COMPARATIVE STUDY OF DEEP LEARNING MODELS FOR SIGN LANGUAGE RECOGNITION

A PROJECT REPORT

Submitted by

SWETHA MARIYA JOSE

TKM23MCA-2060

to

TKM College of Engineering

Affiliated to

The APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATION

Thangal Kunju Musaliar College of Engineering
Kerala

NOVEMBER 2024

DECLARATION

I undersigned hereby declare that the project report "**Comparative study of deep learning models for sign language recognition"** submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of **Dr. Fousia M Shamsudeen**. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Kollam

11/11/2024 Swetha Mariya Jose

DEPARTMENT OF COMPUTER APPLICATION

TKM COLLEGE OF ENGINEERING

(Government Aided and Autonomous)

KOLLAM – 691005

CERTIFICATE

This is to certify that, the report entitled **COMPARATIVE STUDY OF DEEP LEARNING MODELS FOR SIGN LANGUAGE RECOGNITION** submitted by **Swetha Mariya Jose(TKM23MCA-2060)** to the **APJ Abdul Kalam Technological University** in partial fulfillment of the requirements for the award of the Degree of **Master of Computer Applications** is a bonafide record of the project work carried out by her under my guidance and supervision.This report in any form has not been submitted to any other University or Institute for any purpose.

Internal Supervisors

Mini project Coordinator

ACKNOWLEDGEMENT

First and foremost, I thank GOD almighty and my parents for the success of this project. I owe sincere gratitude and heart full thanks to everyone who shared their precious time and knowledge for the successful completion of my project. I am extremely grateful to Prof. Natheera Beevi M, Head of the Department, Dept of Computer Application, for providing us with the best facilities. I would like to place on record my sincere gratitude to my project guide Dr.Fousia M Shamsudeen, Professor, Department of Computer Application for the guidance and mentorship throughout the course. I would like to thank my project coordinator Prof. Sheera Shamsu, Department of Computer Applications, who motivated me throughout the project.Their contributions have played a crucial role in enhancing the overall learning experience. I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspirations throughout my course of study. I owe thanks to my friends and all others who have directly or indirectly helped me in the successful completion of this project.

ABSTRACT

Sign language recognition plays a crucial role in enabling communication for individuals with hearing impairments. With the growing interest in assistive technologies, deep learning has emerged as a powerful tool for improving the accuracy and efficiency of SLR systems. This paper presents a comparative study of various deep learning architectures—Inception-ResNetV2, ResNet50, and MobileNetV2—for sign language recognition tasks. These models are evaluated based on their performance in terms of accuracy, computational efficiency, and real-time inference capabilities. Inception-ResNetV2, a hybrid of InceptionV3 and ResNet50, benefits from both deeper network layers and enhanced feature extraction. ResNet50, known for its residual learning approach, facilitates better gradient flow and helps mitigate the vanishing gradient problem in deep architectures. MobileNetV2, designed for lightweight models, offers high performance with reduced computational load, making it ideal for mobile or edge-device deployment. The models are trained and tested on a large sign language dataset, and their performance is benchmarked using common evaluation metrics, including classification accuracy and inference time. The results demonstrate that while Inception-ResNetV2 and ResNet50 achieve higher accuracy, MobileNetV2 excels in terms of efficiency, making it a viable solution for real-time applications. The findings suggest that the choice of model should be based on the specific application requirements, balancing between accuracy and computational cost. This study contributes to advancing SLR systems by providing insights into the strengths and trade-offs of different deep learning architectures.

CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Sign language is a vital form of communication for individuals with hearing impairments, allowing them to express thoughts, emotions, and ideas effectively. However, despite its importance, sign language remains underrepresented in many technological domains, especially in automated systems designed to facilitate communication between the hearing and non-hearing communities. One of the key challenges in the development of sign language recognition systems is to accurately translate gestures and signs into text or speech, which requires advanced machine learning and computer vision techniques.

Recent advancements in deep learning have significantly improved the accuracy and efficiency of various image and video processing tasks, including gesture and sign language recognition. Convolutional Neural Networks, particularly deep architectures, have shown great promise in learning complex patterns in visual data, making them ideal candidates for SLR applications. Among these, models like ResNet50, Inception-ResNetV2, and MobileNetV2 have gained attention due to their ability to achieve high performance with varying levels of computational efficiency. ResNet50, for instance, uses residual connections to improve learning in very deep networks, while Inception-ResNetV2 combines the strengths of both InceptionV3 and ResNet50 architectures for more accurate feature extraction. On the other hand, MobileNetV2 offers a lightweight architecture, suitable for real-time applications on resource-constrained devices such as mobile phones and edge devices.

This report investigates the performance of these three deep learning architectures— Inception-ResNetV2, ResNet50, and MobileNetV2—in the context of sign language recognition. By evaluating these models on a sign language dataset, we aim to understand their strengths and limitations, providing a detailed comparison based on classification accuracy, computational efficiency, and real-time inference capabilities. The results of this study are intended to contribute to the design of more effective and accessible SLR systems, catering to different application needs, from high-accuracy systems to those optimized for use on mobile and embedded platforms.

1.1 EXISTING SYSTEM

The existing systems for sign language recognition have evolved significantly over time, leveraging both traditional and modern machine learning techniques. Early approaches were based on rule-based systems that relied heavily on manual feature extraction, such as hand shapes, colour detection, and motion tracking. These systems used simple algorithms like Support Vector Machines , K-Nearest Neighbours , and Hidden Markov Models for classification tasks. However, such methods had limitations in terms of scalability and adaptability to real-world conditions. These approaches were often sensitive to environmental variables such as lighting, background noise, and variations in the signer's gesture styles. Furthermore, they struggled to recognize continuous sign language sequences that required capturing temporal dependencies between gestures.

With the rise of deep learning, SLR systems began to incorporate Convolutional Neural Networks for feature extraction and Recurrent Neural Networks , particularly Long Short- Term Memory networks, for modeling the temporal dynamics of sign language gestures. These deep learning models significantly improved the accuracy of gesture recognition, especially for static hand shapes. Hybrid models, such as CNN-LSTM or 3D CNNs, were developed to handle dynamic sequences, enabling the recognition of continuous sign language. However, despite these advances, challenges remain, such as the need for large annotated datasets, real-time recognition capabilities, and generalization across different sign languages and dialects.

In addition to CNNs and RNNs, recent work has explored the use of transfer learning with pre-trained models like ResNet50, Inception-ResNetV2, and MobileNetV2. These models allow for more efficient feature extraction and faster training, reducing the need for large labeled datasets. While ResNet50 and Inception-ResNetV2 are highly effective for deep, complex features, MobileNetV2 provides a more computationally efficient option, making it suitable for deployment on mobile and embedded devices. However, existing systems still face issues related to real-time processing, handling occlusions, and accurately recognizing signs in diverse environmental conditions.

1.2 PROBLEM STATEMENT

Sign language is a critical means of communication for individuals with hearing impairments, yet it remains underrepresented in mainstream technology. Despite advancements in assistive

technologies, current Sign Language Recognition systems face several significant challenges that limit their effectiveness and widespread adoption. Existing systems, although improved through machine learning and deep learning approaches, still struggle with issues such as real-time processing, accuracy across diverse sign languages, and efficient deployment on resource-constrained devices.

Many traditional SLR models rely on static gesture recognition, failing to effectively capture temporal dynamics inherent in continuous sign language communication. This leads to a limited ability to recognize dynamic gestures and sequences that involve motion, such as transitioning between signs or facial expressions. Moreover, most models are highly sensitive to environmental variables such as lighting conditions, background noise, and hand occlusions, resulting in reduced accuracy in real-world scenarios. Furthermore, the lack of large and diverse annotated datasets means that models are often overfitted to specific sign languages, failing to generalize well across different signing styles or regional variations.

Additionally, many existing SLR systems are computationally expensive, which makes them unsuitable for deployment on mobile devices or in real-time applications. As a result, there is a growing need for an SLR system that can achieve both high accuracy and real-time performance, while being computationally efficient enough for use on mobile and embedded platforms.

1.3 PROPOSED SYSTEM

The proposed system aims to develop a robust, scalable, and real-time sign language recognition solution that leverages advanced deep learning architectures to overcome the limitations of existing systems. The system will address core challenges in sign language recognition, including accuracy across diverse sign languages, real-time recognition, and efficiency for deployment on resource-constrained devices.

To achieve these objectives, the system will incorporate several key components. At the core of the system will be deep learning models such as Inception-ResNetV2, ResNet50, and

MobileNetV2. These models will be evaluated for their performance in both image-based and video-based sign language recognition. Inception-ResNetV2 combines the strengths of the Inception architecture, which uses multiple convolutional filters at different scales, and ResNet50's residual connections, allowing the system to capture both deep and fine-grained features from sign language gestures. This hybrid architecture is expected to provide high accuracy for complex sign language recognition tasks. ResNet50's residual connections make it particularly well-suited for deeper networks, enabling the model to efficiently learn complex features from large datasets while avoiding issues like vanishing gradients. MobileNetV2, designed specifically for mobile and embedded devices, will be used to achieve real-time, efficient recognition. MobileNetV2 uses depth-wise separable convolutions, significantly reducing the number of parameters and making the system computationally efficient while maintaining strong performance.

The system will be designed to be scalable, enabling the addition of new sign languages or gestures with minimal retraining. This flexibility will allow the system to evolve as more data becomes available, ensuring continuous improvement in accuracy and functionality. By adopting a modular architecture, the system can easily integrate new sign languages and expand its recognition capabilities over time.

1.4 OBJECTIVES

1. **To Develop a Real-Time Sign Language Recognition Model**:

Build a robust model that can accurately recognize hand gestures representing the American Sign Language alphabet (A-Z) from real-time video input.The system aims to bridge communication gaps between deaf and hard-of-hearing individuals and non-signers.

2. **To Combine Individual Letter Predictions into Words**:

Develop a mechanism that processes continuous sign language gestures and combines individual letter predictions into complete words. Track the recognized letters in real-time and group them into words for fluent and meaningful communication.

3. **To Ensure Robustness through a Diverse Dataset**:

Train the model on a manually collected, diverse dataset that includes variations in hand shapes, backgrounds, lighting conditions, and other environmental factors.Ensure the system can generalize well to different real-world conditions, including variations in signing styles and camera qualities.

4. **To Develop a User-Friendly Graphical User Interface (GUI)**:

Create a GUI that displays both the recognized letters and complete words for easy user understanding. Include confidence scores for predictions, providing users with real-time feedback on the system's accuracy.

5. **To Evaluate Model Performance**:

Perform extensive evaluation to compare the custom CNN model with pretrained models like VGG16 and InceptionV3.Assess accuracy, real-time performance, and reliability across different devices and environments to determine the most effective model for sign language recognition.

Chapter 2

LITERATURE SURVEY

A literature survey, also known as a literature review, involves analyzing scholarly sources related to a particular subject. Examining the available literature, it provides a comprehensive overview of the state of the field, allowing you to identify relevant theories, approaches, and gaps in the existing body of knowledge. When conducting a literature review from an audit perspective, the main focus is on evaluating the relevant literature. This process covers infor- mation that has been published in a specific field of study and sometimes includes information published within a specific time frame.

2.1 PURPOSE OF LITERATURE SURVEY

1. It gives readers easy access to research on a particular topic by selecting high quality articles or studies that are relevant, meaningful, important and valid and summarising them into one complete report.

2. It provides an excellent starting point for researchers beginning to do research in a new area by forcing them to summarise, evaluate, and compare original research in that specific area.

3. It ensures that researchers do not duplicate work that has already been done.
4. It can provide clues as to where future research is heading or recommend areas on which to focus.

5. It highlights the key findings.

2.2 RELATED WORKS

- In their study , Ahmed Kasapbasi , Ahmed Eltayeb Ahmed Elbushra , Omar Al- Hardanee and Arif Yilmaz created a dataset along with a sign language interface system based on Convolutional Neural Networks (CNN) to translate gestures from sign language and hand shapes into natural language. The neural network developed in this project is a CNN that improves the accuracy of recognizing the American Sign Language alphabet .. This study introduces a novel dataset of the American Sign Language alphabet, considering various factors such as lighting conditions and distances. They also compared their dataset with

two other datasets from previous research.The proposed CNN model achieved an accuracy of 99.38% with outstanding prediction and a minimal loss of 0.0250. [1]

- In a recent study,Sundar B and Bagyammal T introduced a vision-based system designed to recognize American Sign Language (ASL) alphabets by utilising Google's MediaPipe combined with Long Short-Term Memory (LSTM) networks They created a custom dataset for the experimental study.The dataset encompassed 26 ASL alphabets, with every gesture being recorded over 30 frames, thereby improving the robustness of the model. Unlike conventional methods relying on geometric or shape- based features, the proposed system leverages MediaPipe's efficient 3D hand tracking to capture 21 hand landmarks from a single image frame, enhancing real-time accuracy.This method achieved an impressive 99% accuracy by taking advantage of
  MediaPipe's 3D hand landmark tracking, allowing for accurate, real-time gesture detection for both static and dynamic signs.[2]

- The authors Prof. Mrs. Maheshwari Chitampalli, Dnyaneshwari Takalkar, Gaytri Pillai, Pradnya Gaykar and Sanya Khubchandani concentrates on creating a computer vision system intended for recognizing sign language gestures, with the goal of reducing communication barriers for individuals who are deaf or hard-of-hearing. The proposed system captures images of an individual signing through a camera, processes the visual frames to identify and interpret hand gestures, and converts them into text output.The main phases include data gathering, preprocessing, gesture segmentation, and feature extraction, with a Convolutional Neural Network (CNN) selected as the primary model for classification. The system underwent testing using an independent dataset to verify its accuracy, achieving an accuracy of 95%.This research highlights the significance of hand detection and tracking, feature extraction, and sign language translation, emphasizing the potential for automated sign language recognition systems to improve accessibility for non-signers and foster inclusive communication technologies. [3]
- Hope Orovwode , Ibukun Deborah Oduntan and John Abubakar introduced a CNN- based approach for identifying static American Sign Language (ASL) alphabet signs, fulfilling a crucial demand for improved communication accessibility for people with hearing and speech disabilities. Using a dataset containing 44,654 images obtained through a HandDetector module, the research segmented the data into training, validation, and test sets, achieving outstanding accuracy in all phases. The model, designed with three convolutional layers along with a SoftMax output layer, was trained using the Adam

optimizer and a categorical cross-entropy loss function, reaching a remarkable 99.86% training accuracy, 99.94% validation accuracy, and 94.68% test accuracy. This system surpassed earlier models, signifying an important advancement in the realm of sign language recognition, with the potential to close communication gaps for the Deaf and mute communities.[4]

- Bader Alsharif , Easa Alalwany , Mohammad Ilyas proposed a highly accurate, real- time system for recognizing American Sign Language (ASL), intending to improve communication with the deaf community. By combining the YOLOv8 deep learning model with MediaPipe's hand landmark detection, the study created a strong classification system capable of identifying ASL gestures and converting them into text. MediaPipe identifies 21 key landmarks on the hand, which improves the YOLOv8 model's accuracy by capturing intricate hand positions. The authors trained the system using a large dataset of over 29,820 annotated ASL images, achieving an exceptional accuracy of 96.3% for the 26 letters of the alphabet, surpassing previous methods in recognition accuracy, class loss, and bounding box loss. This research highlights the potential of this application for facilitating real-time ASL communication, representing a significant leap forward in assistive technology. [5]

- Yulius Obi , Kent Samuel Claudio , Vetri Marvel Budiman , Said Achmad,, Aditya Kurniawan utilized the American Sign Language (ASL) Hand Sign Dataset,featuring 24 classes sourced from Kaggle with a Gaussian blur filter applied, and supplemented by additional classes from Nikhil Gupta to fulfill the 27-class requirement. This dataset consists of 30,526 images for training purposes and 8,958 images for testing, which is significantly more extensive than other datasets used for similar projects. The application reached an outstanding accuracy of 96.3%, underscoring the effectiveness of CNN-based systems for real-time sign language recognition and the advantages of larger, well-organized datasets in enhancing classification performance[6]

2.3 GAPS IDENTIFIED

- **Challenges of Data Scarcity**: Sign language recognition relies heavily on diverse and extensive datasets. However, assembling large datasets that cover various gestures, movements, and signers has proven challenging. Sign languages involve nuanced hand shapes, facial expressions, and body positions, making data collection labor- intensive

and costly. Furthermore, due to privacy concerns, ethical considerations, and the complex logistics of recording such data, only a few publicly accessible datasets exist, which often lack the necessary diversity in terms of signers, backgrounds, and lighting conditions.

- **Limitations of Custom CNNs**: Many studies in sign language recognition have used custom convolutional neural network (CNN) architectures specifically designed for the task. While these can be tailored to certain dataset characteristics, they often lack the depth and training advantage of pretrained models. Custom models tend to underperform compared to more advanced architectures when trained on limited data, as they may not have the same capacity to learn complex, hierarchical representations.

- **Static vs. Dynamic Gesture Limitations**: Many studies simplify the problem by focusing on static images representing individual signs rather than sequences capturing dynamic gestures. Static images may capture only a limited perspective of the sign, often missing essential information related to movement and transition between signs, which are fundamental in real-world signing.

Chapter 3
METHODOLOGY

Sign language recognition is a highly complex task due to the intricate nature of hand gestures, which often involve subtle differences in finger positions, hand orientations, and movements. Achieving accurate recognition requires careful attention to several components that must function together seamlessly: collecting and preparing a robust dataset, training sophisticated machine learning models, and providing an accessible, user-friendly interface for end-users. Each component plays a critical role in the overall system's effectiveness. The quality and diversity of the data directly impact the model's ability to generalize, while the models themselves need to be powerful enough to distinguish between nuanced gestures. Finally, the interface enables users, even those unfamiliar with machine learning, to leverage these advanced tools effortlessly, making the system practical for real-world use.

To achieve high recognition accuracy, three advanced convolutional neural network (CNN) architectures were selected: InceptionResNetV2, ResNet50, and MobileNetV2. Each of these models was chosen for specific strengths in handling image-based recognition tasks. InceptionResNetV2 combines the deep layers of the ResNet family with the Inception module's diverse convolutional filters, making it adept at capturing complex spatial features in images. ResNet50 is known for its efficient training through skip connections, which help prevent the vanishing gradient problem, making it effective even with limited datasets. MobileNetV2, a lightweight model optimized for mobile and low-power devices, was chosen for its efficient performance and suitability for real-time applications, which makes it particularly useful for sign language recognition where responsiveness is crucial. By employing these three models, the system benefits from both accuracy and efficiency across a variety of device capabilities.

The dataset used to train these models underwent extensive preprocessing and augmentation to ensure that the models could generalize well across various lighting conditions, backgrounds, and users. Preprocessing steps included resizing images to a standard input size, normalizing pixel values, and occasionally converting images to grayscale to reduce computational load without sacrificing detail. Data augmentation techniques, such as random rotations, flipping, and brightness adjustments, were applied to artificially expand the dataset and make the models more resilient to variations in real-world conditions. This approach helps the models avoid overfitting

on training data, thereby enhancing their robustness when deployed, especially for recognizing gestures from different angles or in environments with changing lighting.

For user accessibility, the system's interface was built using Tkinter, a Python library well- suited for developing graphical user interfaces (GUIs). Tkinter enables the creation of a simple yet

functional desktop interface that allows users to upload images or use a live video feed for real-time gesture recognition. Through the interface, users can interact with the models without needing to understand the underlying machine learning processes. The live video feed feature allows for a seamless and responsive experience, displaying recognition results in real time, making it ideal for applications that demand quick and accurate feedback.

By leveraging Tkinter's straightforward design capabilities, the app presents a clear and intuitive layout, allowing users to focus on interpreting sign language gestures rather than navigating complex software, thus making advanced machine learning technology accessible to a wider audience .

3.1 BLOCK DIAGRAM

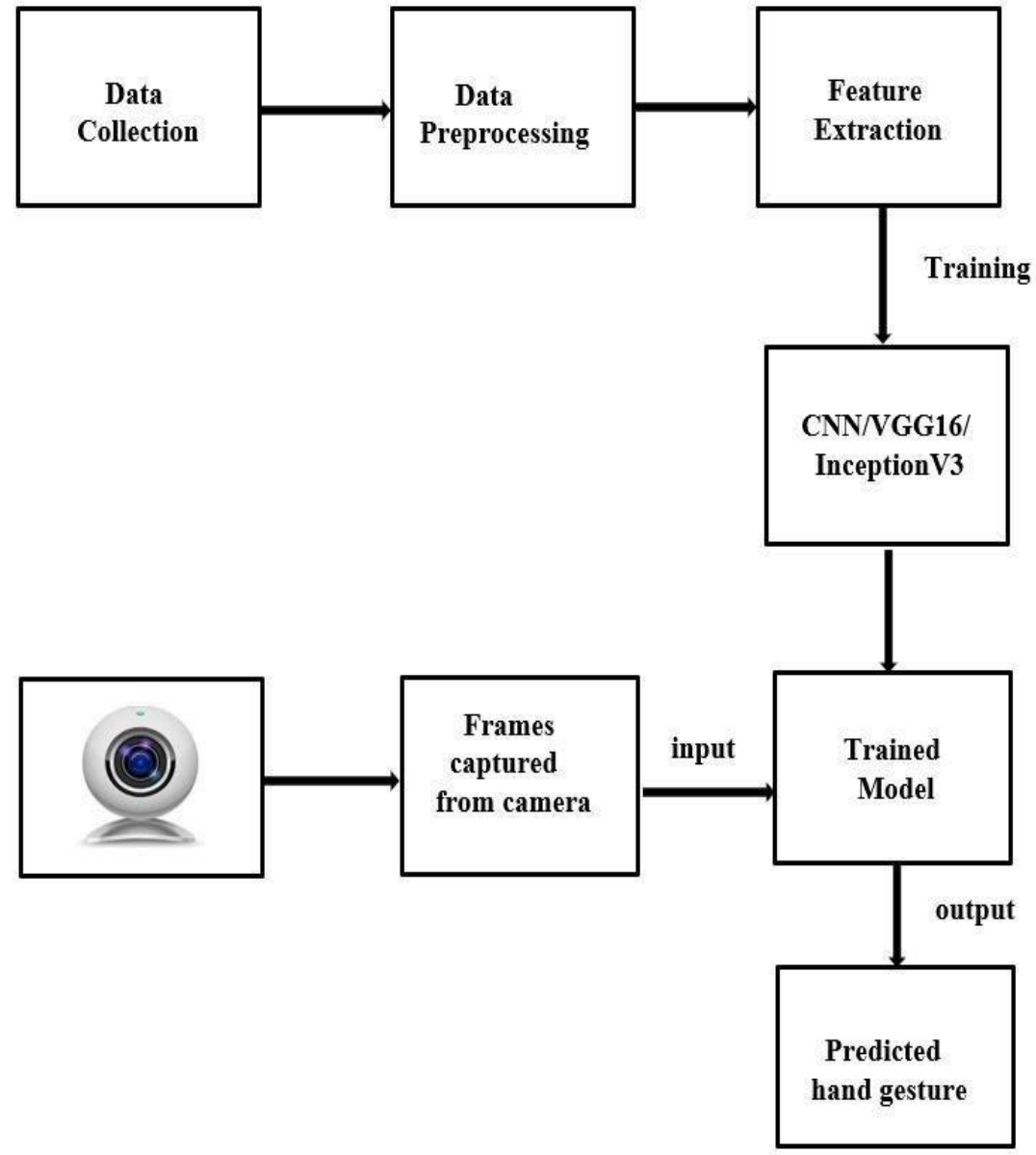

Figure 1

3.1.1 Data Collection

For our sign language recognition project, data collection focused on gathering diverse and high-quality images of various sign gestures to train machine learning models like ResNet50, InceptionResNetV2, and MobileNetV2.

The dataset includes gestures representing the American Sign Language (ASL) alphabet and common phrases, covering a wide range of hand signs essential for comprehensive sign language recognition. We captured images and videos with consistent camera angles and plain backgrounds, incorporating diverse hand positions, lighting conditions, and angles to ensure robustness using OpenCV.

The dataset used in this project is a dataset, which contains over 52,000 images of alpabhets ,with each alphabet having 2000 samples each.

Figure 2

Figure 3

3.1.2 Image Preprocessing

Image preprocessing played a crucial role in preparing the dataset for effective training of MobileNetV2, ResNet50, and InceptionResNetV2 in the sign language recognition project. To ensure compatibility with each model's architecture, all images were first resized to the specified input dimensions required by the respective networks. This resizing standardizes the input, allowing models to process the images consistently and maintain computational efficiency. Following resizing, several data augmentation techniques were applied, such as horizontal flipping, brightness and contrast adjustments, and random rotations. These augmentations help the models become resilient to variations in hand orientation, lighting, and distance, which are often encountered in real-world conditions. These transformations effectively increase the diversity of the dataset, helping to mimic a wide range of environmental factors that might affect gesture recognition accuracy.

Additionally, random cropping was performed on selected images to simulate slight variations in framing, further enhancing the models' ability to generalize by focusing them on key features within different portions of the image. The dataset was then divided into training, validation, and test sets to facilitate rigorous model evaluation. The training set allowed the models to learn gesture patterns, while the validation set was used to fine-tune hyperparameters, ensuring the models did not overfit to the training data. The test set provided a final, unbiased evaluation of the models' performance. This comprehensive preprocessing pipeline ensured that the images were optimized for effective training, ultimately improving each model’s robustness, accuracy, and generalization when recognizing gestures in diverse, real-world settings.

3.1.3 Model Architecture Design

1. ResNet50: ResNet50 is a 50-layer deep convolutional neural network that has shown outstanding performance on a variety of image recognition tasks. It is specificaly designed to handle complex features, making it well-suited for distinguishing subtle variations in hand gestures. The **residual connections** in ResNet50 allow very deep networks to avoid vanishing gradient issues by learning identity mappings, which are critical for accurately capturing intricate details in hand movements and gestures

2. Inception ResNetV2: InceptionResNetV2 merges the **Inception modules** which use multiple filter sizes for capturing spatial hierarchies with **residual connections** from ResNet which help in training very deep networks by bypassing certain layers. This combination makes InceptionResNetV2 effective in extracting detailed features while maintaining high accuracy, even for subtle hand gestures with fine differences.

3. MobileNetV2: MobileNetV2 is a **lightweight convolutional neural network** designed for mobile and embedded devices. It uses **depthwise separable convolutions** and **inverted residuals** with linear bottlenecks, significantly reducing the number of parameters and computation while maintaining high accuracy. This efficiency makes MobileNetV2 particularly suitable for real-time sign language recognition tasks where speed is critical.

3.1.4 Model Training

For the sign language recognition project, model training involved several key steps to fine- tune deep learning models such as MobileNetV2, ResNet50, and InceptionResNetV2 for accurate gesture recognition. We used **transfer learning** to take advantage of the pretrained weights from ImageNet, which significantly reduced training time and improved performance.

The models's final layers were replaced with a fully connected layer corresponding to the number of gestures in the dataset, such as 26 for the ASL alphabet. The dataset was split into training, validation, and test sets, with **data augmentation** techniques applied during preprocessing to enhance the model's ability to generalize across different hand positions, lighting, and orientations.

We monitored the models' performance on the validation set, using metrics like accuracy, precision, and recall to identify optimal hyperparameters such as learning rate and batch size. **Early stopping** was implemented to prevent overfitting, halting training if the validation loss ceased to improve. After training, the models were evaluated on the test set to assess generalization, and a **confusion matrix** was used to identify common misclassifications. The best-performing model, based on test set accuracy, was selected for deployment or further real-world testing. This process ensured that the trained model was both robust and efficient in recognizing sign language gestures.

3.1.4 Model Evaluation

Model evaluation for the sign language recognition project is essential to assess the performance of the trained models and ensure they generalize well to unseen data. After the training phase, the models (MobileNetV2, ResNet50, and InceptionResNetV2) were evaluated using various performance metrics to quantify their ability to correctly recognize sign language gestures.

First, the models were tested on the **test set**, which had been set aside during training to prevent overfitting and ensure that the evaluation was based on unseen data. The **accuracy** metric was the primary evaluation criterion, representing the proportion of correctly classified gestures out of the total number of gestures in the test set.

Highest accuracy was shown by InceptionResNetV2 of 97.95%,ResNet50 showed an accuracy of 97.17% .Lowest accuracy was of MobileNetV2 and it was 63%

3.1.5 Deployment

Deployment of the sign language recognition model is the final step in making the trained model usable for real-world applications. After selecting the best-performing model based on evaluation metrics like accuracy, the model needs to be integrated into a system where it can recognize sign language gestures in real-time.

A TKinter based web application was developed allowing users to interact with a simple interface which is then processed by the trained model-InceptionResnetV2 for recognising sign languages.

The video is captured with the help of OpenCV, the sign from the captured video is analysed and output is shown to the users,the letters are combined to form meaningful sentences.

3.2 SOFTWARE REQUIREMENTS AND SPECIFICATIONS

3.2.1 Operating System

The project is designed to run on modern operating systems like Windows 10/11 or Linux (Ubuntu). This ensures compatibility across different development environments and allows users to work with the software on their preferred platform, whether for local development or deployment. The system's flexibility supports a broad range of user preferences and makes it adaptable to various hardware configurations.

3.2.2 Python 3.11

Python is a high-level, general-purpose programming language known for its simplicity and readability, making it a popular choice for data science, artificial intelligence, and web development. The version used in this project is Python 3.11, which offers several improvements over previous versions, including enhanced performance and new features for developers. Python’s versatility and rich ecosystem of libraries like TensorFlow, NumPy, and Pandas make it ideal for implementing complex machine learning and computer vision tasks such as those required in this project.

3.2.3 Visual Studio Code (VS Code)

The project utilizes Visual Studio Code (VSCode), a free, open-source code editor developed by Microsoft. VSCode is lightweight yet powerful, supporting a wide range of programming languages, including Python. Key features include:

- Built-in Git support for version control.
- Debugging tools to help identify and resolve code issues.
- An extension marketplace to enhance functionality with additional tools.
- Integrated terminal, which facilitates executing scripts directly from the editor.
- Code completion and IntelliSense for better productivity and reduced coding errors. These features make VSCode an ideal choice for development, enabling developers to efficiently manage their codebase, troubleshoot issues, and streamline their workflow.

3.2.4 Jupyter Notebook

Jupyter Notebook provides an interactive development environment that is widely used in data science and machine learning. It allows you to create and share documents that contain live code, equations, visualizations, and narrative text. In your project, Jupyter Notebooks are used for exploring the dataset, experimenting with preprocessing techniques, and visualizing results. It's also useful for running and documenting the iterative process of model training.

Key Features:

- Cell-based execution for modular code.
- Supports rich text, LaTeX, and visualizations within the notebooks. • Easy integration with Python libraries such as NumPy, Pandas, and Matplotlib.

3.2.5 Libraries

The following libraries are essential for the successful implementation of the project:

- **TensorFlow/Keras**: These libraries are crucial for building and training deep learning models like CNN, AlexNet, and ResNet50. TensorFlow provides powerful tools for model construction, training, and evaluation, while Keras offers high-level APIs for creating neural networks efficiently.
- **OpenCV**: Used for image processing tasks like reading, resizing, and augmenting images. It's especially useful for preparing the dataset before feeding it into the models.

- **NumPy**: This library is fundamental for handling numerical data, such as image arrays, and performing various operations like resizing and normalizing the image data.

- **Matplotlib**: Employed for visualizing model training progress, such as plotting training and validation accuracy and loss curves.

- **Scikit-learn**: Provides tools for splitting the dataset into training and testing sets and evaluating model performance using metrics like accuracy.

- **TKinter: Tkinter** is a popular Python library used for building graphical user interfaces (GUIs). It provides a simple and efficient way to create desktop applications with a visual interface. .

3.2.6 Google Chrome

Google Chrome is a widely-used web browser known for its speed, security, and performance. It is essential for testing and deploying the web application developed in this project. Chrome provides excellent developer tools, including the JavaScript console, network monitoring, and performance analysis features. These tools assist developers in debugging issues with the web interface, optimizing performance, and ensuring smooth deployment of the application.
Additionally, Chrome's extension support and compatibility with modern web standards make it a suitable browser for testing the web application's functionality.

Chapter 4

RESULT AND DISCUSSIONS

After training the models on the sign language dataset, we evaluated them using various metrics such as **accuracy**, and confusion matrix. These metrics provide a comprehensive understanding of the models' ability to recognize gestures from real-time video feeds.

After training the models, we achieved an accuracy of 97% across the different architectures (ResNet50 and InceptionResNetV2) on the test set.MobileNetV2 showed low accuracy of 63%. Among the three, **InceptionResNetV2** consistently provided the highest accuracy due to its combination of deep layers and residual connections, which allow it to learn more complex features from the dataset.

A TKinter based web application was developed allowing users to interact with a simple interface which is then processed by the trained model-InceptionResnetV2 for recognising sign languages.The video is captured with the help of OpenCV, the sign from the captured video is analysed and output is shown to the users,the letters are combined to form meaningful sentences.

4.1 MODEL PERFORMANCE EVALUATION

The **accuracy** metric represents the proportion of correct predictions made by the model out of the total predictions. For this project, accuracy was the primary indicator of the model's overall performance.

- **ResNet50**: It showed an accuracy of 97.17% and was close to accuracy of InceptionResNetV2,it did not showed signs of overfitting. It completed in 30 epochs in overall 50 epochs. **Overfitting** occurs when a machine learning model learns not only the underlying patterns in the training data but also the noise and random fluctuations. This results in the model performing very well on the training data but poorly on new, unseen data because it has become too specialized to the training set.
  Essentially, the model "memorizes" the training data instead of generalizing to broader patterns, leading to poor performance in real-world scenarios.

- **InceptionResnetV2**:It showed the highest accuracy of 97.95%,this model was used for building user interface due to its high performance.It completed in 26 epochs in overall 50 epochs.

- **MobileNetV2**:It showed the lowest accuracy of 63%.It completed in 7 epochs in overall 50 epochs.

Model	Accuracy
Inception-ResNetV2	97.95%
ResNet50	97.17%
MobileNetV2	63.10%

4.1.1 Evaluation Curves

1. ResNet50 validation and training accuracy and loss graphs:

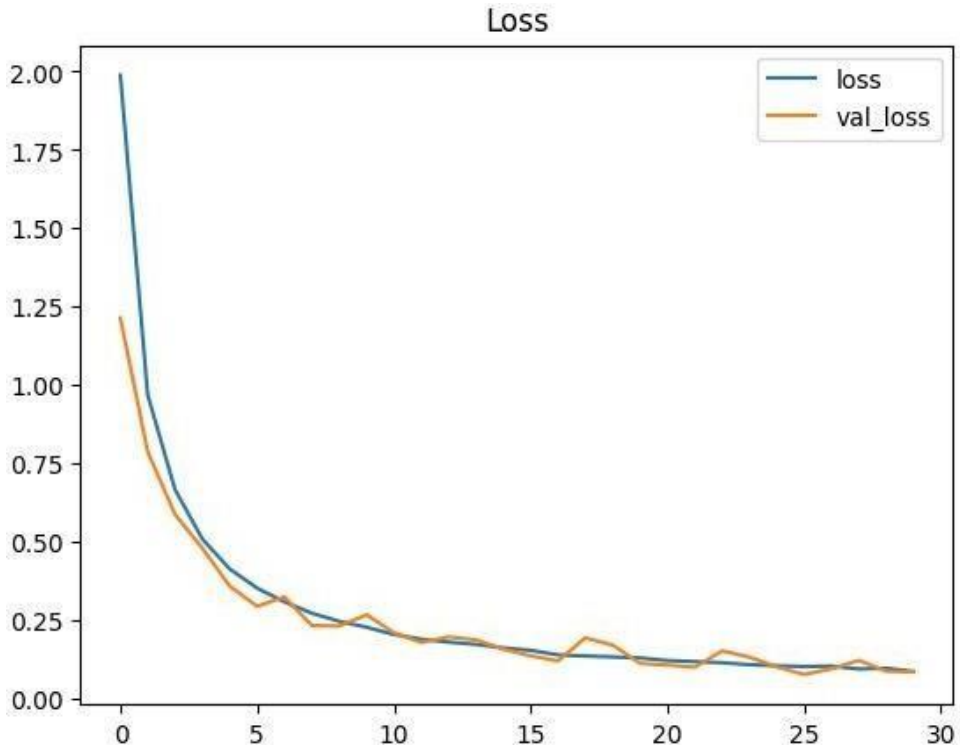

Figure 4

This plot shows the training loss and validation loss over the 30 number of epochs.Both losses decrease steadily and converge near zero, which is ideal as it indicates that the model's predictions are becoming increasingly accurate on both the training and validation sets.The close alignment between training and validation losses suggests that the model is not overfitting ;the loss of unseen data issimilar to that of training data.

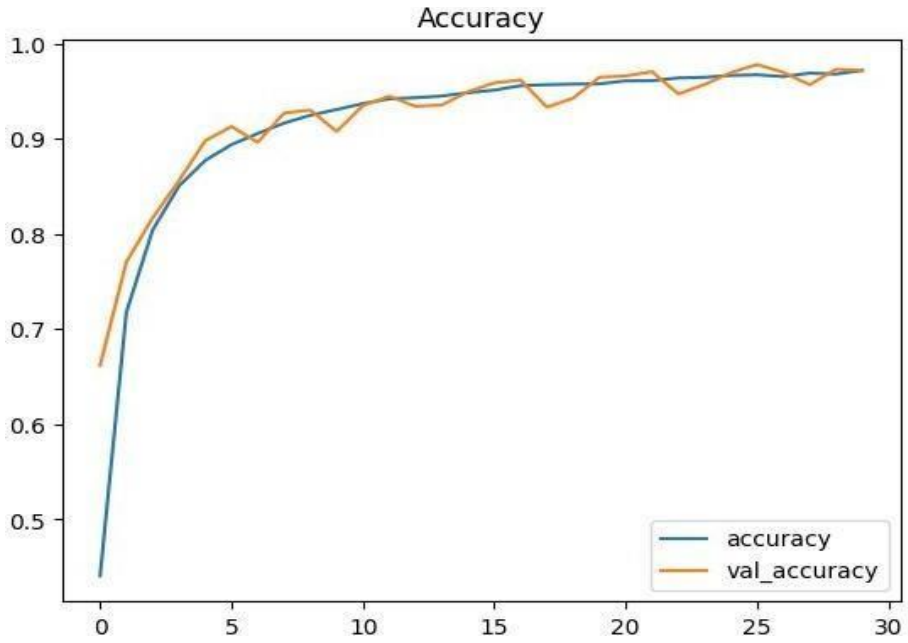

Figure 5

This plot shows the training accuracy and validation accuracy over 30 epochs.The training and validation accuracy curves increase rapidly in the early epochs and stabilize close to 1.0, indicating that the model is learning well and achieving high accuracy on both training and validation sets.Since both curves follow similar trajectories without significant divergence, the model is likely generalizing well without severe overfitting.

2.InceptionResNetV2 traning and validation,accuracy and loss graphs:

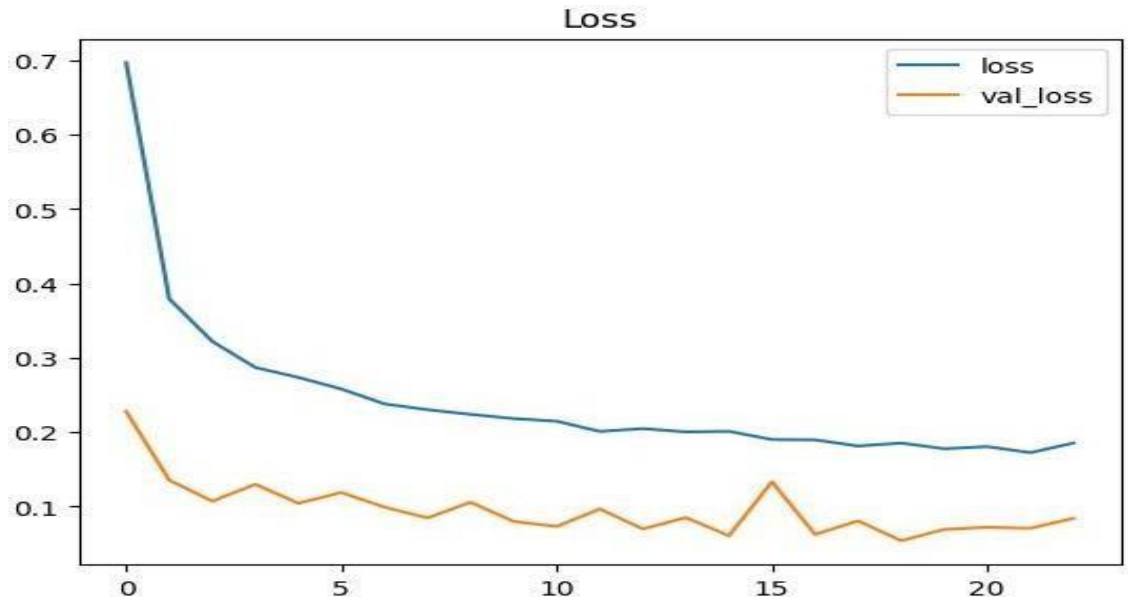

Figure 6

This plot likely represents a secondary model training process or a model trained on a different dataset, given the different scale on the y-axis (loss values are lower initially).The training and validation loss both decrease, but the validation loss fluctuates more than in the previous plot.There’s a slight gap between training and validation loss, which could imply some minor overfitting, although it's not extreme.

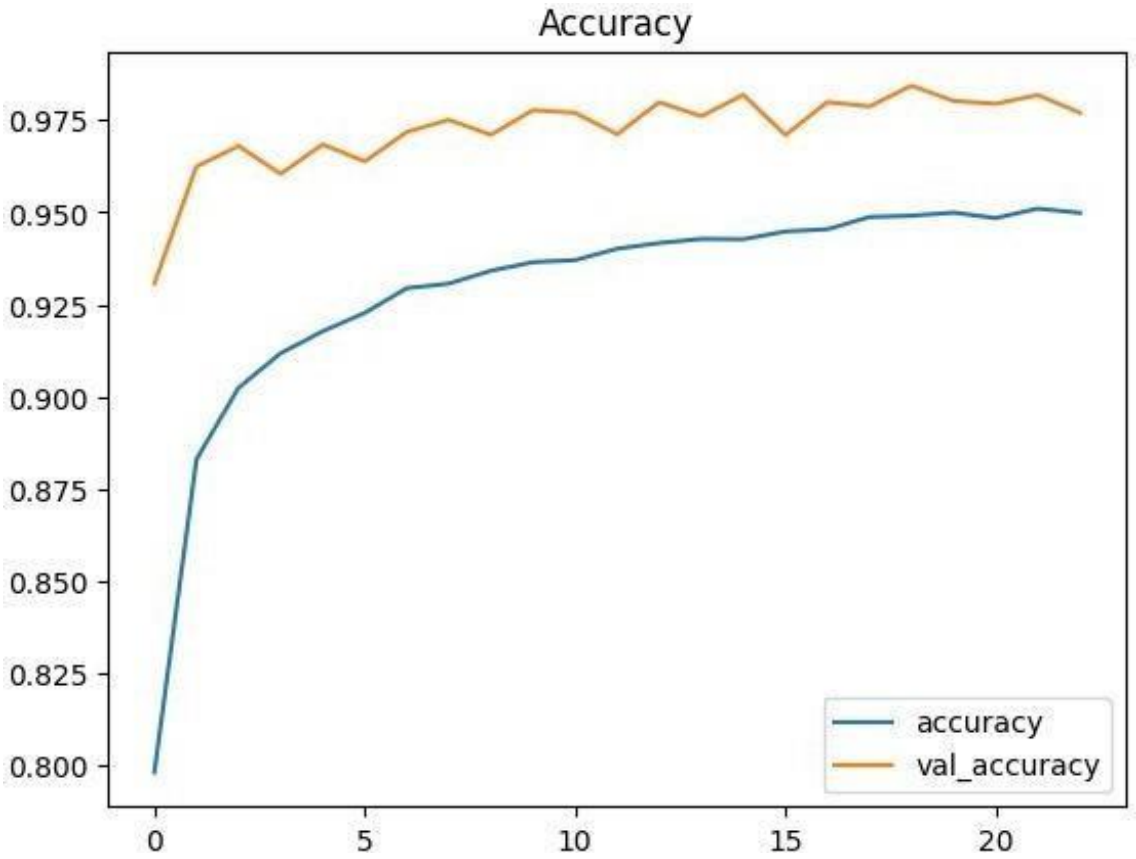

Figure 7

This plot shows the training accuracy and validation accuracy over 26 epochs.The training and validation accuracy curves increase rapidly in the early epochs and stabilize close to 1.0, indicating that the model is learning well and achieving high accuracy on both training and validation sets.Since both curves follow similar trajectories without significant divergence, the model is likely generalizing well without severe overfitting.

3.MobileNetV2 training and validation accuracy and loss graphs:

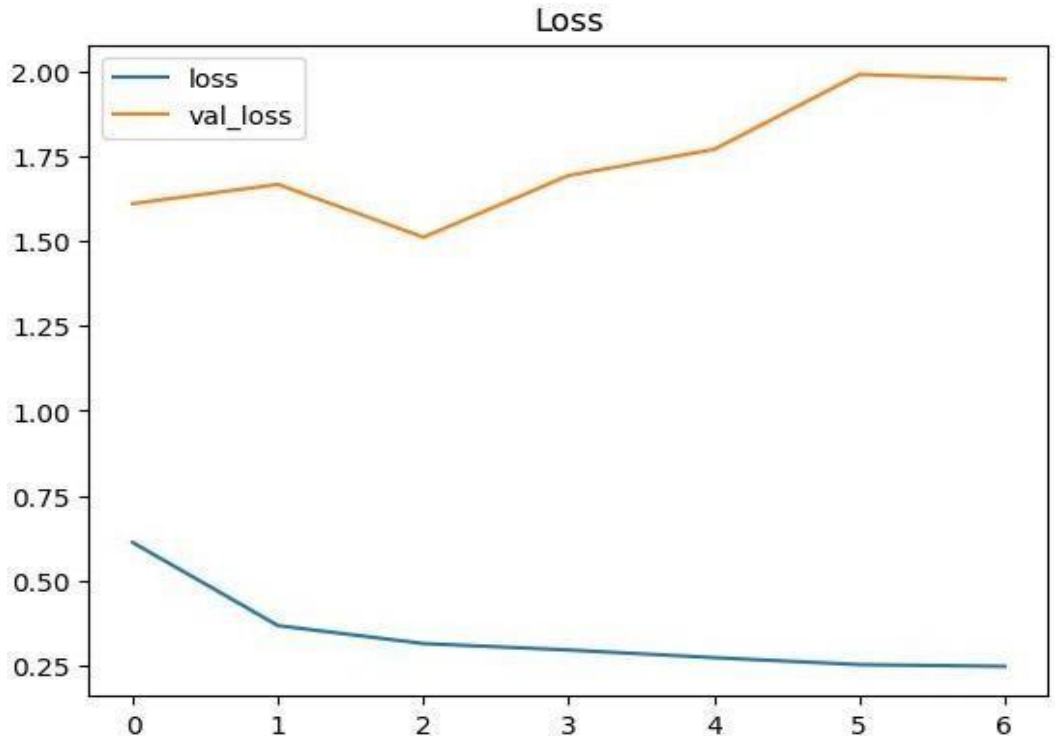

Figure 8

In the loss graph, the training loss (blue line) shows a steady decrease over the epochs, indicating that the model is effectively minimizing error on the training data. However, the validation loss (orange line) follows an opposite trend, gradually increasing over time. This divergence suggests that the model may be overfitting, as it is learning patterns specific to the training data rather than generalizable features. Overfitting causes the model to perform poorly on new, unseen data, which is why the validation loss rises while training loss continues to drop.

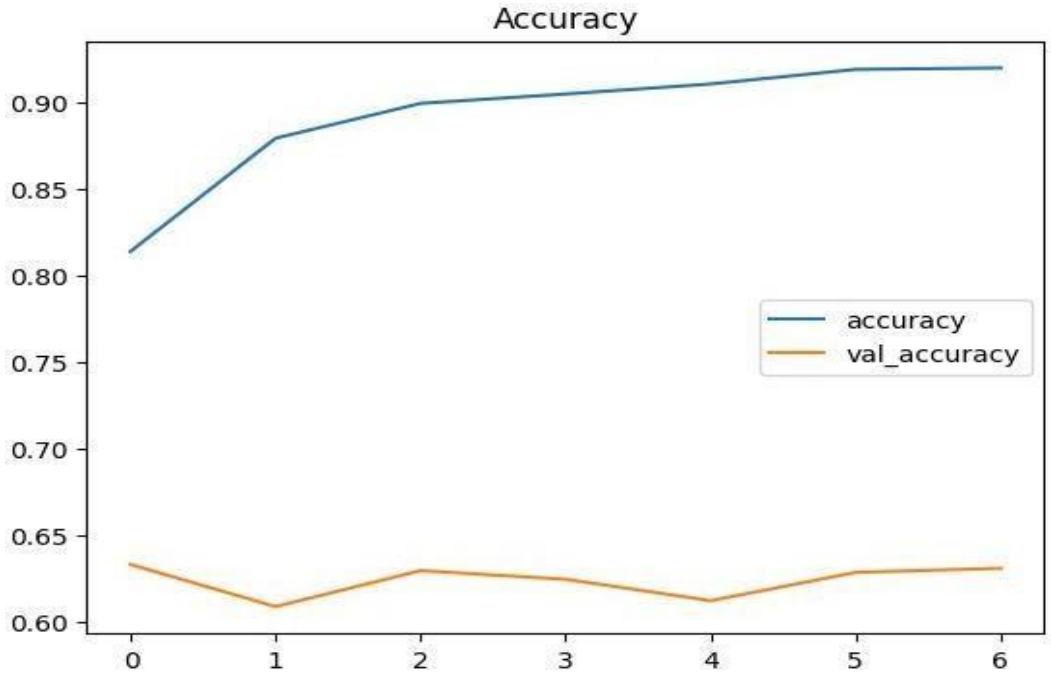

Figure 9

In the accuracy graph, the training accuracy (blue line) improves consistently across epochs, starting around 0.80 and reaching above 0.90 by the final epoch, which aligns with the decreasing training loss. Conversely, the validation accuracy (orange line) remains relatively stagnant, fluctuating around 0.65 without significant improvement. This gap between training and validation accuracy is another indicator of overfitting, as the model is unable to generalize effectively to validation data despite high performance on the training set.

4.2 SCREENSHOTS

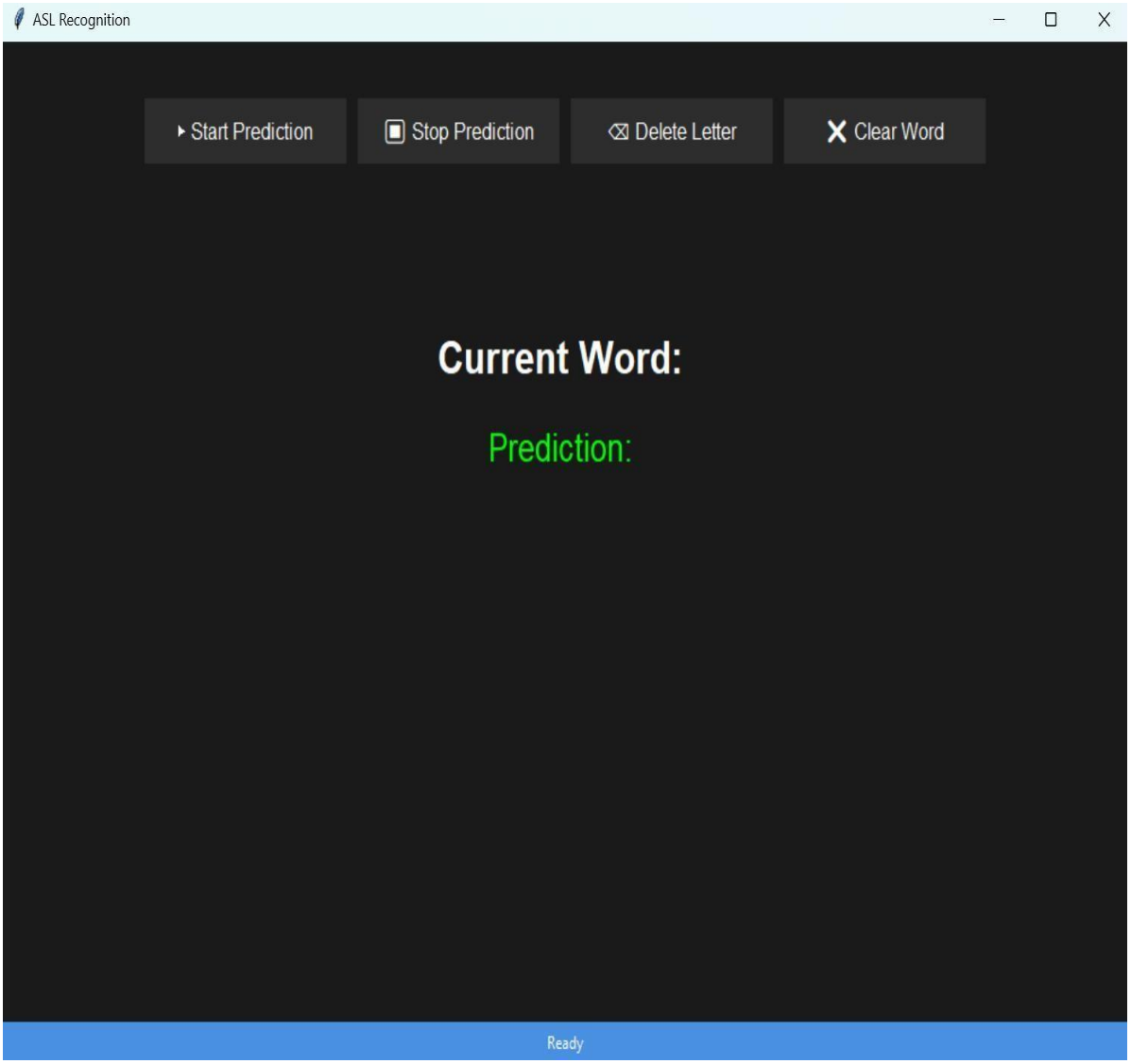

Figure 10

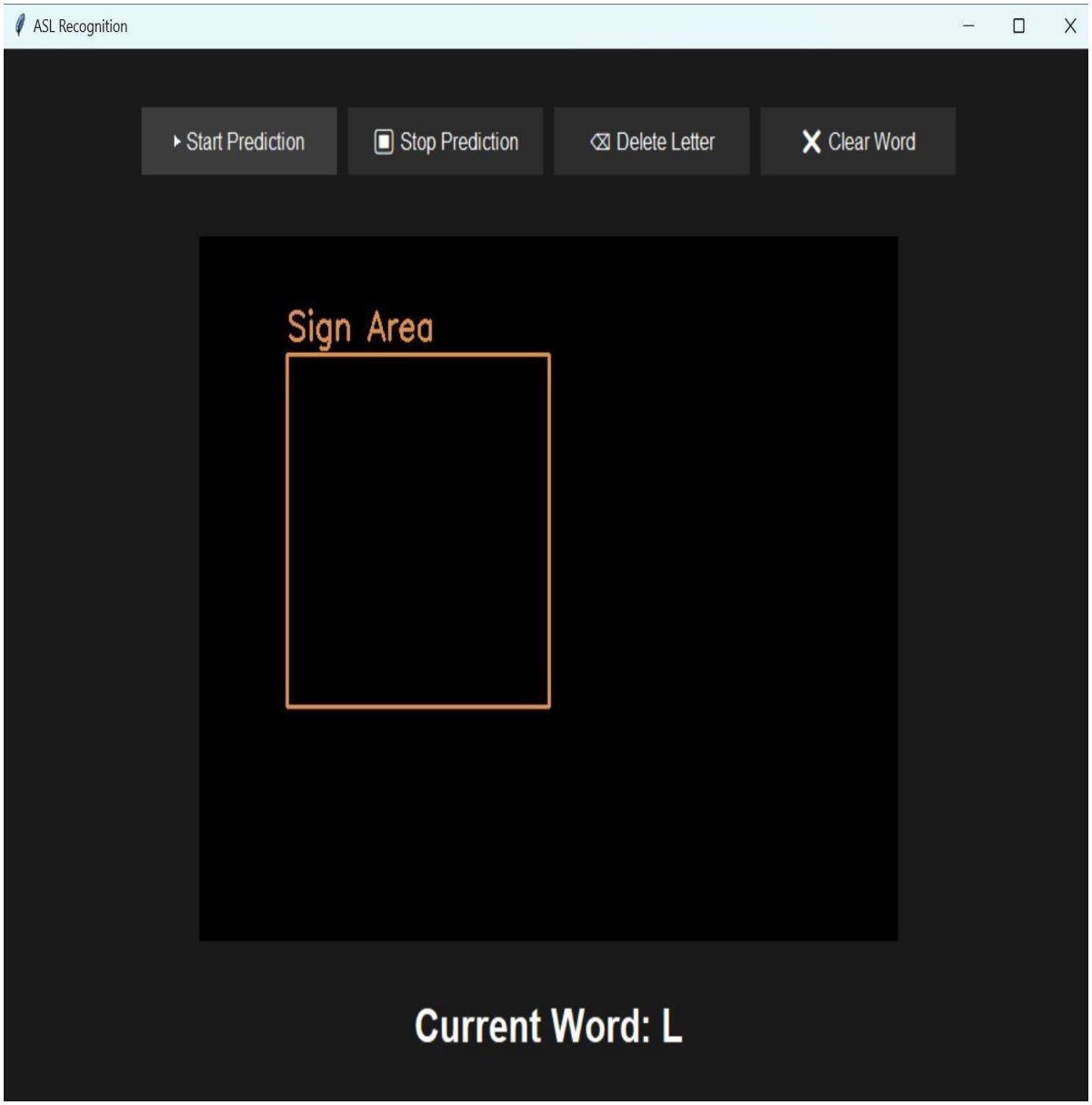
ASL Recognition
Start Prediction
Stop Prediction
Delete Letter
Clear Word
Sign Area
Current Word: L

Figure 11

Chapter 5

CONCLUSION

In this project, we developed a sign language recognition system using deep learning models, specifically MobileNetV2, ResNet50, and InceptionResNetV2. By leveraging TensorFlow and Keras, we trained and evaluated these models on a dataset of sign language gestures, focusing on the accurate identification of hand gestures in real-time.

The results demonstrated that all three models performed effectively, achieving accuracy rates between 63% **and 98%** on the test set, with **InceptionResNetV2** showing the highest performance in terms of accuracy. The system was capable of recognizing gestures in real- time with varying degrees of speed and efficiency. **MobileNetV2**, being the most lightweight model, was the fastest, making it ideal for deployment on resource-constrained devices.

Despite the promising results, challenges such as handling varying lighting conditions, camera positioning, and gesture ambiguity were encountered. These challenges indicated areas for further improvement, such as incorporating temporal data, enhancing data diversity, and fine-tuning the models to handle real-world complexities more effectively.

In conclusion, the project successfully demonstrated the potential of deep learning for sign language recognition. The system can be a valuable tool for enhancing communication, particularly for the hearing-impaired community, by providing a real-time translation of sign language gestures. With further improvements and optimizations, this system has the potential to become an accessible, efficient, and practical solution for real-world applications.

5.1 FUTURE ENHANCEMENTS

While the current sign language recognition system shows promising results, there are several areas where enhancements can be made to improve its accuracy, robustness, and real-world applicability. Below are potential future enhancements:

1. **Temporal gesture recognition**

- Current Limitation: The current system uses individual frames for gesture recognition, which can lead to confusion for gestures that are visually similar or ambiguous when viewed in isolation.
- Future Enhancement: Incorporating **Recurrent Neural Networks** or **Long Short-Term Memory networks** can allow the system to capture temporal information across multiple frames, enabling it to better understand the dynamic nature of sign language gestures, which often rely on movement and context over time. This could significantly improve recognition for gestures that are continuous or context- dependent.

2. **Multi sign language support:**

- Current Limitation: The system may only recognize gestures from one specific sign language.
- Future Enhancement: Expanding the dataset and training the model to recognize multiple sign languages such as British Sign Language, ASL, or International Sign Language would make the system more universally applicable. This could be done by using multilingual sign language datasets or by incorporating transfer learning techniques to adapt a model trained on one sign language to recognize another.

3. **Real time feedback and correction:**

- Current Limitation: The system currently provides predictions without offering feedback on potential errors or areas for improvement.

- Future Enhancement: Implementing a real-time feedback system that provides users with confidence scores or suggestions for improving gestures could enhance the user experience. For example, if the model is uncertain about a prediction, it could display a confidence level or suggest corrections (such as adjusting the hand's orientation or positioning).

REFERENCES

[1] **KASAPBAŞI, Ahmed & El Bushra, Ahmed & AL-HARDANEE, Omar & YILMAZ, Arif**. (2022),DeepASLR: A CNN based Human Computer Interface for American Sign Language Recognition for Hearing-Impaired Individuals,Computer Methods and Programs in Biomedicine Update. 2. 100048. 10.1016/j.cmpbup.2021.100048.

[2] **Sundar, B & Thirumurthy, Bagyammal**. (2022),American Sign Language Recognition for Alphabets Using MediaPipe and LSTM. Procedia Computer Science. 215. 642-651. 10.1016/j.procs.2022.12.066.

[3] **Prof. Mrs. Maheshwari Chitampalli** el.at (2023). REAL TIME SIGN LANGUAGE DETECTION,International Research Journal of Modernization in Engineering Technology and Science.

[4] **Orovwode, Hope & Oduntan, Ibukun & Abubakar, John**. (2023), Development of a Sign Language Recognition System Using Machine Learning 1-8 10.1109/icABCD59051.2023.10220456.

[5] **Alsharif, Bader & Alalwany, Easa & Ilyas, Mohammad.** (2024). Transfer learning with YOLOV8 for real-time recognition system of American Sign Language Alphabet. Franklin Open. 8. 31. 10.1016/j.fraope.2024.100165.

[6] **Obi, Yulius & Claudio, Kent & Budiman, Vetri & Achmad, Said & Kurniawan, Aditya.** (2023), Sign language recognition system for communicating to people with disabilities. Procedia Computer Science. 216. 13-20. 10.1016/j.procs.2022.12.106.