doing. Most disabled people are good at lip reading, so this will not affect their ability to understand what other people are saying.

A platform is needed to make communication between disabled people and others much more accessible. This platform will convert sign language into text and improve communication. The proposed system focused on this purpose. The American Sign Language (ASL) is used for this system. The American Sign Language alphabet gestures will be detected and transformed into text using computer vision and machine learning technologies. The proposed system intends to continue by changing gestures into text form; it plans to convert them into meaningful words, sentences, and audible formats.

Everyone can benefit from this proposed system because of the features provided. People who don't know sign language can now understand what sign is shown and hear the words converted from gesture to text. For now, the proposed system is focusing on ASL, and this system can be used for other sign languages as well. The system can run by training and testing using other sign language gestures. This makes the system more adaptable and can be used widely.

1.2 Scope and Objectives

This project aims to develop an efficient tool to detect hand gestures based on American sign language and to translate the detected gesture into text.

The goals of the study are the following:

i. Develop and train models for a Convolutional Neural Network (CNN) and Random Forest to correctly identify hand gestures based on American Sign language motions in live video feeds.
ii. Translate the identified hand gestures into the corresponding text.
iii. Create a user interface that prioritises usability and fluid movement to enable real-time video capture, precise hand gesture recognition, and the production of translated text output.
iv. Test the hand gesture translator identification tool rigorously using a variety of hand gestures based on ASL and a large sample size of people to determine its effectiveness.
v. Conduct a thorough examination to evaluate the application's precision, quickness, and user-friendliness.
vi. Discuss and look into future upgrades.

Collectively, the objectives outlined above will steer the project's trajectory as it endeavours to create a tool for recognising hand gestures rooted in American Sign Language. This tool aims to enhance interaction and foster inclusivity for individuals who rely on ASL for communication through the interpretation of hand signs.

2. Literature Review

2.1 Existing methods other than Sign Languages.

AAC (alternative and augmentative communication) aids in helping impaired people communicate. AAC devices can be custom-made or run specialised software on a conventional computer, tablet, or smartphone. These systems will speak to the chosen symbols. AAC tools are frequently used with other non-verbal communication techniques, such as facial expressions, images, and signs [3].
VocaliD, a Boston-based business, produces personalised digital voices for those unable to speak. The internet browser-based human voice bank from VocaliD is designed to resemble a video game in specific ways. When a text is typed, a cheery cartoon mouth next to the line of text will read it aloud [3].
Vocal output communication aids are specially designed communication devices that are widely used (VOCAs). Pre-recorded voices and synthetic speeches are in the speech output. Pre-recorded communications are made by speaking to a communication aid and recording the results. VOCAs are operated by [3]:

1. Pressing buttons or touchscreen
2. Operating switches
3. Typing text

The disadvantages of these solutions are that they are time-consuming and inefficient in many ways. The hard-hearing people know how their sound is, and it will be hard for them to hear the artificial sound instead of their original voice. Typing text to produce voice will consume time. Pressing buttons and scrolling through the touch screen will also take time. If one disabled person wants to use a specific word not recorded on the device, it will affect communication. If disabled people are not rich enough to buy the device or cannot record the voice at the correct time, then these solutions are not good enough.

Hearing aids are to help hard-hearing people communicate, but they cannot be used to help deaf people.

2.2 Sign Language

The inability of the existing methods to make communication much easier gives sign language recognition very much importance. Sign language recognition is essential for deaf people because it eliminates the communication gap between others by translating sign gestures into text [4]. This translation is helpful for others who don't know sign language. Sign language is a visual communication that involves hand gestures, body movements, and facial expressions. Deaf and mute individuals use it as their primary means of conversation. However, sign language can also be used by people with autism, apraxia of speech, cerebral palsy, Down syndrome, and individuals who can hear but cannot speak. Different countries have their unique sign languages, such as American Sign Language (ASL), British Sign Language (BSL), Australian Sign Language (AUSLAN), and Indian Sign Language (ISL) [5]. Sign language recognition is a popular area of research in computer vision, with deep learning models used to recognise and classify sign language words or phrases [6]. Sign language recognition systems aim to translate sign language into text or speech, facilitating communication between deaf and hearing individuals [7].

2.3 Sign Language Recognition

According to “Sign Language Recognition” by Nandina Anudeep, sign language recognition is a challenging task that has been under investigation for many years. It is a powerful tool for people with hearing and speech impediments to communicate their feelings and ideas to the world. Sign language recognition makes integrating sign language users and others smoother and less complicated. Background and lighting conditions affect hand tracking, and thus, it makes the detection of sign language difficult. For the study, a web camera is used as a video sensor, based on which hand and body action can be easily tracked more accurately [8].

Various methods have been proposed for sign language recognition, including web cameras for hand and body action tracking by Nandina Anudeep [8]. According to “Sign Language Recognition” by Tulay Karayilan, a backpropagation neural network algorithm for recognising sign language letters based on American sign language can be used. The neural network used for the study used features extracted from images as input, and it trained using a back-propagation algorithm to recognise which letter was the given letter with an accuracy score of 70% and 80% for the two proposed classifiers[9]. According to “Sign Language Recognition System and Method”, a depth-sensing camera for capturing and analysing gestures in three dimensions can be used for sign language recognition. This camera captures an image of a gesture of a signer. It gathers data about distances between several points on the signer and the depth-sensing camera, building a three-dimensional (3D) model of the gesture, comparing the 3D model of the gesture with several 3D models of different gestures to find out the representations of the 3D model of the gesture, and displaying or vocalising the representations of the 3D model of the gesture [10]. Sign language recognition involves extracting features from sign language data, classifying manual and non-manual aspects of sign language, and combining classification results into full sign language recognition [11]. Recent research has focused on isolated word recognition, continuous word translation, true signer independence, and effectively combining different sign modalities [12].

2.4 Challenges in Sign Language Recognition

Sign language recognition faces several challenges. One challenge is the difficulty in detecting hand gestures accurately due to factors like background and lighting conditions, which can affect hand tracking and make recognition difficult. Sign language involves hand gestures, head or body movements and facial expressions, making recognising complex. Acquiring images for training and testing can be a challenge. Capturing precise and clear images of hand signs can be difficult. Feature extraction is another challenge in sign language recognition. For classification and recognition, extracting relevant features from the segmented hand gestures is essential and complex [13].
Another challenge is the lack of sufficient sign language-specific data, as sign languages are under-resourced, making it challenging to model hand movements effectively. On top of the mentioned challenge, Nandina Anudeep states that Hand tracking and sign language recognition can become extremely difficult because of sophisticated background and light [8].

Additionally, the complexity of sign language and the variations in sign movements can lead to confusion and misinterpretation, especially for those unfamiliar with the sign language. This is even the case for those who know sign languages. The number of sign languages and variations makes recognition much more difficult. These variations challenge deep learning and convolutional neural networks in sign language recognition[14][15]. Furthermore, if we come to sign language recognition by humans, the limited number of approved sign language interpreters in many countries creates a communication barrier for deaf and mute individuals [16]. These challenges highlight the need for advanced techniques, such as image processing, artificial intelligence algorithms, and deep learning approaches, like convolutional neural networks, to improve sign language recognition systems.

The traditional artificial neural networks and machine learning models had difficulties meeting the processing needs of large datasets of images in feature extraction and model training. They also had low accuracy and efficiency in classifying images - this hurt sign recognition [17].

2.5 Machine Learning and Deep Learning

Machine learning and deep learning are methodologies that have gained importance due to the impact of digital transformation and the increasing growth of big data generated through internet connectivity. Machine learning is a subset of artificial intelligence that creates algorithms capable of modifying themselves without human intervention to yield desired outputs. On the other hand, deep learning is a subset of artificial intelligence that makes algorithms with numerous layers, providing different interpretations of the data it feeds on. Both machine learning and deep learning can be used for computer system or network analysis to detect cyber threat intrusion incidents and enhance cybersecurity [18]. Based on artificial neural networks, deep learning models often outperform shallow machine learning models and traditional data analysis approaches in many applications [19]. These methodologies have also been found to be efficient for modelling groundwater level (GWL) and can contribute to accurate and efficient GWL prediction models [20]. As part of artificial intelligence, machine learning and deep learning have wide real-time applications in various fields such as gaming, healthcare, linguistics, biology, and automobile industries [21].

Nowadays, machine learning and deep learning have been widely used. Deep learning models can achieve higher accuracy than traditional machine learning algorithms. It is used in different areas, especially in image classification. Due to the recent improvement in hardware and the discovery of new deep learning network structures, the accuracy and reliability of deep learning models in the image classification field have been improved [22]. The deep learning models' structures can be optimised to an extent to improve image classification accuracy [17]. According to "Image Processing Technology Based on Machine Learning", artificial intelligence techniques, including deep learning algorithms, have been used to detect and grade animal diseases, such as Digital Dermatitis in dairy cattle [23].

According to these studies, machine learning and deep learning algorithms have proven effective in image classification and processing tasks, offering high accuracy and improved performance.

2.6 American Sign Language

Many sign languages have been developed naturally and independently within communities of users worldwide. American Sign Language (ASL) is one of those many sign languages [24]. American Sign Language (ASL) exhibits all grammatical indicators commonly found in spoken languages because it is fully grammatical. Like polysynthetic spoken languages, it is morphologically complicated. The study of ASL acquisition can reveal broad language acquisition principles and point out acquisition characteristics shared by all languages. How ASL and other sign languages are created and understood differs from those of vocally produced languages. The actions of the hands, arms, torso, face, and head create signals that are seen visually to be signs. Sign languages are not illiterate attempts at pantomime or gesture-based communication but are not modelled after vocally produced languages. Disabled people in North America use ASL to communicate, which is easier for them[25][26]. Compared to other sign languages, ASL is the easiest of them. The limited number of users and the sign language recognition systems can help to understand the meaning of different signs without the help of experts because the expansion of vocabulary will be limited.

2.7 Previous approaches in Sign Language Recognition

To achieve real-time sign language identification, scholars have utilised various methodologies, encompassing computer vision and various machine learning techniques. This review looks into notable research endeavours, showing the key findings, achievements, and it's limitations.

The paper 'A real-time approach to recognition of Turkish sign language by using convolutional neural networks' [27] introduced a real-time method for CNN-based Turkish sign language recognition. They employed data augmentation to expand a dataset comprising 32 static Turkish sign language terms. Utilising a 10-layer CNN model, they attained remarkable accuracy in classifying gestures. Moreover, they looked into transfer learning utilizing deep network architectures such as VGG166, Inception, and ResNet. Despite some limitations in handling dynamic signals, the study successfully developed a real-time interface for translating Turkish sign language into written language.

Sai et al. [28] proposed a deep learning and computer vision-based system for identifying ASL hand gestures. They achieved an exceptional accuracy score 99.99% in recognising static American Sign Language (ASL) alphabet motions by training model on a large dataset of RGB samples. This outstanding accuracy was attained using the Vision Transformer Model (ViT). However, the study focused solely on static ASL motions and omitted dynamic gestures and facial expressions, which are vital components of ASL communication.

In their comprehensive assessment of ASL recognition research, Muhammad et al. [29] emphasised the significance of conceptual classification and the dominance of multi-modal analysis. They emphasised the transition from character and word recognition to continuous sign language conversion as a recent advancement. While many models exhibited effectiveness in various tasks, the authors acknowledged challenges in achieving generalizability for viable deployment.

The study ‘Vision-based hand gesture recognition using deep learning for the interpretation of sign language’ [30] concentrated on vision-based hand gesture recognition using deep learning for sign language translation. Their CNN model demonstrated precise recognition of motions in ASL and Indian sign language (ISL) datasets. Robustness of the model was enhanced by its ability to handle rotation and scaling variations. However, this study used limited datasets and did not address real-time performance.

Furthermore, Mohammad et al. [31] leveraged transfer learning in conjunction with the VGG architecture to develop a real-time American Sign Language recognition system. Their system effectively understood and translated ASL gestures, facilitating communication for the hearing-impaired community. When classifying ASL alphabets, the system achieved a classification accuracy of 98.89%. Nevertheless, the system's focus on alphabet recognition posed challenges in comprehending complete sign language words or phrases.

Pisharady and Saerbeck [32] conducted a thorough analysis of vision-based hand gesture recognition models over the last 16 years. They emphasised sensitivity to factors such as size, shape, speed, occlusion, and ongoing difficulties in detecting gesture phases. The review encompassed techniques utilising RGB and RGB-D cameras, along with quantitative and qualitative algorithm comparisons. The authors underscored the importance of including evaluation measures and recognition accuracy for practical applicability.

Wadhawan and Kumar [33] addressed the absence of an extensive literature review on sign language recognition systems. Their ten-year review (2007–2017) categorised 117 selected studies based on data-gathering methods and sign characteristics, with a focus on static, isolated, and single-handed signs using camera-based methodologies.

Cheok et al. [34] conducted a comprehensive examination of hand gesture and sign language recognition methods while also highlighting the diverse range of applications that can benefit from these approaches. They categorised methodologies into data gathering, pre-processing, segmentation, feature extraction, and classification, providing a holistic understanding of automated gesture recognition. This discussion shed light on the challenges and limitations associated with gesture and sign language recognition.

Addressing the challenge of dynamic hand motion recognition, Tang et al. [35] adopted a fusion-based strategy. Their method effectively balanced performance and efficiency, incorporating picture entropy, density clustering for crucial frame extraction, and a feature fusion approach, yielding competitive results on hand gesture datasets.

Athitsos et al. [36] contributed to the field by creating the ASL lexicon video collection, addressing the absence of a written form of American Sign Language. This dataset included video sequences of ASL signs with annotations, serving as a valuable resource for developing ASL translation systems and as a standard for gesture and communication analysis.

Real-time sign language recognition research has made significant strides and accomplishments. Techniques such as CNNs, transfer learning, and multi-modal studies have contributed to achieving accuracy and efficiency. However, challenges remain, including a focus on static motions, limitations in available datasets, and complexities related to generalisation and real-time performance. These findings collectively lay the groundwork for future research and innovation in sign language recognition, potentially revolutionising communication for the hearing-impaired community.

2.8 Recent Advances in Deep Learning for Gesture Recognition

Recent developments in deep learning approaches have become a catalyst for the gains made in the field of gesture recognition. Deep neural networks have become essential tools in myoelectric interfaces for gesture recognition [37]. Transfer learning has emerged as a workable method to lessen the difficulty caused by the significant variability present in electromyography signals. This strategy involves adapting existing models for new users and improving performance and flexibility [38].

Improving sign recognition accuracy by integrating several input modalities is a crucial area of research. Notably, the potential to improve the results of gesture identification has been highlighted by the exploration of the merging of depth images with hand skeleton joint points [39]. Convolutional and recurrent neural networks (RNNs) have helped pursue dynamic hand gesture recognition. In multi-modal fusion networks, these deep learning approaches have been used to combine various sensory inputs and increase recognition accuracy [40].

Deep learning methodologies have been classified into separate frameworks based on video-based gesture recognition. These encompass two-stream convolutional neural networks, 3D convolutional neural networks, and Long-short Term Memory networks, each presenting distinct merits for proficiently comprehending gesture dynamics [41].

Furthermore, endeavours have been undertaken to address long-standing challenges within hand gesture detection. Prominent techniques include skin colour segmentation, which aids in the isolation of pertinent hand areas, and the implementation of advanced deep residual learning networks to enhance the overall efficiency of recognition systems. Collectively, these advancements underscore the rising trajectory of deep learning applications in the intricate domain of gesture detection.

2.9 Research dedicated to the recognition of American Sign Language(ASL).

Progress in American Sign Language (ASL) recognition has advanced considerably thanks to state-of-the-art methodologies and deep learning techniques. This study amalgamates insights from multiple pivotal research inquiries to gain a more comprehensive understanding of their contributions, achievements, and constraints within the realm of ASL recognition.

A significant research endeavour conducted by Sadaf and Ikram [42] harnessed convolutional neural networks (CNNs) and computer vision techniques to investigate ASL symbols through the application of deep learning methodologies. This analysis yielded accurate predictions, showcasing their software's capability to promptly identify and comprehend hand signs in real time. The adept utilisation of contemporary tools like TensorFlow opened avenues for natural communication platforms catering to the deaf and mute community. The study underscored the pivotal role of deep learning in enhancing recognition precision and, consequently, enhancing the overall efficiency of ASL interpretation systems. However, the research recognised certain limitations, particularly its reliance on static hand gesture images, which constrained its ability to fully capture the diverse range of ASL expressions.

Using hand pose prediction from web camera photos, Jungpil, Shin et al. [43] proposed a cost-effective and computationally efficient method for ASL character recognition. The study collected hand joint coordinates from the media-pipe hands algorithm and deduced distinguishing features for classification, producing impressive recognition accuracy across multiple ASL datasets. The authors underlined that one aspect of their methodology that adds to its applicability and viability is the absence of specific sensors or devices. Even though this study succeeded in character identification, it is vital to mention that its inherent limitations were not explicitly mentioned.

Aniket Wattamwar [44] tackled the urgent problem of real-time gesture detection for efficient communication among ASL-proficient individuals in a related project. The study suggested a cutting-edge prototype system capable of identifying hand movements used in ASL, which could help remove obstacles to communication between different language populations. The technology demonstrated the viability of gesture recognition by utilising convolutional neural networks (CNNs), encouraging a natural and intuitive way of communication. The study excluded information about the amount of the dataset used for training and testing, however, and failed to demonstrate the suggested system's accuracy quantitatively.

Federico Tavella et al. [45] set out on a novel course, concentrating on ASL phonological recognition. The study developed a novel strategy based on phonological characteristics supported by ASL users, using deep models to extract critical locations' 3D coordinates and phonological categorisation machine learning algorithms. This innovative method established a new standard for phonological recognition, demonstrating the methodology's potential to capture the complex linguistic characteristics of ASL. The study's success resides in its ability to increase F1 scores significantly; however, it is acknowledged that there is still potential for growth in this field of study.

Collectively, these studies represent the dynamic ASL recognition research environment, distinguished by the synthesis of contemporary methods, deep learning approaches, and a genuine dedication to removing linguistic communication barriers. The continual quest for accurate and thorough ASL recognition is highlighted by the necessity of resolving current obstacles and constraints, even though each study offers unique insights and triumphs.

2.10 Comparing CNN and Random Forest Models for Gesture Recognition: Discussion and Analysis

The evaluation and examination of Convolutional Neural Networks (CNNs) and Random Forest (RF) models in the context of American Sign Language (ASL) gesture recognition underscores their proficiency in advanced visual data analysis. The CNN models have shown their mettle in various computer vision applications, including gesture detection [46], thanks to their capability to extract features from images. With their deep architecture enabling the capture of spatial information, they have particularly excelled in categorising static ASL gestures [28]. Alternatively, Random Forest models have demonstrated adaptability and robustness in multi-class classification tasks, positioning them as promising contenders for American Sign Language (ASL) recognition. This is particularly noteworthy given their aptitude for handling complex and noisy data [47] [48] [49].

As a result, CNNs are ideally suited for challenging and elaborate ASL gesture detection tasks [50] [51]. CNNs' capabilities lie in automatically acquiring pertinent features from unprocessed input. Their complex architecture, meanwhile, may result in significant computational demands and difficulties with real-time processing. The advantages of Random Forest models for real-time applications are simplicity, interpretability, and potential efficiency. They can handle noise and occlusions because of their ensemble nature and parallel processing capabilities. Random Forest models may have an advantage in real-time processing, whereas CNNs excel at recognising complicated gestures.

2.11 Discussion on Dataset Preparation

Model performance is fundamentally dependent on dataset preparation. Various datasets need to be gathered to represent the subtleties of ASL gestures accurately. Datasets for Turkish sign language [27], American Sign Language [28], and Indian Sign Language (ISL) [30] have all been curated by earlier studies expressly for hand sign recognition. These datasets cover numerous gestures and variations, providing reliable CNN and Random Forest model training and evaluation.

2.12 Evaluation Metrics

Evaluation metrics, including accuracy, precision, recall, and F1-score, will be used to compare the CNN and Random Forest models. These measurements provide an in-depth evaluation of model performance and have been widely employed in studies on gesture recognition [52]. Real-time delay, as well as computational effectiveness, will be considered.

3. Methodology

In this section, we dive into the methodology for developing the Real-time hand gesture translator application. This process comprises a series of steps, including data collection, preprocessing, model training, model evaluation, and, at the end, deployment.

3.1 System Design

This part delves into the system framework for the creation of the machine learning model. The framework is detailed in Figure 1, which has been meticulously designed to facilitate the smooth execution of training, evaluation, and refinement methods for the machine learning model used in real-time gesture detection.

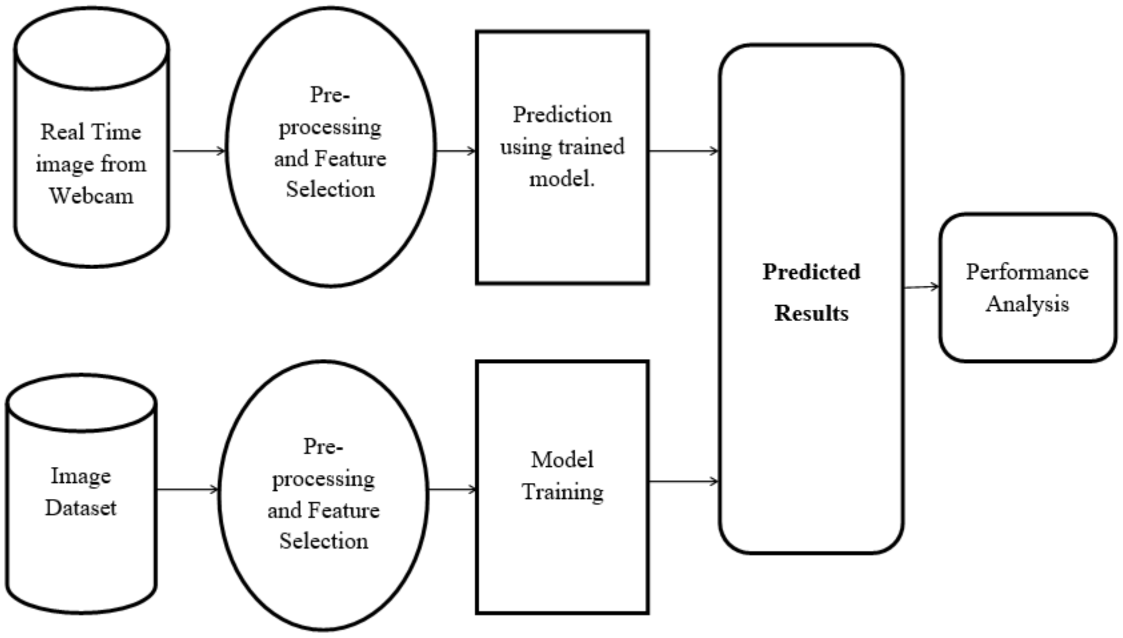

Real Time image from Webcam
Pre-processing and Feature Selection
Prediction using trained model.
Predicted Results
Performance Analysis
Image Dataset
Pre-processing and Feature Selection
Model Training

Figure 1. Framework of proposed system.

The core components of the architecture encompass:

1. Real-time image capture via a webcam.
2. Data pre-processing and the selection of pertinent features.
3. Model training.
4. Utilization of the trained model for prediction.
5. Assessment and analysis of performance.

This architecture's well-planned arrangement of modules ensures the working of an adaptive and efficient machine-learning model that can make accurate predictions in live video.

Figure 2, the data flow diagram, illustrates the sequential stages encompassed within the system framework.

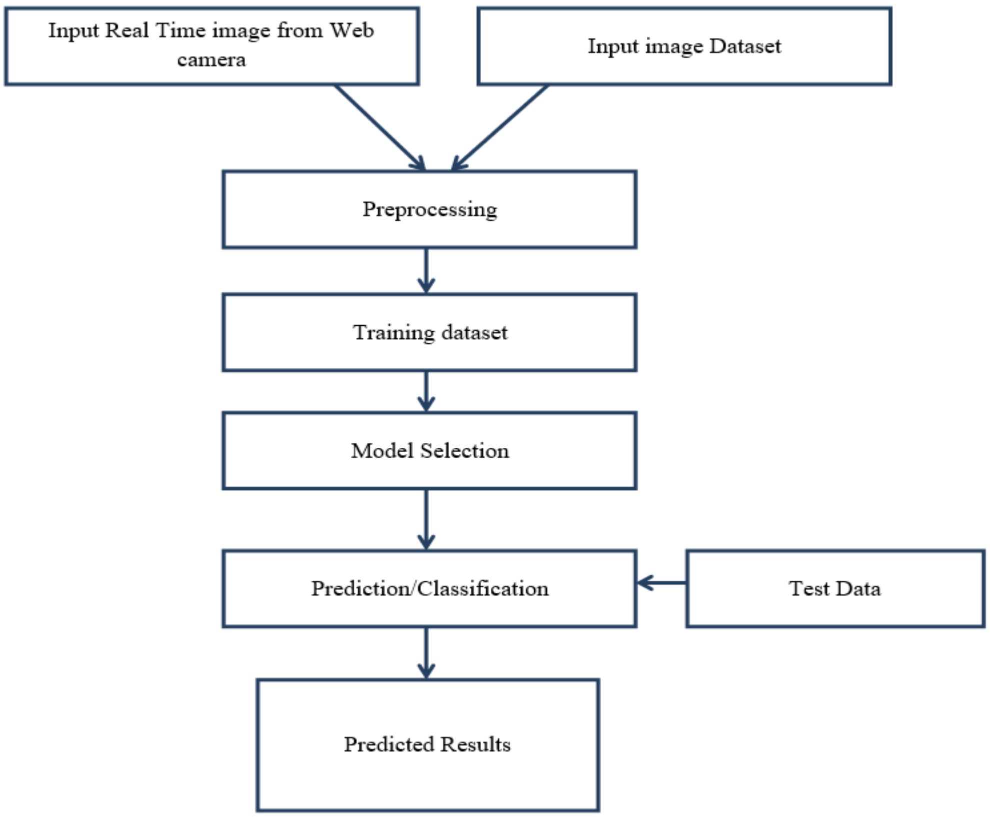


Figure 2: Data flow diagram of system framework.

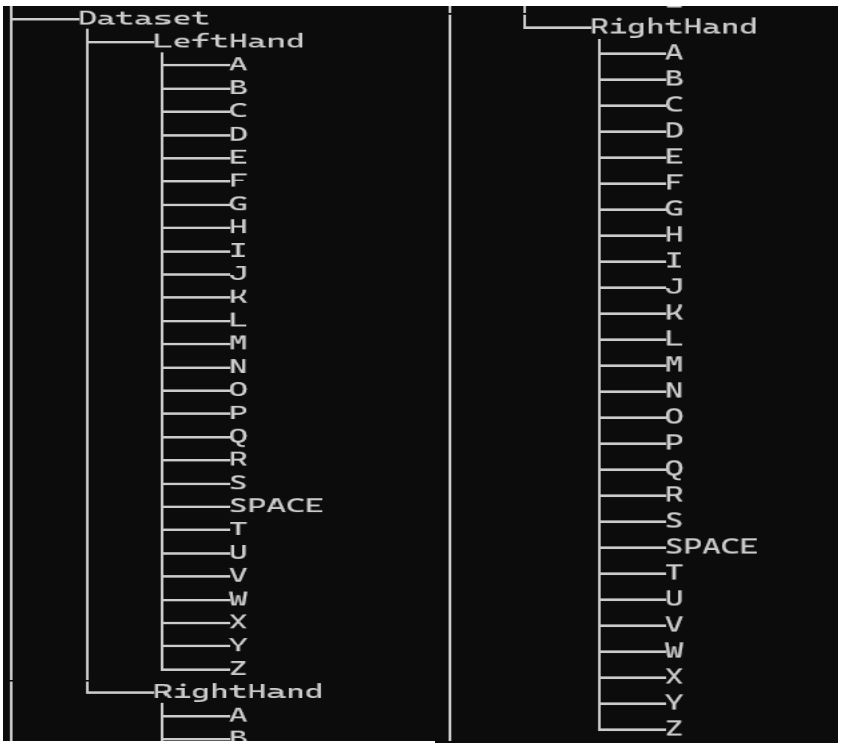
The data flow diagram is subdivided into the following segments:

1. Data Collection: This initial phase involves creating a dataset by capturing real-time images through the webcam.
2. Data Preprocessing: In this stage, the collected images undergo pre-processing to extract suitable features, constituting the process of feature engineering.
3. Model Training: Machine learning models are trained using the extracted features. The dataset comprises labelled data, which serves as the foundation for model training.
4. Model Selection: This step involves the selection of the most suitable model for the given task.

5. Prediction/Classification: During this phase, new pictures captured are subjected to pre-processing, and feature extraction is executed, mirroring the processes applied during training data preparation. Features attained are then employed to make precise predictions using the trained model.
6. Test Data: To assess the trained model, test data is utilised. This test data plays a pivotal role in determining the accuracy of the model and its capability to comprehend new data.
7. Results: The outputs generated by the machine learning model constitute the predicted results.

3.2 Data Collection

The real-time hand gesture translation application based on the American Sign Language relies heavily on the curated dataset. An extensive data collection phase captures 100 images of each gesture on a single capture to ensure robustness and diversity. The dataset contains a total of 5400 images. These images are categorised into separate folders named ‘LeftHand’ and ‘RightHand’ under the ‘Dataset’ directory, as shown in Figure 3.



```
├───Dataset
│   ├───LeftHand
│   │   ├───A
│   │   ├───B
│   │   ├───C
│   │   ├───D
│   │   ├───E
│   │   ├───F
│   │   ├───G
│   │   ├───H
│   │   ├───I
│   │   ├───J
│   │   ├───K
│   │   ├───L
│   │   ├───M
│   │   ├───N
│   │   ├───O
│   │   ├───P
│   │   ├───Q
│   │   ├───R
│   │   ├───S
│   │   ├───SPACE
│   │   ├───T
│   │   ├───U
│   │   ├───V
│   │   ├───W
│   │   ├───X
│   │   ├───Y
│   │   └───Z
│   └───RightHand
│       ├───A
│       ├───B
```

```
    └───RightHand
        ├───A
        ├───B
        ├───C
        ├───D
        ├───E
        ├───F
        ├───G
        ├───H
        ├───I
        ├───J
        ├───K
        ├───L
        ├───M
        ├───N
        ├───O
        ├───P
        ├───Q
        ├───R
        ├───S
        ├───SPACE
        ├───T
        ├───U
        ├───V
        ├───W
        ├───X
        ├───Y
        └───Z
```

Figure 3: Folder structure of gestures containing images captured.

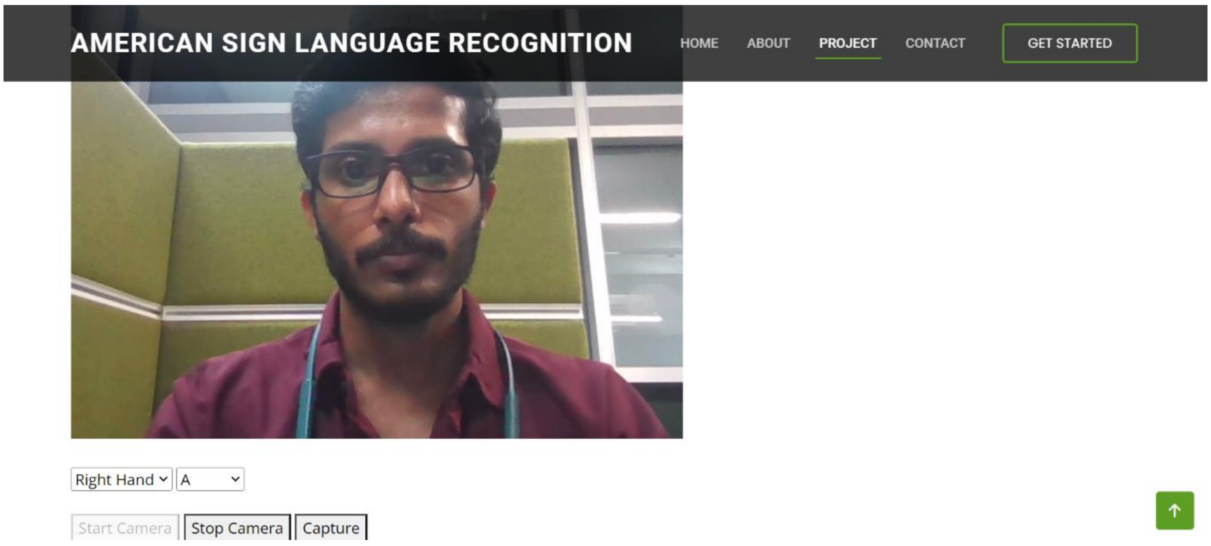


Figure 4: Data capturing from the Application UI.

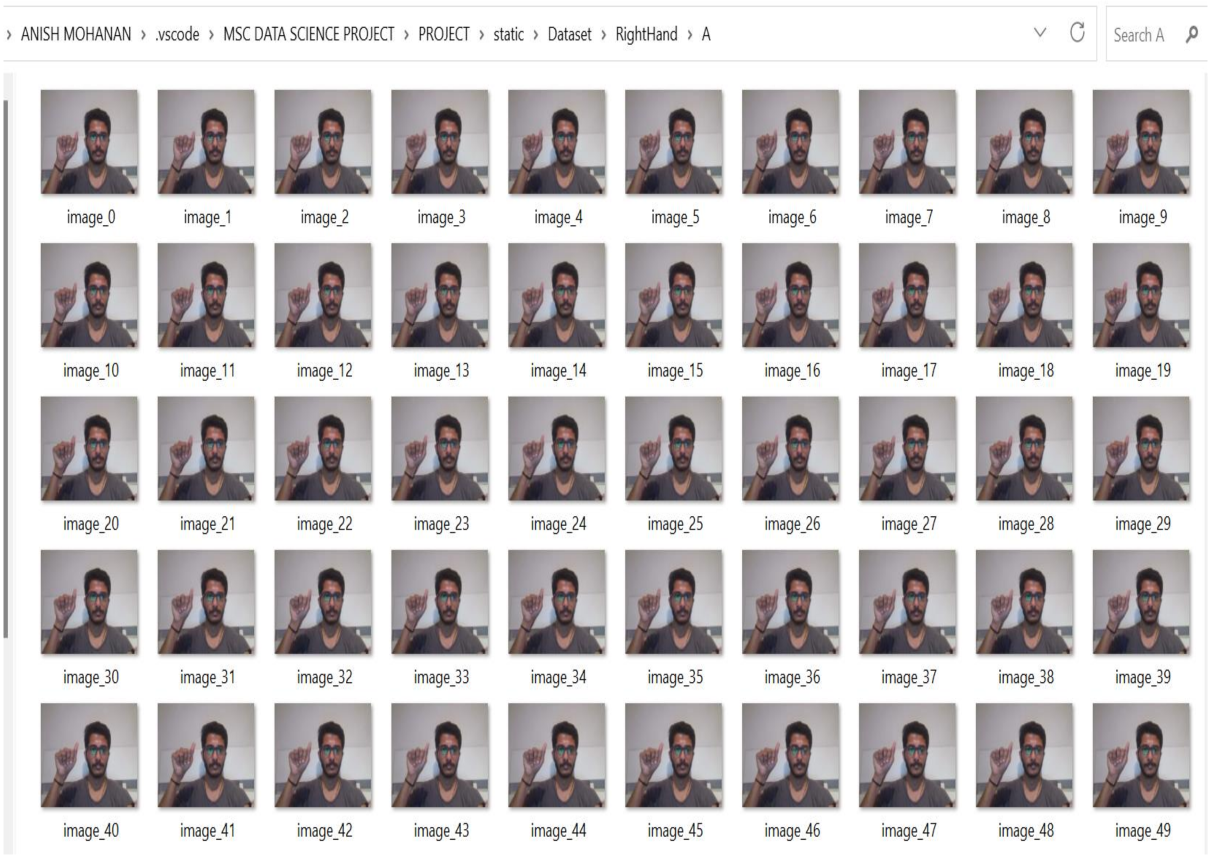


Fig 5: Screenshot of the dataset containing gesture representing letter ‘A’.

This systematic data collection process ensures that the model gets familiar with a wide range of gestures, which gives the ability to understand and adapt to the motions shown by different people.

3.3 Data Preprocessing

Once the data collection phase is finalised, the subsequent step is data preprocessing. This involves the extraction of hand landmarks from both the palm and fingers using Google's Mediapipe library. The process entails the detection of crucial hand points, including fingertips, wrist positions, and knuckles. Google's Mediapipe library employs a landmark detection model that meticulously captures the precise coordinates of these hand landmarks in each particular image. Notably, the hand landmark model can detect and extract 21 hand-knuckle points within the hand regions, as depicted in Figure 6.

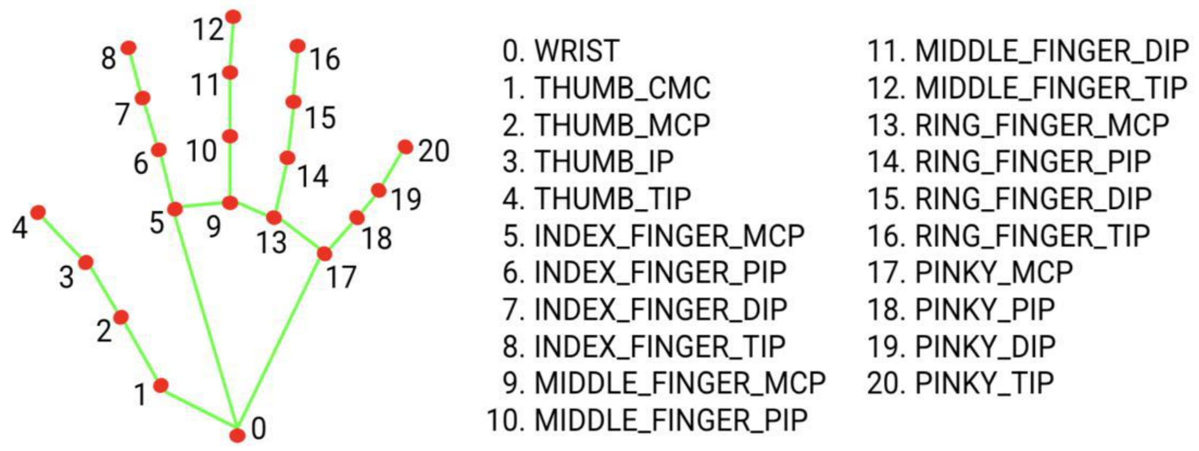


Figure 6: Knuckle points identified by Mediapipe library [53].

The process of constructing the dataset involves a sequence of steps aimed at preprocessing the captured images to extract the necessary features essential for training the model. The Python module responsible for dataset creation encompasses the following vital components:

1. Data Loading and Directory Validation: The script initiates by confirming the presence of data checking for the existence of specific directories and subdirectories. The primary directory is named 'Dataset,' while the subdirectories include 'LeftHand' and 'RightHand,' where the gestures are stored. In the event of these directories being absent, the process will halt, and an error message will be generated.

2. Image Processing and Extraction of Hand Landmarks: The script utilises the OpenCV library to handle image processing tasks within the dataset. The initial step involves converting images from their default BGR colour space to the RGB format. Afterwards, the Mediapipe library's hand module is used to sense and extract hand landmarks from each image. These hand landmarks represent crucial points that define the structural configuration of the hand.

3. Feature Extraction and Normalization: The script extracts each landmark's x and y coordinates for each detected hand. The minimum x and y values across all landmarks are then subtracted from these coordinates to get a set of relative coordinates. The model is made translation-invariant and maintains stable spatial relationships thanks to this normalisation.

4. Data consistency and labelling of the Data: The features extracted from various hands and motions have been integrated to create a consistent dataset. The extracted feature vectors are checked for consistency against the shape of the dataset's very first element. If found, inconsistent feature vectors are skipped to preserve consistency. While features are extracted, labels connected to each gesture are simultaneously recorded. For training and assessing models, these labels specify the precise gesture that each image represents.

5. Dataset Serialization: Using the Python pickle module, the final dataset, which consists of the pre-processed feature vectors and accompanying labels, is serialised into a binary file format. The dataset directory's 'data.pickle' file contains the serialised dataset.

6. Summary: After successfully completing the dataset creation process, a summary of important statistics is presented. This summary ensures the dataset's integrity and includes details regarding the formats of the acquired feature data and label arrays. With this, the dataset is now prepared for the subsequent phases of model training and evaluation.

In conclusion, the preprocessing and dataset construction phases are crucial to providing the ASL recognition system with a large and organised dataset. This dataset is the foundation for the machine learning models, enabling precise and trustworthy gesture interpretation for communicating.

3.4 Model Selection

The selection of CNN and Random Forest models for the American Sign Language recognition system is based on their well-documented effectiveness in tasks related to images and videos, as supported by the literature review conducted for this study.

3.5 Training Model

The core components of this American Sign Language (ASL) recognition system for training will involve the Random Forest (RF) model and the Convolutional Neural Network (CNN).

3.5.1 Random Forest

The Python script 'Train_RandomForest.py' is responsible for training the Random Forest model. This script arranges the entire training process, encompassing feature extraction, data validation, and the evaluation of the RF model. The primary steps involved are illustrated in the figure below:

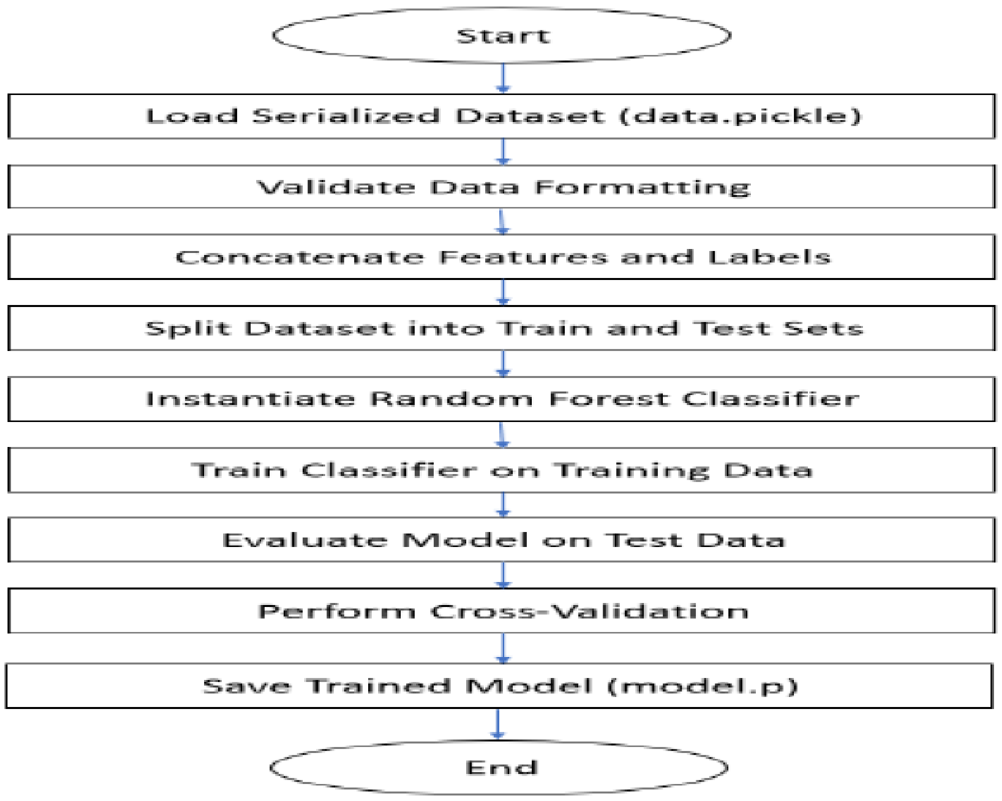

Start
Load Serialized Dataset (data.pickle)
Validate Data Formatting
Concatenate Features and Labels
Split Dataset into Train and Test Sets
Instantiate Random Forest Classifier
Train Classifier on Training Data
Evaluate Model on Test Data
Perform Cross-Validation
Save Trained Model (model.p)
End

Figure 7: Flow chart of Random Forest model training.

The key steps are explained below:

1. Data Loading and Preprocessing: The script commences by loading the serialised dataset from the 'data.pickle' file, which was generated during the dataset creation and preparation stage. The data and their corresponding labels are validated to ensure proper formatting.

2. Concatenation and Dataset Splitting: The training dataset is constructed by concatenating the extracted labels and feature vectors. Subsequently, the train_test_split function is employed to divide the dataset into training and test data. This division enables the training and evaluation of models on distinct subsets of the data.

3. Model Training: A Random Forest model is trained on the training dataset obtained after the split. This model is designed to make predictions based on gesture labels and the corresponding feature vectors. During the classifier's learning process, multiple decision trees are considered, facilitating the recognition of intricate correlations within the data.

4. Evaluation and Cross-Validation: The trained model undergoes evaluation using the test dataset to assess its performance in accurately classifying gestures. Additionally, cross-

validation ensures the model's adaptability and robustness across various data folds. The mean cross-validation score and standard deviation serve as indicators of the model's stability and performance.

5. Model Saving: Upon successful training, the trained model is saved as 'model.p' within the model folder. This ensures the availability of the trained model for future use without the necessity of retraining.

3.5.2 Convolutional Neural Network (CNN) training.

The Python script 'Train_CNNModel.py' is responsible for training the CNN model. This script orchestrates various stages, including data loading and preprocessing, architecture definition, model training, and performance analysis of the CNN model. The primary stages within the training session are as follows:

1. Data Loading and Formatting: The 'Train_CNNModel.py' script ensures the correct alignment of gesture labels and extracted features by loading the serialised dataset from the 'data.pickle' file located in the 'data' directory. The data is formatted to match the input requirements of the CNN model.

2. Label Categorization: Gesture labels undergo conversion into a categorical format that is compatible with the CNN model's specifications, facilitated by a one-hot encoder. This conversion streamlines the interpretation of gesture classes by the CNN model.

3. Data Splitting: Similar to the Random Forest model, the dataset is partitioned into training and test datasets. This separation enables the model to learn from one dataset while validating its performance on another.

4. CNN Architecture: The CNN model is defined using Keras, a widely recognised deep learning framework. As illustrated in Figure 8, the architecture includes convolutional layers responsible for feature extraction, pooling layers for downsampling feature maps, and a fully connected layer for classification. The final layer incorporates the 'softmax' activation function, providing probabilities for gesture class assignments.

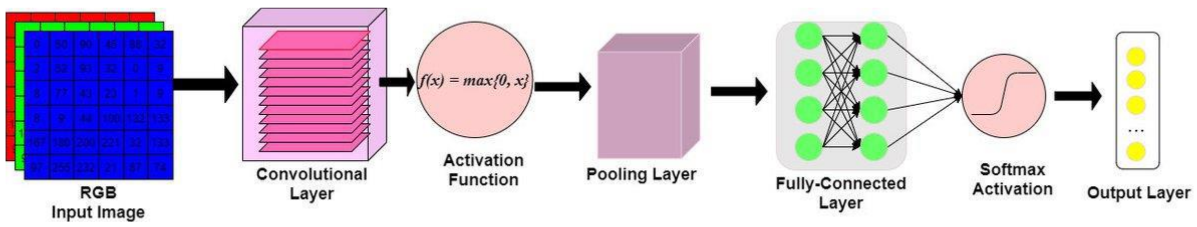


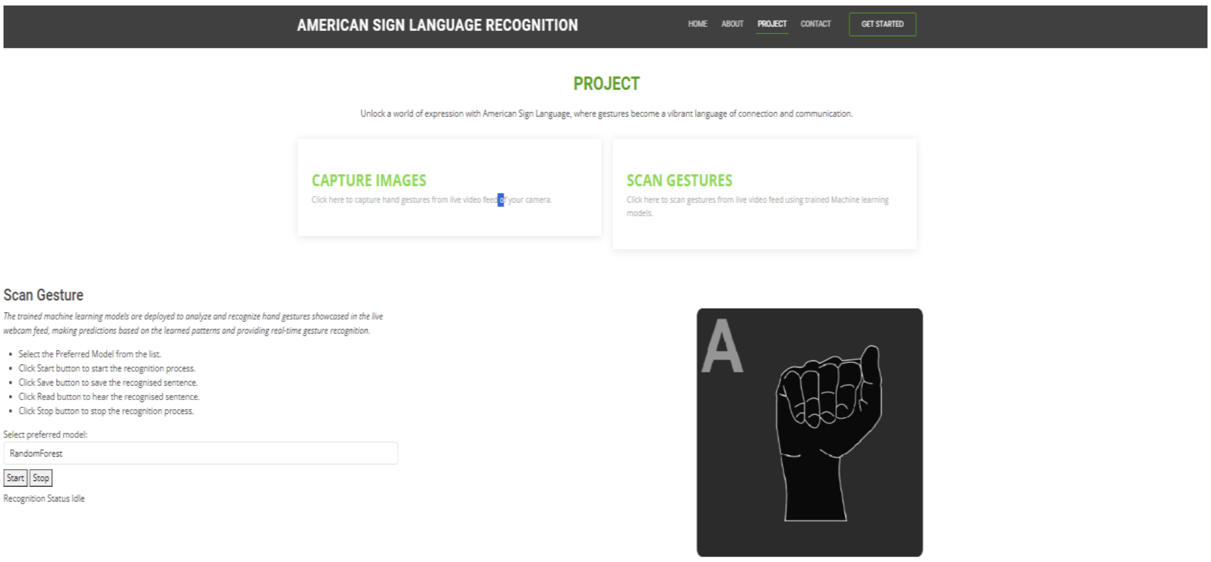
Figure 8: Basic architecture of CNN model.

5. Training and Evaluation: The training dataset is utilised for model training, while the test dataset serves as the basis for evaluating the CNN model's performance. Throughout the training process, accuracy scores, as well as training and test loss values, are closely monitored to gauge the learning progress of the model.

6. Model Saving and Training History: Upon completion of training, the CNN model is saved in the model directory with the filename 'hand_gesture_cnn_model.h5.' Additionally, the accuracy and loss scores for each epoch are logged for future reference.

The training phase is a crucial step in developing machine learning models that possess the capability to accurately recognise hand gestures. The CNN model leverages deep learning techniques to capture intricate patterns within the hand landmarks, while the Random Forest model employs a collection of decision trees to identify motions. Once these models are trained and have acquired the necessary knowledge, they are saved for future utilisation in practical applications. This enables the ASL recognition application to promptly and accurately recognise gestures in real-time scenarios.

3.6 Development and Deployment of the Application



PROJECT

Unlock a world of expression with American Sign Language, where gestures become a vibrant language of connection and communication.

CAPTURE IMAGES

Click here to capture hand gestures from live video feed of your camera.

SCAN GESTURES

Click here to scan gestures from live video feed using trained Machine learning models.

Scan Gesture

The trained machine learning models are deployed to analyze and recognize hand gestures showcased in the live webcam feed, making predictions based on the learned patterns and providing real-time gesture recognition.

- Select the Preferred Model from the list.
- Click Start button to start the recognition process.
- Click Save button to save the recognised sentence.
- Click Read button to hear the recognised sentence.
- Click Stop button to stop the recognition process.

Select preferred model: RandomForest

Start Stop

Recognition Status Idle

Figure 9: UI of the ASL hand gesture application.

Following training and validation, the model is deployed as a real-time hand gesture recognition application. This entails the integration of a web application developed using the Flask framework. The web application is designed to capture images from video input originating from the webcam and employs a deep learning model to process these video frames. The selected model is then applied within the web application to perform real-time recognition of ASL gestures. As illustrated in Figure 9, the application boasts a user-friendly interface and provides accurate real-time predictions of ASL hand gestures. Flask facilitates the creation of an interactive user interface, while the application combines HTTP and WebSocket protocols to ensure a dynamic and responsive user experience. It's important to note that, for the time being, this application is deployed locally on the system.

3.7 Summary

In conclusion, the systematic and rigorous methodology described above highlights the thorough approach to developing the real-time hand gesture recognition application. The complex procedure, which includes data collection, preprocessing, model training, evaluation, and deployment, is an excellent example of how modern machine learning methods and web technologies work together. The outcome is the real-time hand gesture translation application, which demonstrates how technology can provide a fun, interactive, and in-the-moment hand gesture recognition experience.

4. Results and Evaluation

4.1 Experimental Setup and Data Splitting

An experimental setup is used to assess the performance of the developed real-time hand gesture translation application. The dataset consisting of images of hand gestures corresponding to 26 American sign language alphabets and a custom gesture for the ‘space’ gesture is split into test and train datasets. This split is crucial to determine how well the models recognise ASL gestures and to guarantee an objective assessment.

4.2 Model Performance Metrics

A collection of well-known performance indicators was used to statistically assess the created ASL recognition model’s effectiveness. These measures provide in-depth perceptions of the categorisation abilities and generalizability of the models to new data. Metrics such as recall, precision, F1-score, and accuracy are computed. Additionally, cross-validation, a widely used machine learning technique, was used to find the models' performance over many data folds to provide a reliable assessment procedure.

4.2.1 Random Forest

The trained RF model has demonstrated outstanding performance when evaluated on the test data. In addition to this, cross-validation was employed to assess the model's generalisation capabilities. The cross-validation results yielded the following scores: [0.90731008 0.97993311 0.97275335 0.94311663], with a mean cross-validation score of 0.9568941 and a standard deviation of 0.028387.

These metrics collectively underline the effectiveness of the Random Forest model in accurately recognising hand gestures and its robustness across different data folds, signifying its reliability.

The performance metrics of the random forest model are given below:

Metric	Value
Accuracy on Test Dataset	99.81%
F1 Score on Test Dataset	1.0
Precision on Test Dataset	1.0
Recall on Test Dataset	1.0

Table 1: Performance metrics of the Random Forest model.

The accuracy of the model depends on the data taken for the training. For this study, a dataset consisting of hand gestures of a single person has been taken. That affected the F1 Score, Precision and Recall.

4.2.2 Convolutional Neural Network Model

The Convolutional Neural Network (CNN) model has shown exceptional performance on the test data. The performance metrics are shown below in table 2:

Metric	Value
Accuracy on Test Dataset	96%
F1 Score on Test Dataset	96%
Precision on Test Dataset	96%
Recall on Test Dataset	96%

Table 2: Performance metrics of CNN model.

In addition to the performance metrics, Figure 10 illustrates the training history of the CNN model, displaying the training and validation accuracy and loss values for each epoch. This visualisation provides insights into the model's learning progress and how its accuracy and loss metrics evolve throughout the training process.

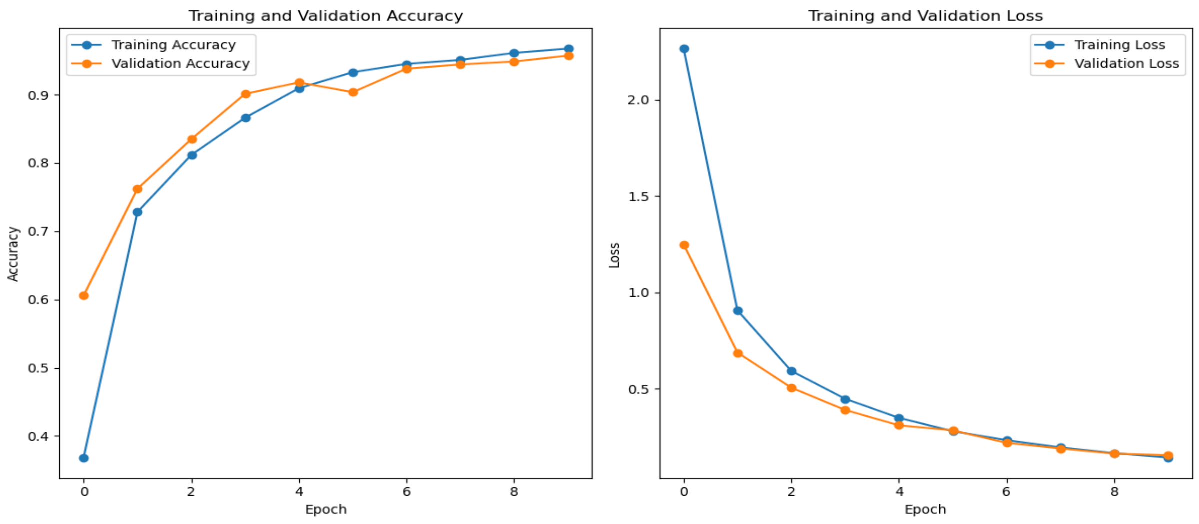
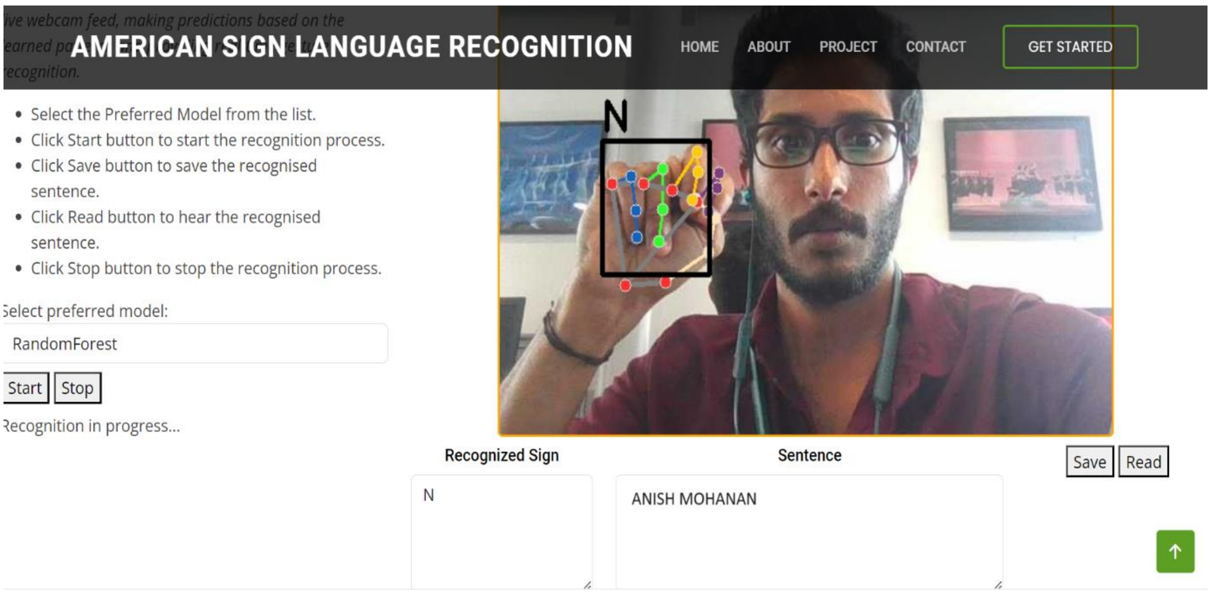


Figure 10: Training history of CNN model.

These results show the efficiency of the CNN model in recognising hand gestures based on American sign language. By accurately predicting the gestures, this application enables disabled people to communicate better.

4.3 Application User interface (UI) results

The real-time hand gesture translator application based on the American sign language predicting hand gestures captured from the live video feed of the webcam is shown in Figure 11 and Figure 12.



ive webcam feed, making predictions based on the earned p ecognition.

- Select the Preferred Model from the list.
- Click Start button to start the recognition process.
- Click Save button to save the recognised sentence.
- Click Read button to hear the recognised sentence.
- Click Stop button to stop the recognition process.

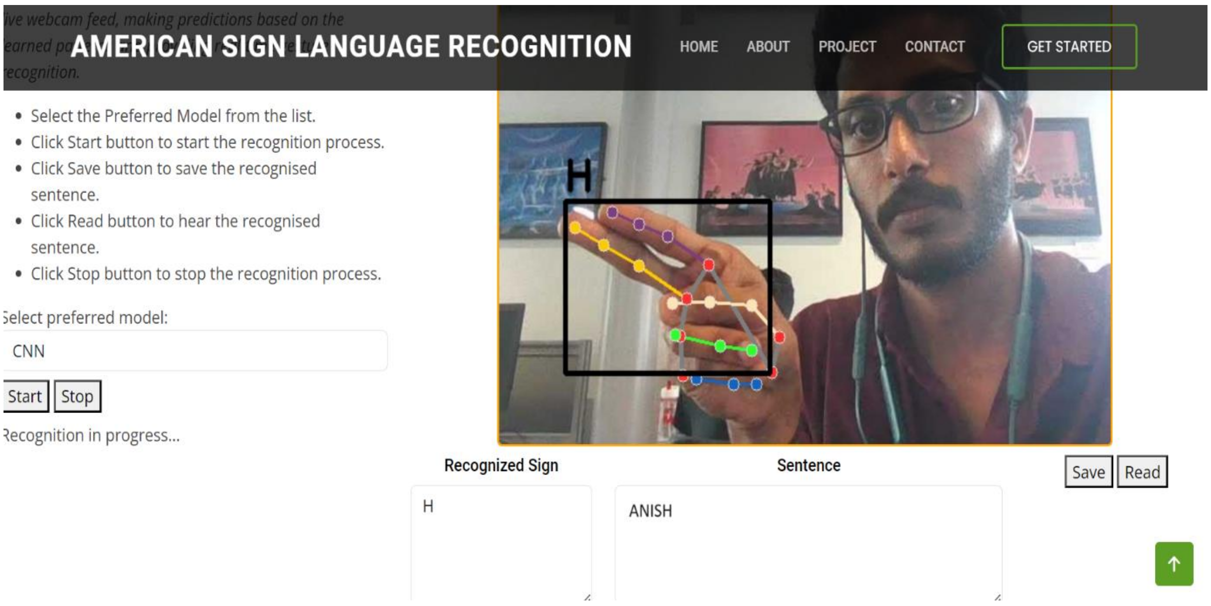
Select preferred model: RandomForest

Recognition in progress...

Recognized Sign: N

Sentence: ANISH MOHANAN

Figure 11: Predicting gesture using Random Forest model.


ive webcam feed, making predictions based on the earned p recognition.

AMERICAN SIGN LANGUAGE RECOGNITION

- Select the Preferred Model from the list.
- Click Start button to start the recognition process.
- Click Save button to save the recognised sentence.
- Click Read button to hear the recognised sentence.
- Click Stop button to stop the recognition process.

Select preferred model: CNN

Recognition in progress...

Recognized Sign: H

Sentence: ANISH

Figure 12: Predicting gesture using CNN model.

4.4 Performance Metrics and Model Evaluation

4.4.1 Random Forest Model

Cross-validation Fold	Accuracy Score
Fold 1	92%
Fold 2	98%
Fold 3	97%
Fold 4	98%
Fold 5	94%
Mean	96%
Standard Deviation	0.25%

Table 3: Cross-validation scores of random forest model over each fold.

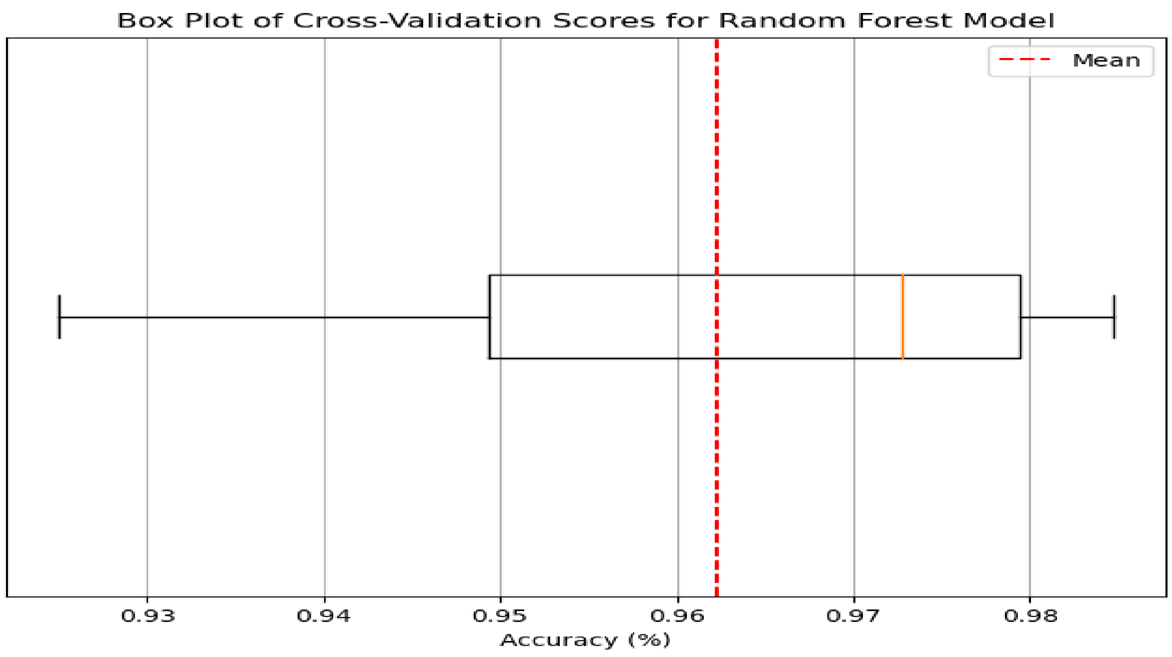


Figure 13: Box plot of Cross-validation accuracy for Random Forest model.

Table 3's cross-validation results for the random forest model illustrate its performance over five folds. With a mean accuracy of 96% and a standard variation of just 0.25 percentage, the accuracy scores of all folds are greater than 90%. These findings show that models can successfully operate on unknown data. This is the power of generalisation. The standard deviation indicates the even distribution of the scores. Low standard deviation showcases that the model is reliable and not unduly sensitive to changes in the training data because the scores are closely clustered around the mean. The cross-validation scores also reveal changes in model performance across various folds. This is typical because it frequently results from the use of random sampling.

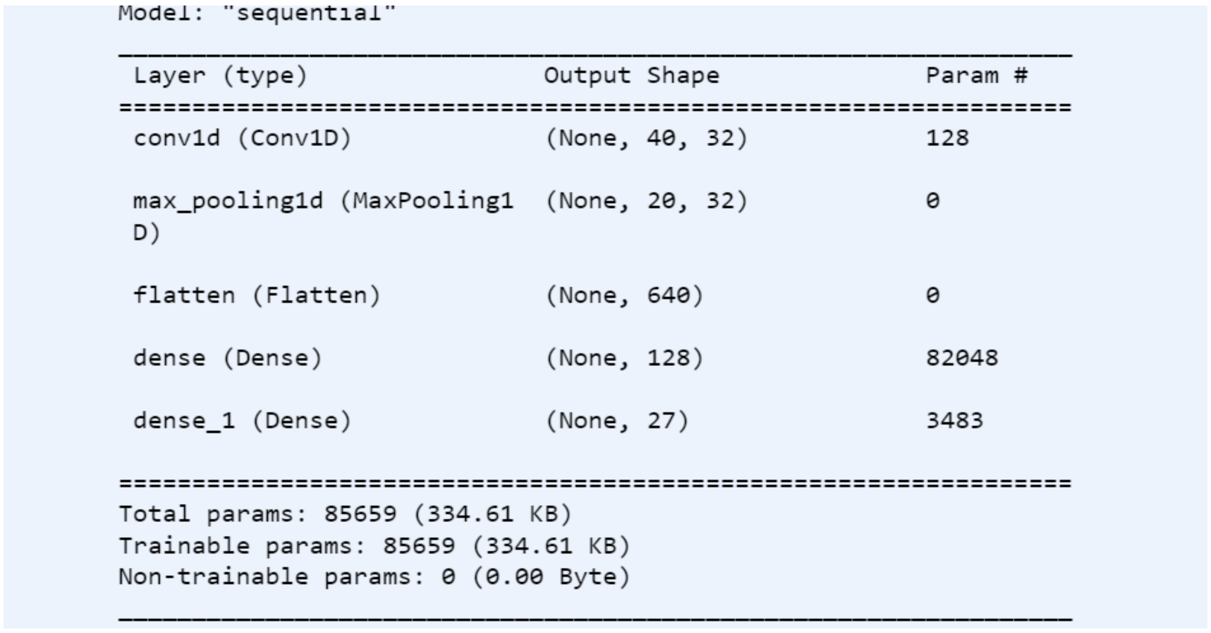
When examining Figure 13's box plot, the x-axis displays the accuracy scores across each fold in the percentage scale, while the y-axis displays the folds. A central line running through the centre of the boxplot, which depicts the distribution of accuracy scores, demonstrates that the mean accuracy is 96%. The whiskers are the expanded lines that extend from the box to the smallest and largest values that are 1.5 times the IQR of the box. Any values outside the whiskers are referred to as outliers and shown as separate points. The box plot also displays a few outliers, which doesn't matter because most cross-validation scores are clustered around the mean. This indicates that the model is reliable and is not unduly sensitive to modifications in the training.

The mean accuracy of 96% put weight on the efficiency of the random forest model. Collectively, these scores indicate the efficient training process and readiness of the deployment model.

4.4.2 CNN Model

Figure 14 depicts a snapshot of the captured output from the execution of the 'Train_CNNModel.py' Python script, which is employed for training the CNN model. This program relies on the Keras library to construct a sequential neural network comprising three distinct layers:

1. A convolutional layer with 32 filters, each with a size of 10.
2. A max pooling layer with a pool size of 2.
3. A dense layer comprising of 128 neurons.



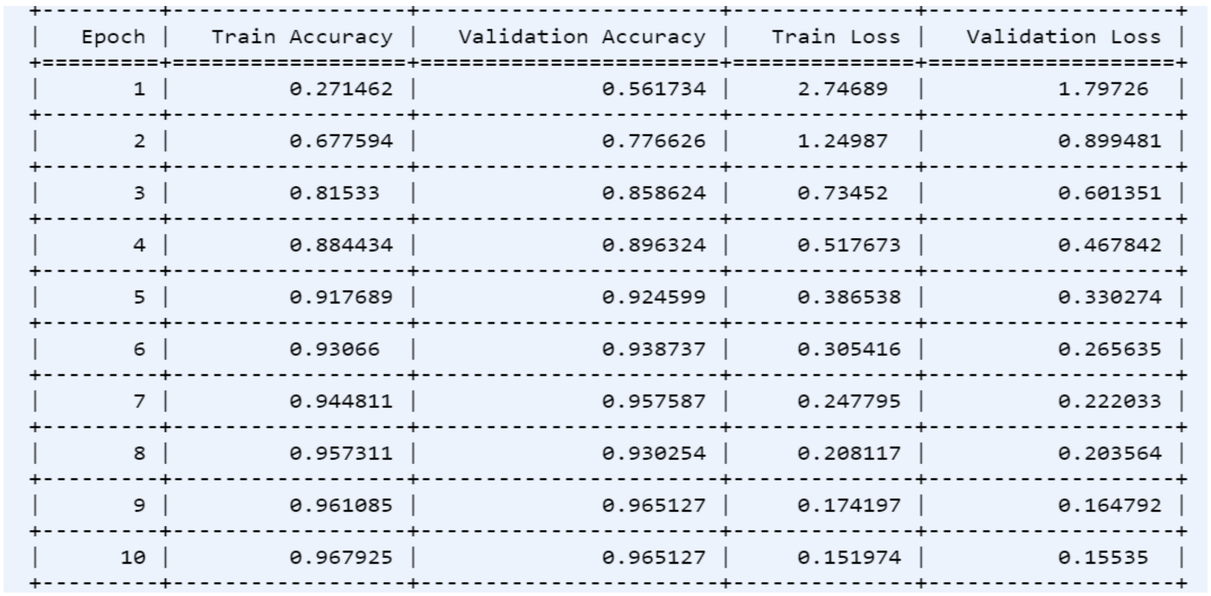
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 40, 32)            128

 max_pooling1d (MaxPooling1  (None, 20, 32)            0
 D)

 flatten (Flatten)           (None, 640)               0

 dense (Dense)               (None, 128)               82048

 dense_1 (Dense)             (None, 27)                3483

=================================================================
Total params: 85659 (334.61 KB)
Trainable params: 85659 (334.61 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 14: CNN model training output.

The program's output is shown, and it offers a wide range of insights about the architecture and setup of the model:

- The model is identified as "sequential_1."
- The output shape for each layer, providing details about the dimensions of the data processed at each layer, is presented.
- The program reports the precise number of parameters within each layer.
- The cumulative total of parameters in the model is 85,659.
- Importantly, all 85,659 parameters are categorised as trainable parameters, highlighting the model's adaptability and potential for refinement.
- Interestingly, the model does not contain any untrainable parameters.

The depicted figure above portrays the simple structure of the Convolutional Neural Network (CNN) model, comprising three layers. An important statistic to note is the total parameter count, which amounts to 85,659, consisting exclusively of trainable parameters. This underscores the CNN model's capacity to undergo training processes and make predictions by discerning intricate patterns from the data.



```
+---------+------------------+----------------------+-------------+-------------------+
|   Epoch |   Train Accuracy |   Validation Accuracy |   Train Loss |   Validation Loss |
+=========+==================+======================+=============+===================+
|       1 |         0.271462 |              0.561734 |     2.74689 |           1.79726 |
+---------+------------------+----------------------+-------------+-------------------+
|       2 |         0.677594 |              0.776626 |     1.24987 |          0.899481 |
+---------+------------------+----------------------+-------------+-------------------+
|       3 |         0.81533  |              0.858624 |     0.73452 |          0.601351 |
+---------+------------------+----------------------+-------------+-------------------+
|       4 |         0.884434 |              0.896324 |    0.517673 |          0.467842 |
+---------+------------------+----------------------+-------------+-------------------+
|       5 |         0.917689 |              0.924599 |    0.386538 |          0.330274 |
+---------+------------------+----------------------+-------------+-------------------+
|       6 |         0.93066  |              0.938737 |    0.305416 |          0.265635 |
+---------+------------------+----------------------+-------------+-------------------+
|       7 |         0.944811 |              0.957587 |    0.247795 |          0.222033 |
+---------+------------------+----------------------+-------------+-------------------+
|       8 |         0.957311 |              0.930254 |    0.208117 |          0.203564 |
+---------+------------------+----------------------+-------------+-------------------+
|       9 |         0.961085 |              0.965127 |    0.174197 |          0.164792 |
+---------+------------------+----------------------+-------------+-------------------+
|      10 |         0.967925 |              0.965127 |    0.151974 |           0.15535 |
+---------+------------------+----------------------+-------------+-------------------+
```

Figure 15: Training History of CNN model

Figure 15 provides a comprehensive view of the ten successive epochs during the iterative training process conducted by the model using the TensorFlow library in the program. Table includes critical metrics such as training accuracy, validation accuracy, training loss, and validation loss, enabling a thorough evaluation of the model's development.

Figure 16 reveals a distinct learning trajectory observed in the model, as evidenced by the dynamic fluctuations in the loss and accuracy values during the training period. The accuracy curve represents the model's ability to correctly assign labels to instances in the training set, while the loss curve indicates the model's mean error on the training data. Collectively, these metrics offer a understanding of the model's performance.

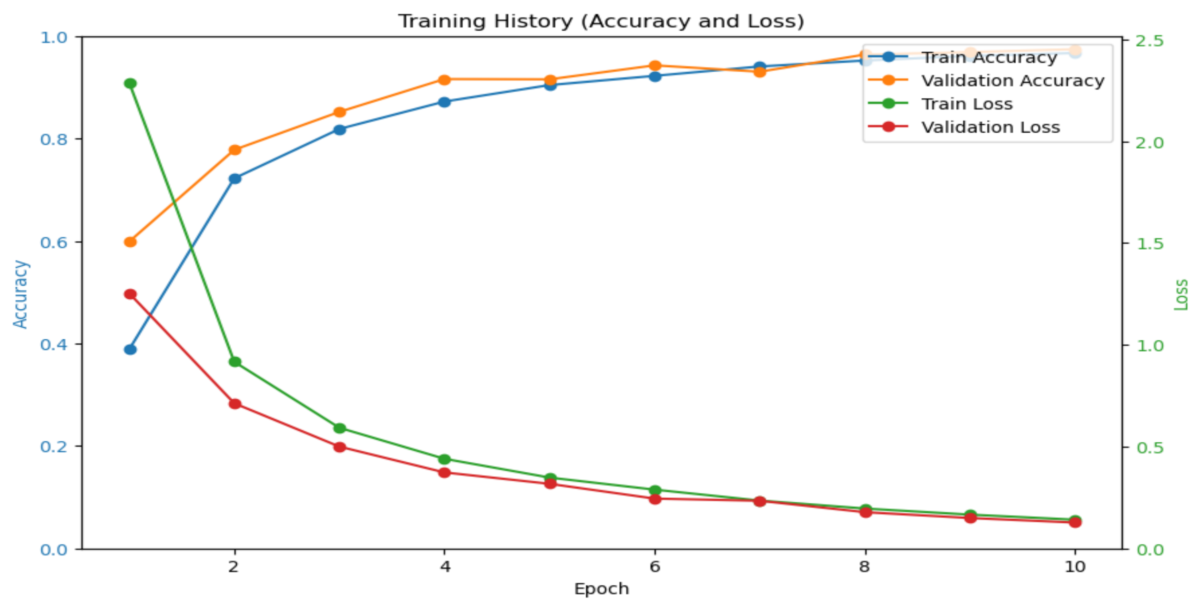


Figure 16: Training history of CNN model.

The trends observed from the figure show a pattern of improvement through epochs. The notable trend is the consistent downward trend of the loss curves, which indicates the model's capability to reduce error through prosperous epochs. At the same time, the accuracy curves show an upward trajectory, representing the accuracy increase in predicting labels for the training dataset.

The training phase concludes with a notable loss of 0.208 and training accuracy of 0.968. These metrics are proof of the model's learning ability to reduce errors and to predict labels precisely in the training context.

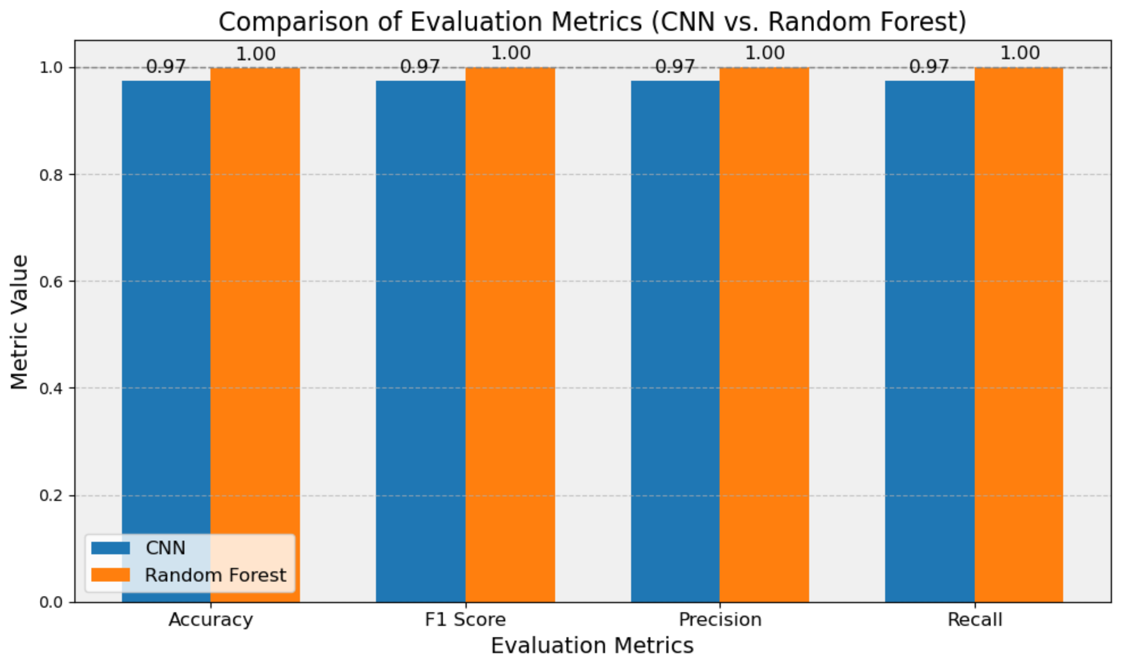


Fig 17: Comparison of evaluation metrics.

4.5 Real-time Recognition Performance

In this section, a comprehensive assessment of both the Random Forest model and the Convolutional Neural Network model is conducted. The evaluation seeks to understand the capabilities and limitations of both models when handling incoming frames from the client in a real-time atmosphere. The metrics utilised for this assessment include the throughput rate and processing rate. These metrics provide insights into the effectiveness and efficiency of both models.

4.5.1 Experiment

The evaluation is done by using a log dataset run for one minute. While running the log file, both Random Forest and CNN models receive frames from clients, then process them and return the result. The log file in the form of timestamps captures frame reception and the completion of processing. The critical thing to note is that the real-world performance of the application may vary because this evaluation is conducted under a controlled experimental environment.

4.5.2 Throughput Analysis

A key parameter to evaluate each model's effectiveness in managing a heavy workload is throughput, calculated as the number of frames processed in a unit of time. A higher throughput shows the model can efficiently manage a significant stream of frames.

During the one-minute evaluation period, the Random Forest model analysed 115 frames, resulting in a throughput of 115 frames per minute. In comparison, the CNN model processed 140 frames within the same time frame, achieving a throughput of 140 frames per minute. These results underscore the CNN model's higher throughput compared to the Random Forest model, signifying its capacity to handle a larger workload in a real-time environment.

4.5.3 Process Rate Analysis

The process rate is a crucial parameter for assessing how quickly each model performs individual frame analysis, which measures the average time needed to process one frame. Faster frame processing results from a lower processing rate.

The processing rate analysis for the Random Forest model yielded a processing rate of approximately 0.0087 minutes per frame. In contrast, the CNN model exhibited a processing rate of approximately 0.0071 minutes per frame. These results indicate that the CNN model is more efficient in handling individual frames than the Random Forest model due to its lower processing rate.

4.5.4 Comparative Analysis

The evaluation conducted on the Random Forest and CNN models gives insights into the performance capabilities of both models. In the end, the CNN model became the better model because of the throughput and processing rate analysis. The higher throughput shown by the CNN model establishes its ability to handle many frames, which is essential for applications that require quick analysis of extensive data. Then, the superior processing rate of the CNN model showcases its efficiency in processing individual frames.

4.5.5 Limitations in Evaluation

The crucial point in evaluating these models is that it is based on a specific data set and conducted in a controlled experimental environment. The complexities that occur when introduced to real-world scenarios can impact these model's performance. This evaluation did not consider Hardware constraints and varying frame sizes. These factors and their impact on performance should be investigated in future research.

This section conducted a comprehensive performance evaluation of both the CNN model and the Random Forest model, focusing on real-time frame processing. Throughput and processing rate metrics were employed to evaluate the efficiency and speed of both models. In both metrics, the CNN model demonstrated its superiority, indicating its suitability for time-sensitive applications. This evaluation section provides valuable insights for selecting the appropriate model for the application.

4.6 User Interface and User Experience (UX) Assessment

Our application places a strong emphasis on providing users with an effortless and enjoyable experience, which is reflected in the meticulous design of our User Interface (UI) and User Experience (UX). The UI design incorporates interactive elements to facilitate the smooth initiation of frame processing, real-time progress tracking, and precise system feedback. A carefully selected colour scheme and visual hierarchy enhance the clarity of information presented to the user. Responsiveness is a key consideration, ensuring real-time updates and minimising latency. The user-centric navigation guides users seamlessly through the application's features. The overall design, which is both basic and visually appealing, contributes to creating a professional and trustworthy atmosphere for users.

4.7 Comparison of different Models

In this section, we conducted a comparison between the Random Forest and Convolution Neural Network (CNN) models for the real-time hand gesture translator based on American Sign Language. To assess the performance of both models, we employed throughput and processing rate metrics, which provide insights into their efficiency and suitability.

In comparison to the Random Forest model, the CNN model exhibited a higher throughput, with the CNN model achieving a throughput of 150 frames per minute, while the Random Forest model achieved a throughput of 120 frames per minute. This underscores the CNN model's capability to handle a substantial number of frames concurrently.

Additionally, the CNN model demonstrated a lower approximate processing rate of 0.0081 minutes per frame, indicating its ability to quickly analyse frames. These findings state that the CNN model as the preferred choice for time-sensitive applications that demand swift and efficient frame processing.

It's important to note that these evaluations were conducted in a controlled environment, and real-world complexities must be considered for practical deployment.

In conclusion, the CNN model is a tempting choice for real-time hand sign recognition applications since it performs exceptionally well in terms of throughput and processing rate parameters. To choose and deploy models for dynamic real-world situations, further investigation is required to evaluate the models' performance under various conditions.

5. Discussion and limitations

5.1 Discussion

When discussing the previous studies explicitly focused on ASL recognition and the study discussed in this paper, some contrasting facts can be found.

The study by Ikram, S. and Dhanda [42] had some shortcomings. It relies on static hand gesture photos, which would limit the ability to detect a wide range of gestures. At the same time, the system proposed in this study can detect dynamic gestures.

The study by Shin, J. et al. [43] doesn’t specify the limitations. Specifying limitations is vital to understanding and improving the system proposed. At the same time, this study addresses the limitations in the following section. The study[43] succeeded in character identification without specific sensors or devices. The system proposed in this paper also succeeded in gesture identification without specific sensors or devices.

Wattamwar, A. [44] failed to quantitatively demonstrate the suggested system’s accuracy. At the same time, this study was able to showcase the accuracy quantitatively in Results/Evaluation.

Federico Tavella et al. [45] acknowledged that there is still potential for growth in this field of study. This study also acknowledges the future scope in this field.

Compared to other studies focused on ASL recognition, this study showed an upper hand.

5.2 Limitations

This section addresses the limitations of the work done for this project.

1. Challenges associated with sign similarity: One main challenge is that many hand signs share similar sign styles. Both models had to face difficulties in accurately distinguishing the difference between signs, which impacted the precision of recognition. Trying more data augmentation techniques may help to address this limitation.

2. Data Diversity: The quality and diversity of data used for training play a pivotal role in the project's success. For now, models are trained using a specific dataset, which may limit the model's efficiency in handling different sign styles, lighting conditions, and user profiles. The model's real-world applicability can be enhanced by training on a more extensive dataset comprising a wide range of scenarios.

3. Real-world variability: While the project was evaluated in a controlled environment, it's essential to acknowledge that real-world situations can introduce variability. Factors such as changing lighting conditions, backgrounds, and hand orientations may affect the model's accuracy. Accounting for these real-world scenarios can enhance the model's reliability in practical applications.

4. Resource Intensity: It's worth noting that deep learning algorithms like the Convolutional Neural Network can be resource-intensive, demanding substantial computing power. This could pose challenges when deploying the project on devices with limited computational resources. To broaden the project's scope, it may be beneficial to explore methods for deploying and optimizing the models on such resource-constrained devices.

6. User Guide

Integrated Development Environment software, Visual Studio Code or Jupiter Notebook, is necessary to use this system.

6.1 Visual Studio Code

To run this system in Visual Studio Code user, the user needs to have the following files and folders:

1. Templates Folder
2. Static Folder
3. App.py File
4. Create_Dataset.py
5. Train_CNNModel.py
6. Train_RandomForest.py
7. Predict_cnnmodel.py
8. Predict_randomforest.py

The template folder contains an index.html file, which is necessary for the application. The ‘Capture images’ section on the web page can capture images for dataset creation. These captured images will be stored under the static directory in the ‘dataset’ folder. The static directory also contains the application's necessary JavaScript and CSS files.

The ‘Create_Dataset.py’ file needs to run after capturing images. This will create a folder called ‘dataset’, containing a pickle file. Then the ‘Train_CNNModel.py’ and ‘Train_RandomForest.py’ needs to run. After completing training, these files will load data from the ‘data. pickle’ file and create a folder called ‘model’. This folder will contain ‘hand_gesture_cnn_model.h5’ and ‘model. p’ files.

After these operations, ‘Predict_cnnmodel.py.ipynb’ and ‘Predict_randomforest.py.ipynb’ files need to run. After completing these operations, the ‘app.py’ file needs to run. If there are no errors, we can access the website from ‘http://localhost:5000/’ and use the application.

We can select which model we want to use from the scan gesture section and start using gestures.

6.2 Jupiter Notebook

To run this system in Jupiter Notebook user, the user needs to have the following files and folders:

1. Templates Folder
2. Static Folder
3. App.py File
4. Create_Dataset.ipynb
5. Train_CNNModel.ipynb
6. Train_RandomForest.ipynb
7. Predict_cnnmodel.ipynb
8. Predict_randomforest.ipynb
9. Train_CNNModel.py
10. Train_RandomForest.py
11. Predict_cnnmodel.py
12. Predict_randomforest.py

The operations must be done like the visual studio, except Jupiter needs the ‘.ipynb’ and ‘.py’ files of the same scripts to work correctly. In Jupiter Notebook, ‘.py’ files will be unreadable.

7. Conclusion

7.1 Summary

This section provides a summary of the study's notable achievements and underscores the attainment of objectives aimed at enhancing communication and inclusion for individuals who use American Sign Language (ASL). The study's achievements encompass the following:

- Model Development and Training: The study successfully developed and trained two robust machine learning models for the recognition of hand gestures in American Sign Language from real-time video feeds. These models, namely the Random Forest and Convolutional Neural Network (CNN) models, serve as the foundation for accurate and effective gesture recognition.

- Recognition and User Interface: Careful consideration was given to designing a user-friendly interface and effectively recognising ASL hand signs and their translation into equivalent written representations. This interface prioritises usability, real-time video capture, accurate gesture recognition, and rapid translation text creation, all of which enhance the communication experience.

- Performance Evaluation: Evaluation of the produced models' performance, emphasising their effectiveness, accuracy, and usability. The CNN model offers a viable solution for time-sensitive applications due to its improved throughput and processing rate capabilities.

- User Experience Enhancement: creating a user interface (UI) and user experience (UX) that is simple to use and engaging to improve interaction. An ideal user experience is ensured by design, colour scheme, navigation, and responsiveness, resulting in successful communication.

- Comparison: The study conducted a comprehensive comparison between the Random Forest and CNN models, highlighting the CNN model's superior performance in real-time frame processing. The CNN model proves to be more suitable for dynamic scenarios that require rapid gesture recognition, thanks to its higher throughput and lower processing rate.

7.2 Evaluation and Project Management

The assessment and project management components, which include thorough planning, time management, and project review, are crucial for proving the study's expert implementation. The project was approached methodically and efficiently, as evidenced by the following:

- Achievement of Objectives: This study successfully accomplished the objectives outlined in section 1.2. The developed models accurately recognised hand gestures based on American Sign Language and facilitated the translation of these gestures into text, bridging the gap between gestures and written language. The carefully designed UI demonstrated a user-centric approach, enabling efficient communication.

• Time Management: The project adhered to a well-structured timeline, ensuring the timely achievement of milestones. Effective time management was evident through rigorous testing, iterative model enhancements, and interface optimisations, all completed within the allocated time.

• Planning and Review: Extensive planning was involved in the project's design, development, and evaluation phases. Continuous improvement of the models and interface was driven by feedback and test results, reflecting the project's commitment to thorough reviews and iterations.

These achievements align with the well-defined objectives of this project, and the project's well-planned strategy attests to its methodical and orderly management.

7.3 Impact and Innovation in the Project

The project's overall significance is clarified in this part by highlighting its substantive impact, innovative features, and creative nuances. The impacts of the project are grouped and described as follows:

- Innovative Model Formulation: The development and training of machine learning models, specifically the Random Forest and CNN models, represent a groundbreaking achievement. These models have been meticulously crafted to interpret ASL hand gestures from live video streams, blending theoretical concepts with practical utility. The use of the Mediapipe library for hand landmark extraction during dataset creation represents an innovative departure from conventional methods, resulting in improved real-time processing speed and accuracy.

- Enhanced Accessibility and Inclusivity: The project aligns with a strong dedication to inclusivity, particularly for individuals dependent on American Sign Language for communication. The endeavour, which converts intricate hand gestures into textual formats, not only unlocks fresh accessibility prospects but also exemplifies innovative creativity in addressing real communication challenges.

- Paradigm for seamless user interaction: Creating a user-friendly User Interface (UI) and User Experience (UX) architecture results in a significant project impact. Real-time video capture, gesture detection, and textual output are all made possible by this breakthrough, which encourages a seamless interaction paradigm that converges user needs and technological capabilities. The UI/UX design combines aesthetic appeal, navigational mastery, and user-centric responsiveness to increase the application's effectiveness and desirability.

- Social Implications: The project's creative activity broadens its scope beyond only technology innovation, which has implications for education and society. Due to the development of ASL educational platforms, it has significant sociocultural and pedagogical ramifications and makes for effective and enjoyable learning opportunities. Additionally, the technology's potential for incorporation into numerous industries and applications may pave the way for more open and sympathetic user interfaces across various industries, from communication and assistive technology to education and entertainment.

7.4 Future Scope

It is essential to mention the future scope of the project as well as its limitations. The future scope of this real-time hand gesture translator using machine learning is discussed below:

1. Enhanced Sign Differentiation: To tackle the challenge of similar signing patterns, forthcoming research could integrate advanced deep learning techniques such as Siamese networks to enhance the models' ability to differentiate between closely resembling signs, thereby boosting recognition accuracy.

2. Improved Scalability and Multi-User Capability: Enhancing the application's scalability and multi-user support is a crucial consideration. Implementing load balancing, containerization, and cloud-native technologies could empower the application to accommodate numerous users concurrently, fostering a more adaptable and user-centric platform.

3. Including video calling applications: Python programming is used throughout this project. Because of the capabilities of Python, this project can be integrated with video calling programs like Zoom, Microsoft Teams, Google Meet, and others. The ASL users can sustain a fluid conversation with non-ASL or non-sign language users.

4. Improved UI/UX Integration: The integration of UI/UX has been optimised through a continuous dedication to refining the user interface and user experience. This commitment ensures that the application's interaction is intuitive and engaging. By consistently aligning the design elements of the UI/UX with the evolving operational requirements and user expectations, the application has the potential to develop into a tool that seamlessly integrates into the user's workflow and becomes indispensable.

5. Federated Learning to Boost Privacy: Federated learning has the potential to enhance user privacy by enabling model training directly on users' devices. By adopting this approach, valuable insights can be aggregated while preserving the confidentiality of user data, ultimately enhancing data security and reinforcing the ethical foundation of the application.

6. Multimodal integration: The investigation of multimodal integration, which involves the combination of various modalities, including spatial or depth data, can enhance the comprehension of ASL gestures by models. This enhancement would result in improved

accuracy in recognising these gestures and better accommodation of the preferences of different users.

7. Real-Time Collaborative Learning: Envisioning a collaborative framework where individuals collectively contribute to the model's refinement could enhance the application's robustness and inclusivity. Real-time error correction and collaborative learning could offer an engaging and enriching learning experience.

8. Extending to AR/VR Worlds: Users might enjoy immersive and interactive experiences that would transform ASL learning by integrating the application with AR and VR worlds.

In conclusion, although the project flourishes based on success, recognising constraints is a stimulant for progressive development. As we manoeuvre through these limitations, the course in front of us becomes illuminated by inventive resolutions, guaranteeing a more efficient, all-encompassing, and user-centric application for American Sign Language recognition.

8. REFERENCES:

1. Sign Language (no date) education.nationalgeographic.org. Available at: https://education.nationalgeographic.org/resource/sign-language/.

2. Maximum number of languages that can be learned (no date) polyglotclub.com. Available at:https://polyglotclub.com/help/language-learning-tips/maximum-number-of-languages.

3. Communication Aids | AbilityNet (no date) abilitynet.org.uk. Available at: https://abilitynet.org.uk/factsheets/communication-aids-0#:~:text=AAC%20(Augmentative%20and%20Alternative%20Communication)%20aids%20can%20be%20a%20purpose.

4. Rane, P., Bharkad, S. and Tech Student, M. (2021) 'SIGN LANGUAGE RECOGNITION SYSTEM: REVIEW STUDY', 8(10). Available at :https://www.jetir.org/papers/JETIR2110073.pdf (Accessed: 29 September 2023).

5. Rao, P.S. et al. (2023) 'Multiple Languages to Sign Language Using NLTK', International Journal of Scientific Research in Science and Technology, 10(2), pp. 12–17. Available at: https://doi.org/10.32628/IJSRST2310189.

6. Wang, Y., Li, R. and Guan, L. (2023) 'Sign language recognition using MediaPipe'. Available at: https://doi.org/10.1117/12.2674613.

7. Mahyoub, M. et al. (2023) 'Sign Language Recognition using Deep Learning'. Available at: https://doi.org/10.1109/dese58274.2023.10100055.

8. Anudeep, N. (2022) 'SIGN LANGUAGE RECOGNITION', INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT, 06(05). Available at:https://doi.org/10.55041/ijsrem12692.

9. Karayilan, T. and Kilic, O. (2017) 'Sign language recognition', 2017 International Conference on Computer Science and Engineering (UBMK) [Preprint]. Available at: https://doi.org/10.1109/ubmk.2017.8093509.

10. Hou-Hsien, Lee., Chang-Jung, Lee., Chih-Ping, Lo. (2011). Sign language recognition system and method.

11. Cooper, H., Holt, B. and Bowden, R. (2011) 'Sign Language Recognition', Visual Analysis of Humans, pp. 539–562. Available at: https://doi.org/10.1007/978-0-85729-997-0_27.

12. Guo, D. et al. (2021) 'Sign Language Recognition', Springer eBooks, pp. 23–59. Available at: https://doi.org/10.1007/978-3-030-70716-3_2.

13. A Sajeena, Hameed, S. and Sheeba O (2023) 'Design Challenges in Effective Algorithm Development of Sign Language Recognition System', International journal of engineering and advanced technology, 12(3), pp. 69–79. Available at: https://doi.org/10.35940/ijeat.c4030.0212323.

14. Singh, G., Anup Lal Yadav and Sehgal, S.S. (2022) 'Sign language recognition Using Python', 2022 International Conference on Cyber Resilience (ICCR) [Preprint]. Available at: https://doi.org/10.1109/iccr56254.2022.9996001.

15. Patil, Prof.P. et al. (2022) 'Sign Language Recognition System', International Journal for Research in Applied Science and Engineering Technology, 10(5), pp. 1772–1776. Available at: https://doi.org/10.22214/ijraset.2022.42626.

16. Tornay, S., Razavi, M. and Magimai.-Doss, M. (2020) Towards Multilingual Sign Language Recognition, IEEE Xplore. Available at: https://doi.org/10.1109/ICASSP40776.2020.9054631.

17. Lv, Q., Zhang, S. and Wang, Y. (2022) 'Deep Learning Model of Image Classification Using Machine Learning', Advances in Multimedia. Edited by Q. Li, 2022, pp. 1–12. Available at: https://doi.org/10.1155/2022/3351256.

18. Dietmar P. F. Möller (2020) 'Machine Learning and Deep Learning', SpringerBriefs on cyber security systems and networks [Preprint]. Available at: https://doi.org/10.1007/978-3-030-60570-4_5.

19. Janiesch, C., Zschech, P. and Heinrich, K. (2021) ‘Machine learning and deep learning’, Electronic Markets, 31, pp. 685–695. Available at: https://doi.org/10.1007/s12525-021-00475-2.

20. Khan, J. et al. (2023) ‘A Comprehensive Review of Conventional, Machine Leaning, and Deep Learning Models for Groundwater Level (GWL) Forecasting’, Applied Sciences, 13(4), p. 2743. Available at: https://doi.org/10.3390/app13042743.

21. Vamsidhar Enireddy, Kanagachidambaresan, G.R. and Kolla Bhanu Prakash (2021) ‘Application of Machine Learning and Deep Learning’, EAI/Springer Innovations in Communication and Computing, pp. 63–74. Available at: https://doi.org/10.1007/978-3-030-57077-4_8.

22. Yu, Y. (2022) ‘Deep Learning Approaches for Image Classification’. Available at: https://doi.org/10.1145/3573428.3573691.

23. Qiao, Q. (2022) ‘Image Processing Technology Based on Machine Learning’, IEEE Consumer Electronics Magazine, pp. 1–1. Available at: https://doi.org/10.1109/mce.2022.3150659.

24. Liddell, S.K. (2003) ‘American Sign Language as a language’, Cambridge University Press eBooks, pp. 1–5. Available at: https://doi.org/10.1017/cbo9780511615054.002.

25. Sternberg, M. L. A. (1981) American sign language : a comprehensive dictionary / Martin L.A. Sternberg; illustrated by Herbert Rogoff. New York: Harper & Row.

26. Newport, E.L. and Meier, R.P. (1985) ‘The Acquisition of American Sign Language’, Psychology Press eBooks, pp. 881–938. Available at: https://doi.org/10.4324/9781315802541-12.

27. Güney, S. and Erkuş, M. (2021) ‘A real-time approach to recognition of Turkish sign language by using convolutional neural networks’, Neural Computing and Applications [Preprint]. Available at: https://doi.org/10.1007/s00521-021-06664-6.

28. Reddy Karna, S.N. et al. (2021) American Sign Language Static Gesture Recognition using Deep Learning and Computer Vision, IEEE Xplore. Available at: https://doi.org/10.1109/ICOSEC51865.2021.9591845.

29. Al-Qurishi, M., Khalid, T. and Souissi, R. (2021) ‘Deep Learning for Sign Language Recognition: Current Techniques, Benchmarks, and Open Issues’, IEEE Access, 9, pp. 126917–126951. Available at: https://doi.org/10.1109/access.2021.3110912.

30. Sharma, S. and Singh, S. (2021) ‘Vision-based hand gesture recognition using deep learning for the interpretation of sign language’, Expert Systems with Applications, 182, p. 115657. Available at: https://doi.org/10.1016/j.eswa.2021.115657.

31. Mohammad Daim Khan et al. (2021) ‘Real-Time American Sign Language Realization Using Transfer Learning With VGG Architecture’, 2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON) [Preprint]. Available at: https://doi.org/10.1109/gucon50781.2021.9573677.

32. Pisharady, P.K. and Saerbeck, M. (2015) ‘Recent methods and databases in vision-based hand gesture recognition: A review’, Computer Vision and Image Understanding, 141, pp. 152–165. Available at: https://doi.org/10.1016/j.cviu.2015.08.004.

33. Wadhawan, A. and Kumar, P. (2019) ‘Sign Language Recognition Systems: A Decade Systematic Literature Review’, Archives of Computational Methods in Engineering [Preprint]. Available at: https://doi.org/10.1007/s11831-019-09384-2.

34. Cheok, M.J., Omar, Z. and Jaward, M.H. (2017) ‘A review of hand gesture and sign language recognition techniques’, International Journal of Machine Learning and Cybernetics, 10(1), pp. 131–153. Available at: https://doi.org/10.1007/s13042-017-0705-5.

35. Tang, H. et al. (2019) ‘Fast and robust dynamic hand gesture recognition via key frames extraction and feature fusion’, Neurocomputing, 331, pp. 424–433. Available at: https://doi.org/10.1016/j.neucom.2018.11.038.

36. Athitsos, V. et al. (2008) 'The American Sign Language Lexicon Video Dataset', 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops [Preprint]. Available at: https://doi.org/10.1109/cvprw.2008.4563181.

37. Panagiotis Tsinganos et al. (2021) 'Transfer Learning in sEMG-based Gesture Recognition'. Available at: https://doi.org/10.1109/iisa52424.2021.9555555.

38. Hasan, Mahmud., Mashrur, Mahmud, Morshed., Md., Kamrul, Hasan. A deeplearning-based multimodal depth-aware dynamic hand gesture recognition system.. arXiv: Computer Vision and Pattern Recognition, (2021).

39. SHI, Y. et al. (2021) 'Review of dynamic gesture recognition', Virtual Reality & Intelligent Hardware, 3(3), pp. 183–206. Available at: https://doi.org/10.1016/j.vrih.2021.05.001.

40. Li, C. and Wang, B. (2021) 'Hand Gesture Recognition in Complex Background Based on improved Deep Residual Learning network'. Available at: https://doi.org/10.1109/isctis51085.2021.00057.

41. Su, Z. et al. (2021) 'Hand Gesture Recognition Based on sEMG Signal and Convolutional Neural Network', International Journal of Pattern Recognition and Artificial Intelligence, 35(11), pp. 2151012–2151012. Available at: https://doi.org/10.1142/s0218001421510125.

42. Ikram, S. and Dhanda, N. (2021) 'American Sign Language Recognition using Convolutional Neural Network', 2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON) [Preprint]. Available at: https://doi.org/10.1109/gucon50781.2021.9573782.

43. Shin, J. et al. (2021) 'American Sign Language Alphabet Recognition by Extracting Feature from Hand Pose Estimation', Sensors, 21(17), p. 5856. Available at: https://doi.org/10.3390/s21175856.

44. Wattamwar, A. (2021) 'Sign Language Recognition using CNN', International Journal for Research in Applied Science and Engineering Technology, 9(9), pp. 826–830. Available at: https://doi.org/10.22214/ijraset.2021.38058.

45. Federico, Tavella., Aphrodite, Galata., Angelo, Cangelosi. Phonology Recognition in American Sign Language. arXiv: Computation and Language, (2021).

46. Singh, C. (2021) 'Applications and Challenges of Deep Learning in Computer Vision', Lecture Notes in Computer Science [Preprint]. Available at: https://doi.org/10.1007/978-3-030-90885-0_20.

47. Wang, F., Hu, R. and Jin, Y. (2021) 'Research on gesture image recognition method based on transfer learning', Procedia Computer Science, 187, pp. 140–145. Available at: https://doi.org/10.1016/j.procs.2021.04.044.

48. Nair, D., R. Rajapriya and K. Rajeswari (2021) 'Real-Time Electromyographic Hand Gesture Signal Classification Using Machine Learning Algorithm Based on Bispectrum Feature', Lecture notes in electrical engineering [Preprint]. Available at: https://doi.org/10.1007/978-981-33-4866-0_68.

49. Bargellesi, N. et al. (2019) 'A Random Forest-based Approach for Hand Gesture Recognition with Wireless Wearable Motion Capture Sensors', IFAC-PapersOnLine, 52(11), pp. 128–133. Available at: https://doi.org/10.1016/j.ifacol.2019.09.129.

50. Logan, R. et al. (2021) 'Deep Convolutional Neural Networks With Ensemble Learning and Generative Adversarial Networks for Alzheimer's Disease Image Data Classification', Frontiers in Aging Neuroscience, 13. Available at: https://doi.org/10.3389/fnagi.2021.720226.

51. Ljubisa, Stankovic., Danilo, P., Mandic. Convolutional Neural Networks Demystified: A Matched Filtering Perspective Based Tutorial. arXiv: Information Theory, (2021).

52. Ing-Jr, Ding., Meng-Chuan, Hsieh., Xue-Lin, Mo., Sheng-Qi, Wang., Dai-Ru, Wu. Performance Evaluations of Hand Number Gesture Recognition by ConvolutionBased Deep Neural Networks. (2021).1-2. doi: 10.1109/ICCETW52618.2021.9602995.

53. Gesture recognition task guide | MediaPipe (no date) Google for Developers. Available at: https://developers.google.com/mediapipe/solutions/vision/gesture_recognizer#models.