Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique

Université de Carthage

Ecole Nationale d'Ingénieurs de Carthage

وزارة التعليم العالي و البحث العلمي
جامعة قرطاج
المدرسة الوطنية للمهندسين بقرطاج

Industrial Systems and Logistics Engineering Department

End of Year Project Report

American Sign Language Recognition

Realized by: Maram Dhahri

2nd Year Industrial and Logistics Systems Engineer

Supervisor: Mr. Ala Balti

Academic year
2023 - 2024

45 Rue des Entrepreneurs
Charguia II, Code postal : 2035

Acknowledgement

I would like to express my gratitude to my supervisor, Mr. Ala Balti, for his invaluable assistance and guidance throughout the project. This initiative has enabled me to gain a deeper understanding of artificial intelligence and deep learning.

I would like to express my gratitude to the members of the jury who agreed to evaluate this work. Furthermore, I would like to thank the administrative and teaching staff of the National Engineering School of Carthage for providing us with all the resources we needed to succeed.

Table of Contents

List of Figures

Introduction

The Translation System (TS) is a system used by individuals with disabilities to interact with other members of society. Given that computers are an integral part of modern life, the advancement of human-computer interaction (HCI) has facilitated communication between deaf and mute individuals.

The primary objective of our proposed system is to develop an intelligent system that can act as a translator between normal and deaf or dumb individuals, as well as a communication path between individuals with speaking deficiencies and normal individuals. This system employs a Convolutional Neural Network (CNN) based on the deep learning algorithm to effectively extract useful properties for recognizing American Sign Language (ASL), as well as for classifying hand signs. This paper presents a methodology for interpreting American Sign Language (ASL) and provides a comprehensive overview of deep learning-based approaches for gesture recognition. The system was found to be suitable and reliable for individuals with hearing impairments. Additionally, an efficient and cost-effective hand gesture recognition (HGR) system for real-time video stream from a mobile device camera was developed.

Chapter 1: Introduction to ASL Recognition Using CNN and Deep Learning

1.1 Introduction

American Sign Language (ASL) serves as the primary mode of communication for millions of individuals worldwide. In light of the pivotal role that ASL plays in fostering inclusivity and accessibility, this chapter serves as an introductory exploration into the realm of ASL recognition using Convolutional Neural Networks (CNNs) and Deep Learning techniques. The chapter commences with an overview of ASL and its significance. It then proceeds to examine the intricacies of the ASL recognition problem, highlighting the challenges and opportunities inherent in interpreting hand gestures for communication. Furthermore, this chapter offers insights into the development environment employed for this project, namely Google Colab, and introduces key libraries that are essential for implementing ASL recognition algorithms effectively.

1.2 Introduction to American Sign Language (ASL)

American Sign Language (ASL) is a complete, natural language that employs signs made with the hands, facial expressions, and body postures to communicate meaning. It is the primary language of many Deaf communities in the United States and Canada, and it is also used by hearing individuals who are involved with the Deaf community. ASL is not simply a visual representation of English; it has its own grammar, syntax, and linguistic features that are distinct from spoken languages. Just as spoken languages exhibit regional variations and dialects, ASL also has regional variations and dialects. This means that signs may differ in different geographic areas. One of the unique aspects of ASL is its use of space and movement to convey concepts. The placement, movement, and orientation of signs in relation to the body and other signs are crucial for conveying meaning. ASL is a vibrant and rich language that is used in a variety of contexts, including education, social interactions, and professional settings. It is recognized as a legitimate language by linguists and is protected by law in many countries. Learning ASL can facilitate communication and understanding between individuals who are deaf or hard of hearing and those who are not. It can also provide insight into Deaf culture and identity.[1]

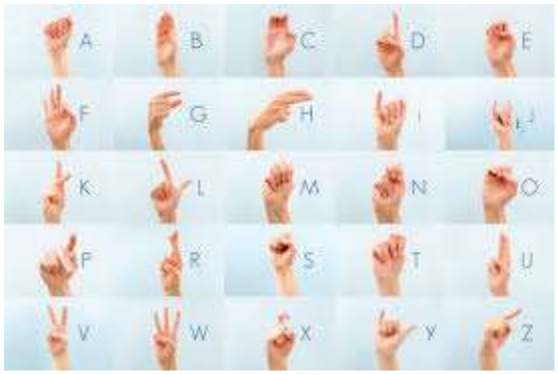


Figure 1American Sign Language

1.3 Motivation for ASL Recognition Using CNN and Deep Learning

The motivation for developing ASL recognition systems using convolutional neural networks (CNNs) and deep learning techniques can be attributed to the desire to bridge communication gaps between individuals who use ASL and those who do not. The following are some key motivations:

- ➢ Accessibility: ASL is the primary language for many Deaf individuals, but it can be challenging for those who do not know sign language to communicate effectively with them. The development of ASL recognition systems can facilitate the bridging of this communication gap by enabling real-time translation of ASL gestures into text or spoken language.
- ➢ Inclusivity: Access to communication is a fundamental human right. The development of ASL recognition systems can enhance the inclusivity of digital platforms, devices, and services for Deaf individuals, thereby enabling them to participate more fully in various aspects of society. ASL recognition systems can be valuable tools in educational settings, aiding Deaf students in accessing instructional materials and participating in classroom discussions. They can also be used by hearing individuals who are learning ASL to practice and improve their signing skills.
- ➢ Efficiency: Traditional methods of interpreting ASL, such as using human interpreters, can be time-consuming and costly. Automated ASL recognition systems offer a more efficient

and cost-effective alternative, particularly in scenarios where real-time communication is essential.

- ➢ Technological Advancements: With the advent of recent advancements in deep learning and computer vision, the accuracy and performance of ASL recognition systems have improved significantly. This makes it an opportune time to explore the development of such systems and leverage cutting-edge technologies to address communication barriers faced by Deaf individuals.[2]

1.4 Understanding the ASL Recognition Problem

The recognition of American Sign Language (ASL) involves the identification and interpretation of hand gestures and movements, which are then translated into textual or spoken form. This process presents several challenges due to the complexity and variability of sign language gestures. An overview of the key aspects of the ASL recognition problem is provided below:

- Variability in Gestures: ASL signs can vary in shape, movement, orientation, and speed, making it challenging to accurately recognize and classify them. Furthermore, the context, regional dialects, and individual signing styles may influence the interpretation of signs.
- Non-manual Components: In addition to hand movements, ASL includes non-manual components such as facial expressions, body postures, and head movements, which convey important grammatical and semantic information. Therefore, recognizing and integrating these non-manual components into the recognition process is essential for understanding the meaning of ASL gestures accurately. ASL recognition systems frequently require real-time functionality to facilitate interactive communication between users. Achieving low latency and high accuracy in real-time recognition is crucial for enabling seamless and natural interactions.
- Data Collection and Annotation: Constructing effective ASL recognition models necessitates the availability of large, diverse datasets of ASL gestures for training. Collecting and annotating such datasets can be labor-intensive and time-consuming, particularly considering the necessity to capture variations in signing styles and environmental conditions. American Sign Language (ASL) has a vast vocabulary of signs,

each representing a distinct word, phrase, or concept. Developing a recognition system that can accurately recognize a diverse range of signs while avoiding confusion between similar-looking signs represents a significant challenge.

- User Adaptation: ASL recognition systems may require adaptation to the signing styles and preferences of individual users, particularly in applications where users interact with the system over an extended period. Personalization of the recognition process to individual users can enhance accuracy and user satisfaction.

Addressing these challenges necessitates a multidisciplinary approach that integrates techniques from computer vision, machine learning, signal processing, and linguistics. Recent advancements in deep learning and neural network architectures, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have demonstrated potential for enhancing the accuracy and resilience of ASL recognition systems. Nevertheless, further research and development are necessary to surmount the remaining challenges and develop ASL recognition systems that are precise, efficient, and accessible to all users.[3]

1.5 Environment and Key Techniques Used in this project

In a project focused on ASL recognition using CNNs and deep learning techniques, the environment and key techniques typically involve a combination of software tools, frameworks, and methodologies tailored to address the specific challenges of sign language recognition. Here's an overview:

1.5.1 Environment

Programming Language: Python is commonly used due to its extensive libraries for machine learning, computer vision, and deep learning.

Figure2 Python

- ➔ Integrated development environments (IDEs), such as Jupyter Notebook, PyCharm,Google Colab, or Visual Studio Code, are frequently employed for code development and experimentation.
- ➔ Hardware: Deep learning models often necessitate substantial computational resources, necessitating the use of a powerful GPU (Graphics Processing Unit) or access to cloud-based GPU instances for the efficient training of large models.[4]

1.5.2 Key Techniques

- Convolutional Neural Networks (CNNs): CNNs are widely used for image recognition tasks due to their ability to effectively capture spatial hierarchies and patterns in images. They form the backbone of many ASL recognition systems for extracting features from sign language images.
- Deep Learning Architectures : Architectures such as deep convolutional neural networks (DCNNs), recurrent neural networks (RNNs), and their variants (e.g., CNN-LSTM) are commonly employed to model temporal dependencies and capture complex patterns in sequential data, which is essential for recognizing dynamic gestures in sign language.
- Data Augmentation: Given the limited availability of annotated ASL datasets, data augmentation techniques such as rotation, scaling, translation, and flipping are employed to artificially increase the size and diversity of training data, thereby improving the generalization ability of the models.
- Transfer Learning: Transfer learning involves leveraging pre-trained models (e.g., ImageNet) as a starting point and fine-tuning them on ASL datasets. This approach can expedite the training process and improve the performance of ASL recognition models, especially when training data is limited.

- Gesture segmentation is a technique that is essential for accurate recognition. This may involve methods based on hand detection, motion tracking, or temporal segmentation algorithms.
- Model optimization is also a crucial aspect of this process. Techniques such as batch normalization, dropout, and gradient clipping are employed to enhance the convergence of ASL recognition models, prevent overfitting, and augment their generalizability.
- Evaluation Metrics: Metrics such as accuracy, precision, recall, F1-score, and confusion matrices are utilized to assess the performance of ASL recognition models on validation and test datasets.[4]

1.5.3 The utilized development environment: Google Colab Pro

Google Colab offers several advantages for the development of ASL recognition projects. It provides free access to powerful GPUs (Graphics Processing Units) and TPUs (Tensor Processing Units), which are essential for training deep learning models efficiently. This is particularly beneficial for researchers and students who may not have access to high-performance hardware. Colab is cloud-based computing, which allows users to execute code and train models in the cloud without the need for powerful local hardware. This enables collaboration and facilitates reproducibility since code and data are stored centrally. Colab seamlessly integrates with Google Drive, allowing users to access and manipulate datasets, models, and code stored in their Drive accounts directly from the Colab environment. This simplifies data management and allows users to work with larger datasets. Colab is hosted on Google's cloud infrastructure, enabling users to execute code and train models in the cloud without the need for local hardware. This facilitates collaboration and reproducibility, as code and data are stored centrally.

Figure 3 Google Colab

Integration with Google Drive: Colab integrates seamlessly with Google Drive, allowing users to access and manipulate datasets, models, and code stored in their Drive accounts directly from the Colab environment. This simplifies data management and sharing among collaborators.

- Pre-installed Libraries and Tools: Colab is pre-installed with a selection of popular data science libraries, including TensorFlow, PyTorch, Keras, and scikit-learn, as well as essential tools such as Jupyter Notebook. This streamlines the setup process, allowing users to commence coding immediately.
- Interactive Development Environment: Colab provides an interactive development environment similar to Jupyter Notebook, enabling users to write and execute code in a cell-based format. This facilitates iterative development and experimentation with different algorithms and parameters.
- Notebook Sharing and Collaboration: Colab notebooks can be easily shared with collaborators via a simple URL link. This enables real-time collaboration on ASL recognition projects, as multiple users can edit and run the same notebook simultaneously.
- Integration with GitHub: Colab offers seamless integration with GitHub, enabling users to clone repositories, collaborate on projects, and utilize version control for their code within the Colab environment.[5]

1.5.4 Main Libraries Used in this Project

In an ASL recognition project utilizing Google Colab or similar environments, several key libraries play crucial roles in developing and implementing machine learning and computer vision models. Among the most commonly used libraries in such projects are :

- TensorFlow, a widely-used open-source machine learning framework developed by Google, and PyTorch, another popular open-source deep learning framework known for its flexibility and dynamic computation graph. It provides comprehensive tools and resources for the construction, training, and deployment of deep learning models, including convolutional neural networks (CNNs) for image recognition tasks such as ASL recognition.[6]

Figure 4 TensorFlow

- PyTorch: PyTorch is another popular open-source deep learning framework that is renowned for its flexibility and dynamic computation graph. Many researchers and practitioners prefer PyTorch for its intuitive interface and easy debugging capabilities when working on ASL recognition and related projects. [6]

- Keras is a high-level neural networks API that runs on top of TensorFlow or other backends like Theano or Microsoft Cognitive Toolkit (CNTK). It simplifies the process of building and training neural networks, making it a convenient choice for prototyping and experimenting with ASL recognition models.[6]
- OpenCV is a powerful library for computer vision tasks. It provides a comprehensive array of functions and algorithms for image processing, feature extraction, object detection, and gesture recognition, which are essential for preprocessing ASL images and extracting relevant features.[6]
- Scikit-learn: Scikit-learn is a versatile machine learning library in Python that offers a diverse range of algorithms and tools for classification, regression, clustering, and

dimensionality reduction. It can be utilized for tasks such as data preprocessing, feature selection, and the evaluation of ASL recognition models.[7]

- NumPy is a fundamental library for numerical computing in Python. It provides support for multi-dimensional arrays, mathematical functions, linear algebra operations, and random number generation, which are essential for handling image data and performing matrix operations in ASL recognition projects.[7]
- Matplotlib and Seaborn are additional libraries that facilitate the creation of charts, plots, and graphs to visualize ASL dataset distributions, model performance metrics, and other relevant information during the development and evaluation phases of the project. Matplotlib and Seaborn are visualization libraries that facilitate the generation of charts, plots, and graphs to represent the distributions of ASL datasets, the performance metrics of models, and other pertinent information throughout the development and evaluation phases of the project.[7]

1.5 Conclusion

In this introductory chapter, we established the foundation for comprehending the ASL recognition challenge and its implications. By examining the significance of American Sign Language (ASL) and the difficulties associated with ASL recognition, we have demonstrated the necessity of employing advanced techniques such as convolutional neural networks (CNNs) and deep learning for accurate interpretation. Furthermore, by introducing the development environment and key libraries employed in this project, we established the foundation for subsequent chapters' exploration into ASL recognition methodologies and implementations. As we embark on this endeavor, our objective is to contribute to the advancement of ASL recognition technology, with the ultimate goal of fostering greater inclusivity and accessibility for individuals within the deaf and hard-of-hearing community.

Chapter 2: Methodology and Techniques

2.1 Introduction

In this chapter, we embark on a comprehensive exploration of the methodology and techniques involved in developing effective ASL recognition systems using Convolutional Neural Networks (CNNs) and deep learning. Beginning with an introduction to CNNs and their significance in image recognition tasks, we delve into the design of specialized CNN architectures tailored for ASL recognition. Subsequently, we examine training strategies essential for optimizing CNNs for accurate ASL recognition. Furthermore, this chapter delves into data collection, preprocessing, and exploration techniques crucial for preparing ASL datasets for effective model training. From dataset collection and preprocessing to exploratory data analysis and visualization, each step is meticulously crafted to ensure robust and accurate recognition of ASL gestures. Lastly, training and testing strategies are explored to evaluate the performance and reliability of ASL recognition CNNs, paving the way for the development of robust and efficient recognition systems.

2.2 Introduction to Convolutional Neural Networks (CNNs)

Convolutional neural networks (CNNs) are a class of deep neural networks that are primarily designed to process and analyze visual data, such as images and videos. CNNs have been remarkably successful in various computer vision tasks, including image classification, object detection, segmentation, and more. This article provides an introduction to the key concepts of CNNs, including convolutional layers and the convolution operation. CNNs employ convolutional layers, wherein a set of learnable filters (also referred to as kernels) slide over the input image, performing element-wise multiplication and summation operations to generate feature maps.
Feature Extraction: Each filter detects distinct patterns or features within the input image, such as edges, textures, or shapes, capturing hierarchical representations of the input data. Following convolution, pooling layers are employed to downsample the spatial dimensions of the feature maps, thereby reducing computational complexity and the number of parameters. Max pooling and average pooling are two common pooling techniques. Max pooling retains the maximum value within each pooling region, whereas average pooling computes the average. Activation functions,

such as ReLU (Rectified Linear Unit), sigmoid, or tanh, introduce non-linear transformations to the feature maps, enabling the network to learn complex patterns and relationships in the data. ReLU is a widely used activation function due to its simplicity and effectiveness, as it replaces negative pixel values with zero while leaving positive values unchanged. Following convolutional and pooling layers, convolutional neural networks (CNNs) typically include one or more fully connected (dense) layers. These layers are used to learn high-level features and perform classification or regression tasks. Before the dense layers, the feature maps are flattened into a vector, which is then fed into the fully connected layers.[8]

2.3 Designing a CNN Architecture for ASL Recognition

Figure 1 presents a CNN-based deep learning architecture for the purpose of recognizing hand signs. Convolutional neural networks (CNNs), a type of deep learning neural network, are frequently employed for image identification and classification applications. CNNs are designed to automatically detect features in input images through a process called convolution, which extracts specific features such as edges, corners, and textures [9]. In conclusion, the CNN architecture was selected for hand sign recognition due to its inherent capacity to capture spatial features and patterns within images, which aligns perfectly with the distinctive visual characteristics of hand signs, resulting in superior recognition accuracy.

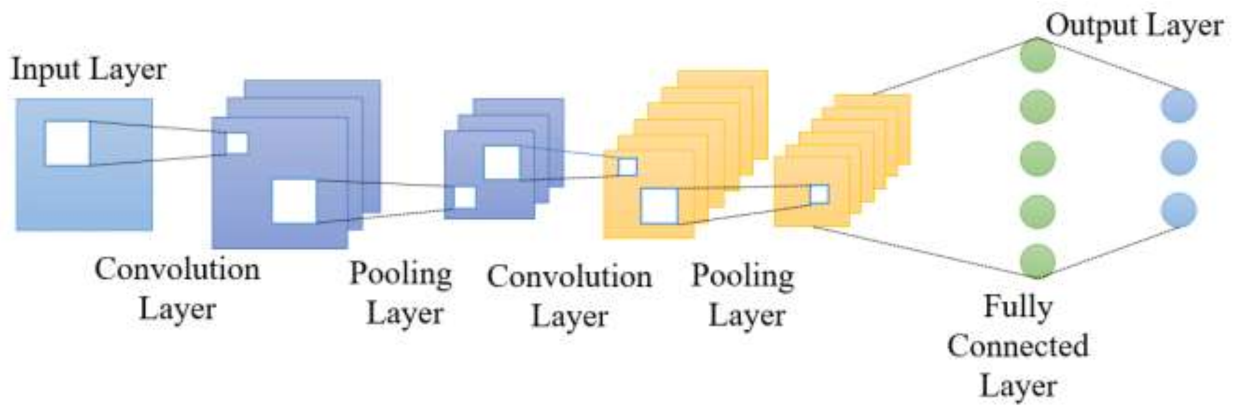


Figure 5 Proposed architecture

2.4 Data Collection and Exploration Techniques

In the field of ASL recognition using Convolutional Neural Networks (CNNs), meticulous attention to data collection, preprocessing, and exploration techniques is of paramount importance for ensuring the model's accuracy and robustness. In the field of ASL recognition using Convolutional Neural Networks (CNNs), meticulous attention to data collection, preprocessing, and exploration techniques is of paramount importance for ensuring the model's accuracy and robustness. Data collection involves the assembly of a diverse dataset of ASL images or videos, the capture of various hand gestures across different individuals and environmental conditions, and the subsequent preparation of the data for effective training through standardization of its format and augmentation of its diversity. Moreover, exploration techniques such as visualization, statistical analysis, and data cleaning reveal insights into the distribution, characteristics, and potential challenges of the dataset, empowering researchers to make informed decisions throughout the model development process. By meticulously executing these steps, practitioners can establish a robust foundation for developing reliable ASL recognition systems that accommodate the diverse nuances of sign language communication.[10]

2.4.1 Dataset Collection for ASL Recognition

The primary source of data for this project was the compiled dataset of American Sign Language (ASL) called the ASL Alphabet from Kaggle user Akash [3]. The dataset consists of 29,000 images which are 200x200 pixels. There are 29 total classes, each with 3000 images, 26 for the letters A-Z and 3 for space, delete and nothing. This data is solely of the user Akash gesturing in ASL, with the images taken from his laptop’s webcam. These photos were then cropped, rescaled, and labeled for use.

2.4.2 Data exploration Techniques for ASL Images

2.4.2.1Exploratory data analysis

Exploratory Data Analysis (EDA) represents a foundational step in the process of understanding the intricacies of the ASL dataset for the purpose of developing Convolutional Neural Network (CNN) based recognition systems. Through EDA, researchers examine the composition of the dataset, visualizing ASL images or video frames along with their labels to gain insight into the diversity and distribution of hand gestures. Descriptive statistics provide insight into the range and variability of pixel intensities. Insights gleaned from correlation analysis reveal potential patterns or relationships between features. Furthermore, EDA facilitates the identification of class imbalances and data anomalies, thereby guiding subsequent data cleaning and preprocessing efforts. By meticulously exploring the dataset, researchers are able to gain the knowledge necessary to make informed decisions throughout the model development process. This ultimately paves the way for the development of more accurate and robust ASL recognition systems.[5]

2.4.2.2 Visualization of ASL image samples

Prior to delving into the intricate details of developing an ASL recognition system, it is imperative to embark on the journey of visualizing the ASL image samples within the dataset. These visual representations offer invaluable insights into the richness and complexity of sign language gestures encoded within each image. By examining these samples, researchers can begin to appreciate the diversity of hand shapes, orientations, and movements captured in the dataset. Moreover, the visualization of the ASL images provides a foundation for the understanding of the inherent challenges in the recognition task, including variations in lighting conditions, background clutter, and hand articulations. This initial exploration enables researchers to gain a deeper understanding of the dataset's characteristics, thereby setting the stage for subsequent preprocessing steps and model development.

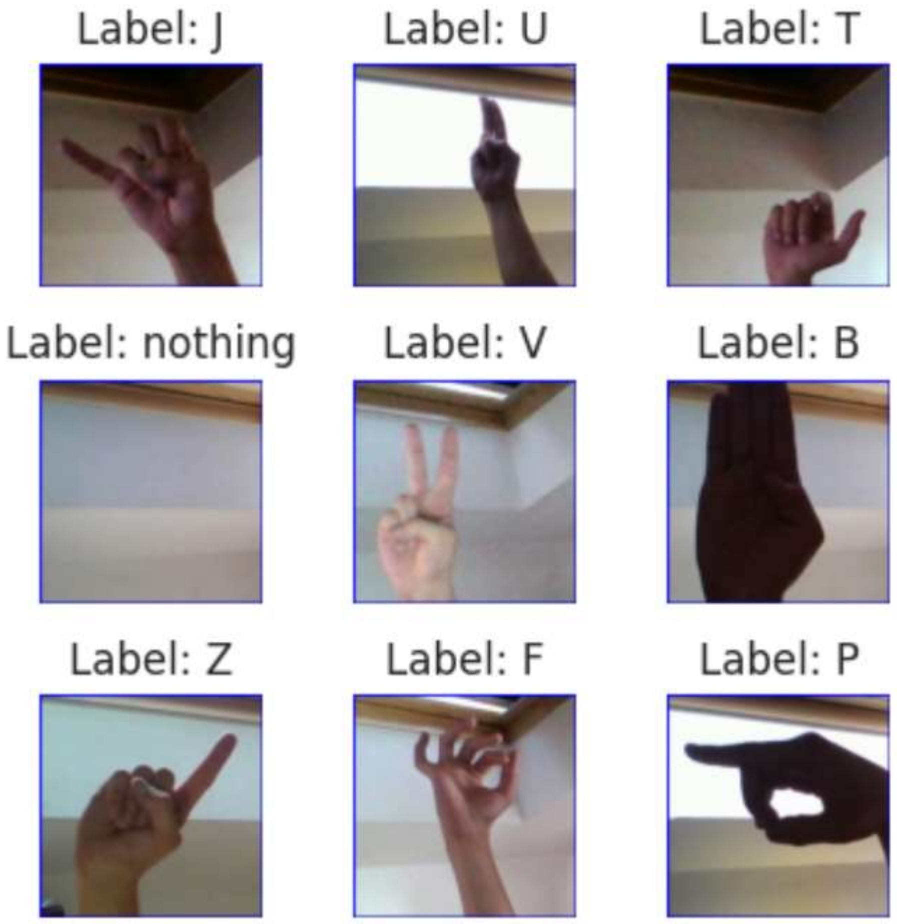


Figure 6 Examples of images from the dataset used for training

2.4.2.3 Class distribution in the database

An understanding of the class distribution within the database is a fundamental prerequisite for the development of an effective ASL recognition system. This distribution reveals the relative frequencies of different ASL gestures represented in the dataset, thereby providing crucial insights into the dataset's balance and potential biases. By analyzing the class distribution, researchers can identify classes that are overrepresented or underrepresented, which in turn informs decisions on data augmentation strategies or class weighting during model training. Furthermore, an imbalance in class distribution may indicate difficulties in accurately recognizing certain ASL gestures, prompting further investigation and refinement of the recognition algorithm. By gaining a comprehensive understanding of the class distribution, researchers can better tailor their modeling approach to ensure equitable representation and enhance the system's overall performance.

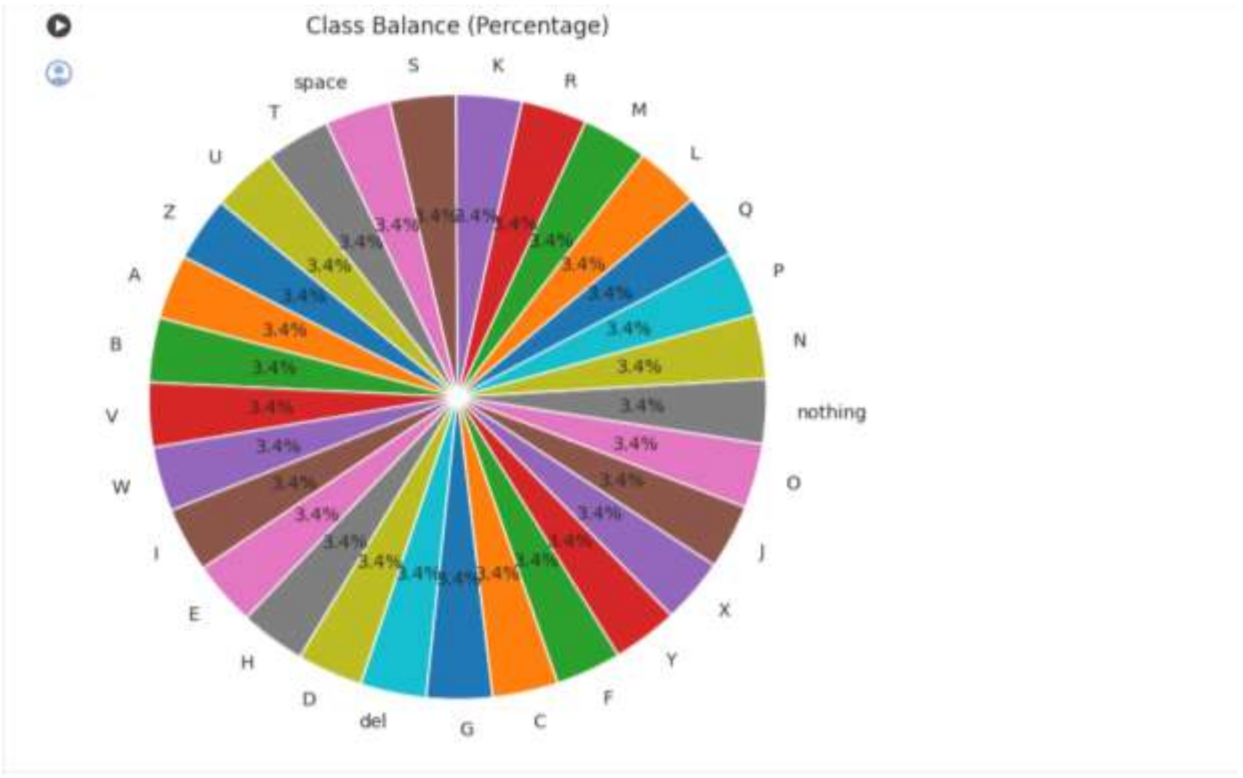


Figure 7 Class Balance

2.5 Training and Testing for ASL Recognition

2.5.1 Training for ASL Recognition

Convolution: A third function is created via convolution, a mathematical procedure that mixes two functions. In image analysis and machine vision, convolution is a widely used technique for filtering and extracting features from images. In convolution, a low matrix called a filter is applied to every pixel of a picture to produce a new pixel value. The kernel is typically a small square matrix, and the size of the kernel determines the size of the output image. The process involves sliding the kernel over each pixel of the input image and performing a mathematical operation between the kernel and the input image. The result is a new pixel value in the output image. Convolution can be used for a variety of tasks, such as edge detection, image sharpening, blurring, and feature extraction. It is also an essential component of deep learning neural networks, where it is used to extract features from images or other data types.[11]

Pooling: Convolutional neural networks (CNNs) frequently employ the pooling approach to scale back the spatial dimension of feature maps. Pooling is used to downscale the feature maps while keeping the most crucial details. Max pooling is the most commonly used type of pooling in CNNs, where a kernel of a fixed size slides over the feature map and selects the maximum value within the kernel. Max pooling is a strategic inclusion in CNN architecture for hand sign recognition as it serves two key purposes. Firstly, it assists in reducing the spatial dimensions of feature maps, enabling the network to focus on the most essential information while discarding redundant details. Secondly, max pooling enhances the network's robustness against slight variations in hand sign positioning, scale, or orientation, ensuring accurate recognition by capturing the most prominent features regardless of their precise location within the image.

The reduction of spatial descriptors in feature maps through pooling facilitates the capture of the most significant features of the input data while discarding irrelevant details. This enhances the model's effectiveness in responding to variations in the input feature and prevents overfitting.[12]

Activation function: ReLU is a typical activation function in neural networks, including Convolutional Neural Networks (CNNs). It is a simple function that returns the input value if it is positive, and zero otherwise. The primary advantage of using ReLU over other activation functions is that it is computationally efficient, allowing for faster training of deep neural networks. Additionally, ReLU has been shown to improve the accuracy of the network, compared to other activation functions. Another advantage of ReLU is that it helps to address the problem of vanishing gradients, which can occur when using other activation functions such as sigmoid or tanh. This is because ReLU does not saturate for positive input values, which means that it does not cause the gradients to become small, and thus, it does not hinder the learning process. Furthermore, ReLU is an optimal choice for CNN architecture in hand sign recognition due to its non-linearity and computational efficiency. Its ability to facilitate gradient propagation addresses vanishing gradient problems, promoting faster convergence during training. This, in turn, enables the network to effectively learn intricate features from hand sign images, resulting in improved recognition performance.

This sequential model architecture is based on the EfficientNetB0 model, a state-of-the-art convolutional neural network architecture that is known for its efficiency and effectiveness in image classification tasks. The following section will present a detailed breakdown of the components and their functionalities.[13]

EfficientNetB0 (Functional):

This layer represents the EfficientNetB0 backbone model, which is responsible for feature extraction from input images. The model comprises multiple convolutional layers, which achieve high efficiency by balancing model depth, width, and resolution.

The output shape is (None, 7, 7, 1280), with a total of 4,049,571 parameters. (None, 3, 3, 1280)

GlobalMaxPooling2D: This layer applies global max pooling to the output of the average pooling layer, reducing the spatial dimensions to a single dimension, effectively extracting the most important features. (None, 1280)

Dense:

This dense layer comprises 256 neurons and applies a linear transformation to the input data, followed by a rectified linear unit (ReLU) activation function.

Output Shape: (None, 256)

Total Params: 327,936

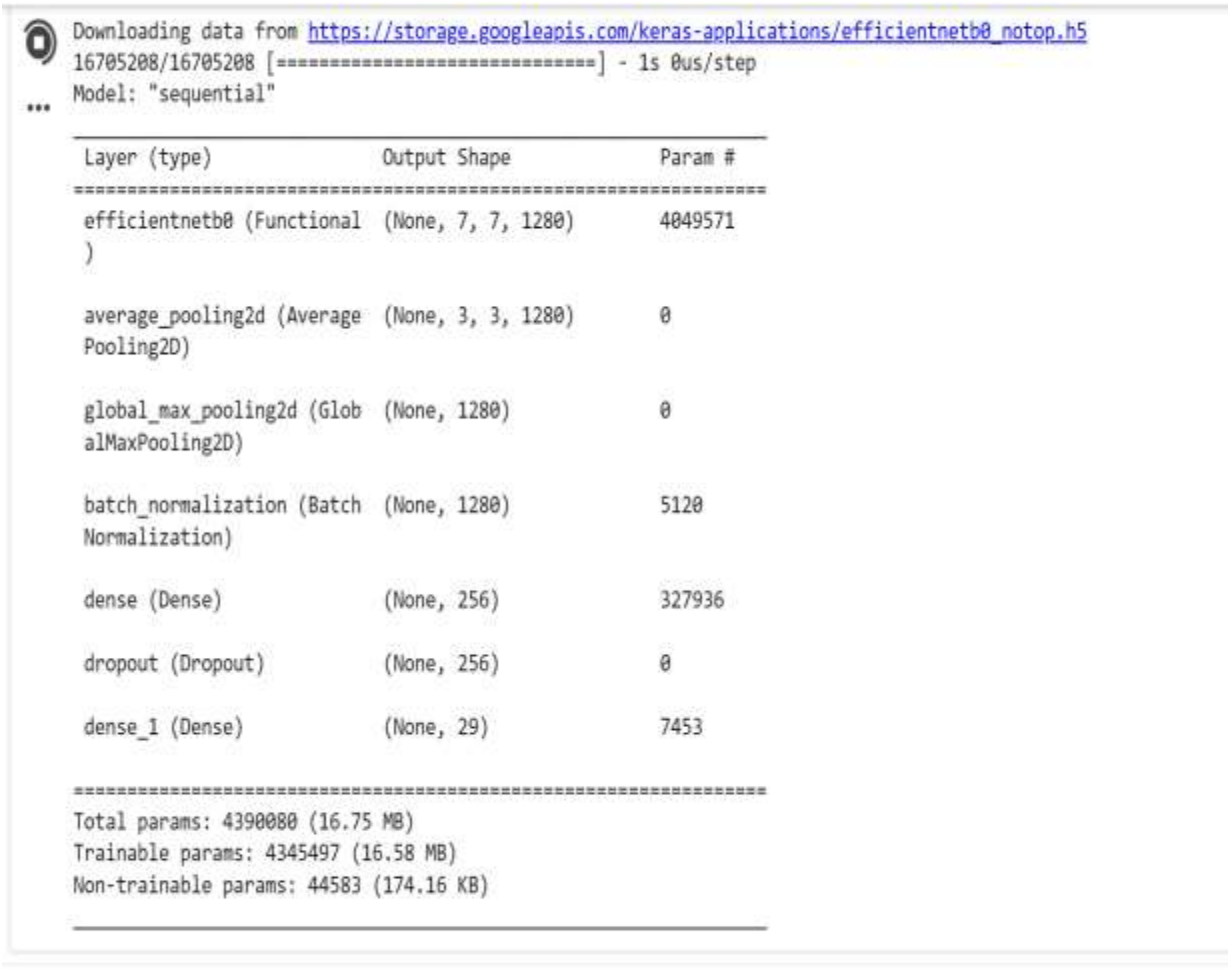
Dropout:

Dropout regularization is applied to the output of the dense layer to prevent overfitting by randomly dropping a fraction of the neurons during training.

Output Shape: (None, 256)

Dense_1:

This is the output layer of the model, consisting of 29 neurons, which corresponds to the number of classes in the ASL dataset. The total number of parameters is 44,583, with a corresponding size of 174.16 KB



```
Downloading data from https://storage.googleapis.com/keras-applications/efficientnetb0_notop.h5
16705208/16705208 [==============================] - 1s 0us/step
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 efficientnetb0 (Functional  (None, 7, 7, 1280)        4049571
 )

 average_pooling2d (Average  (None, 3, 3, 1280)        0
 Pooling2D)

 global_max_pooling2d (Glob  (None, 1280)              0
 alMaxPooling2D)

 batch_normalization (Batch  (None, 1280)              5120
 Normalization)

 dense (Dense)               (None, 256)               327936

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 29)                7453

=================================================================
Total params: 4390080 (16.75 MB)
Trainable params: 4345497 (16.58 MB)
Non-trainable params: 44583 (174.16 KB)
_________________________________________________________________
```

Figure 8 Show that the number of parameters in each layer

2.5.2 Testing for ASL Recognition

To ascertain the efficacy of our image preprocessing methodology, we conducted a test on a set of images comprising both those from the original dataset and our own collected data. Our findings indicate that the model trained on preprocessed images exhibited superior performance compared to the model trained on the original images. This outcome is likely attributed to the former's reduced susceptibility to overfitting.

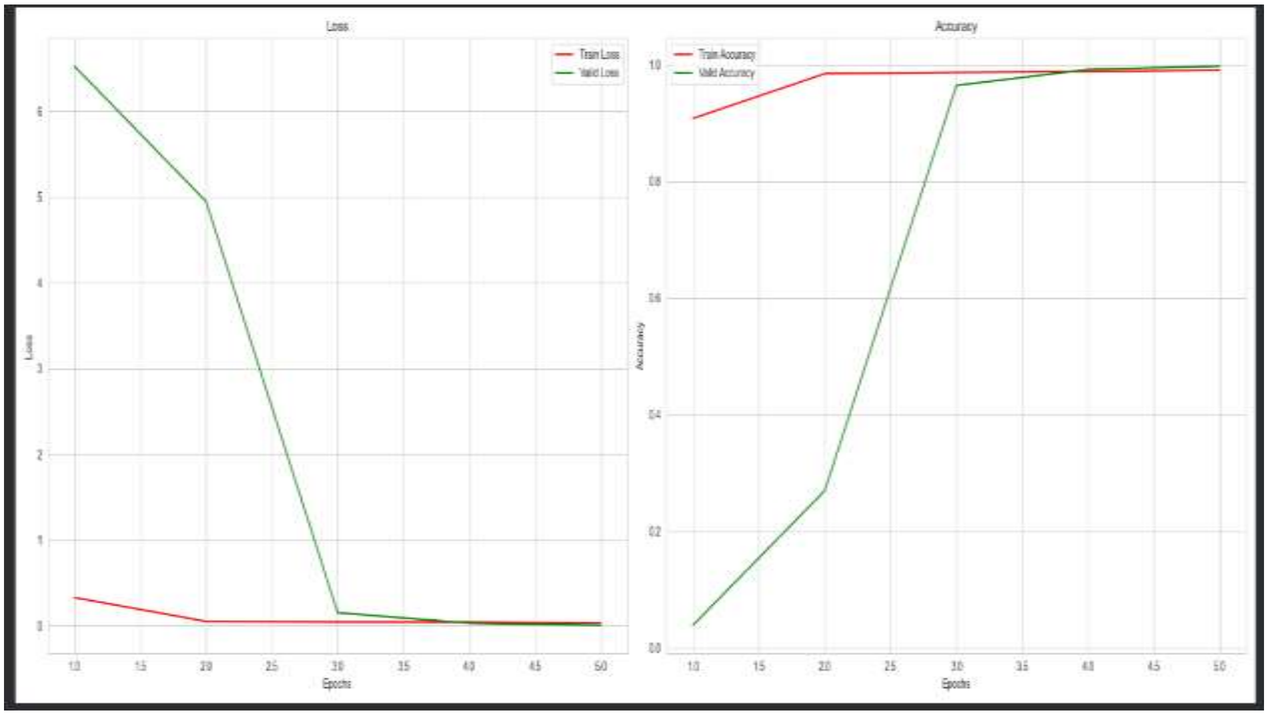


Figure 9 Training and Validation Performance of Model Trained on Original Images

2.6 Conclusion

In this chapter, we have laid the foundation for the development of ASL recognition systems by exploring a myriad of methodologies and techniques. By introducing Convolutional Neural Networks (CNNs) and their application in ASL recognition, we've established a framework for designing specialized CNN architectures tailored to interpret ASL gestures effectively. Moreover, we've delved into training strategies essential for optimizing CNNs, ensuring their robustness and accuracy. Additionally, data collection, preprocessing, and exploration techniques have been meticulously examined to prepare ASL datasets for training, highlighting the importance of thorough analysis and preparation. As we move forward, armed with these methodologies and techniques, we are poised to develop robust and efficient ASL recognition systems capable of enhancing accessibility and inclusivity for individuals within the deaf and hard-of-hearing community.

Chapter 3: Results, Discussion and Model deployment

3.1 Introduction

This chapter delves into the outcomes of our ASL recognition model, discussing its performance, implications, and deployment strategies. We begin by dissecting the results obtained, evaluating metrics such as accuracy and loss, and identifying areas for improvement. Subsequently, we explore potential future directions for advancing ASL recognition technology, emphasizing the integration of advanced model architectures and continual learning strategies. Moreover, we delve into the practical aspects of deploying our model in real-world scenarios, considering factors such as hardware requirements, integration, and user interface design.

3.2 Discussion and Future Directions

In this section, we engage in an in-depth discussion of the results obtained from our ASL recognition model, which achieved an accuracy of 0.8859 and a loss of 0.4256. We meticulously analyze these performance metrics to gain insights into the model's efficacy in interpreting ASL gestures.

The achieved accuracy of 0.8859 signifies the proportion of correctly classified ASL gestures out of the total number of gestures in the test dataset. This metric serves as a measure of the model's overall correctness in recognizing hand signs, providing a valuable indication of its performance. While an accuracy of 0.8859 demonstrates a commendable level of success, it also prompts us to explore areas where further improvement may be warranted.

Moreover, the observed loss of 0.4256 indicates the model's degree of error during training and testing phases. A lower loss value signifies better convergence and model optimization, suggesting that our ASL recognition model has effectively minimized discrepancies between predicted and actual labels. However, a loss of 0.4256 prompts us to scrutinize aspects of the model architecture, training strategy, and dataset characteristics that may contribute to this level of error.

```
50/50 [==============================] - 1s 47ms/step - loss: 0.4256 - accuracy: 0.8859
```

Figure 10 Loss and accuracy of Model Trained on Original Images

Furthermore, as we discuss the strengths and weaknesses observed during testing, we acknowledge the model's proficiency in correctly classifying a significant portion of ASL gestures while also recognizing areas for enhancement. We consider factors such as class imbalances, data variability, and model complexity in our analysis, aiming to identify strategies for mitigating limitations and optimizing performance.

Looking towards future directions, we explore the incorporation of advanced model architectures, such as attention mechanisms or transformer networks, to further enhance recognition accuracy and efficiency. Additionally, we consider the integration of multi-modal fusion techniques, combining visual data with skeletal or depth information, to capture richer representations of ASL gestures. Furthermore, we delve into continual learning strategies, allowing our ASL recognition model to adapt and improve over time as it encounters new data or evolving signing styles.

By meticulously analyzing our model's performance metrics, strengths, and weaknesses, and exploring avenues for future improvement, we aim to pave the way for the development of more robust and accurate ASL recognition systems, ultimately enhancing accessibility and inclusivity for individuals within the deaf and hard-of-hearing community.

3.3 Interpretation of Results

3.3.1Précision

```
50/50 [==============================] - 3371s 177s/step
0.7900201792154989084
```

Figure 11 Precision

A prediction accuracy of 79% (Figure 3.3) suggests that the ASL recognition model accurately predicts hand gestures' classes in 79% of cases. While this accuracy is commendable, it's crucial to acknowledge the implications of misclassification in the remaining 21% of cases. Misclassification can lead to misunderstandings or misinterpretations of sign language, potentially impacting effective communication. Therefore, despite a relatively high accuracy rate, addressing misclassification errors remains essential for improving the model's overall effectiveness and ensuring inclusive communication accessibility for users of ASL recognition technology.

3.3.2 F1-score

```
Weighted F1-score: 0.4920901792154989084
```

Figure 12 F1-score

The F1-score represents a crucial evaluation metric in conjunction with precision. An F1-score of 49% (Figure 3.4) indicates the potential for enhanced accuracy in the identification of ASL gestures. The F1-score represents the harmonic mean of precision and recall, offering insights into the model's overall performance. A lower F1 score may indicate deficiencies in both precision and recall, thereby highlighting areas where the model may encounter difficulties in effectively recognizing specific ASL signs. Consequently, enhancing the F1-score is crucial for refining the model's capacity to accurately interpret a diverse range of ASL gestures, thereby enhancing communication accessibility for individuals within the deaf and hard-of-hearing community.

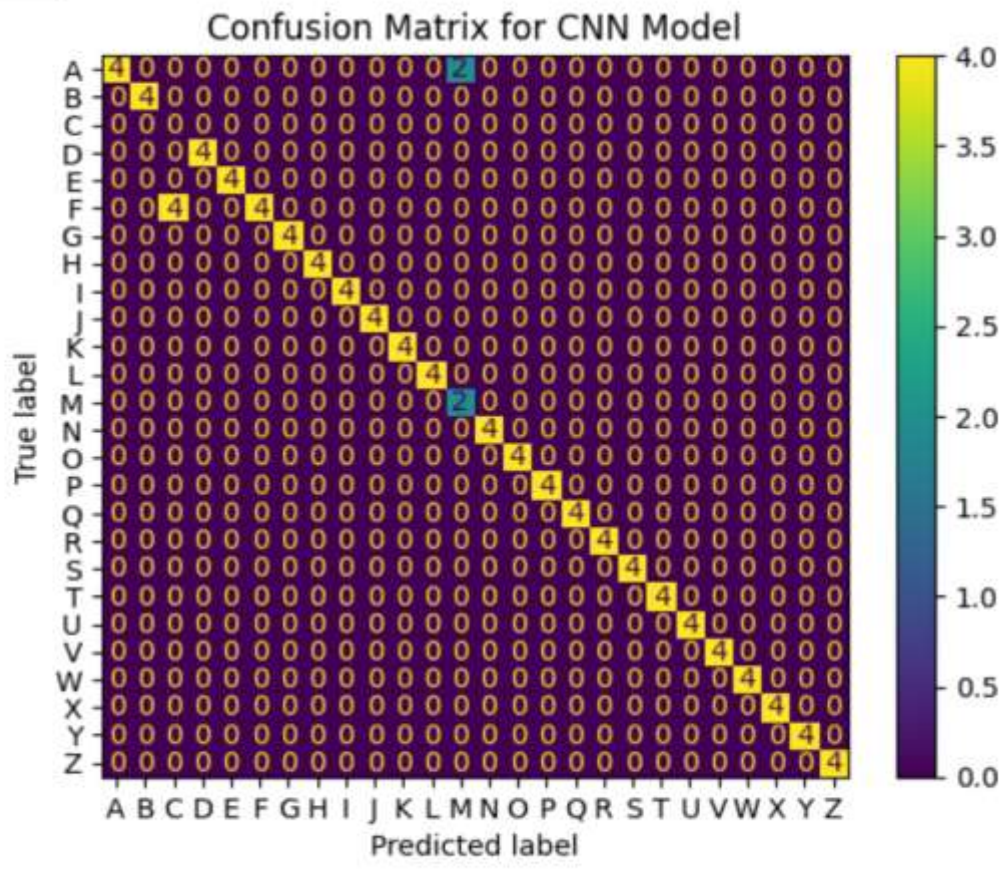
3.3.3 Confusion Matrix

The confusion matrix is a crucial instrument for evaluating the efficacy of a classification model, offering a comprehensive overview of the model's predictions in comparison to the ground truth labels. In the context of ASL recognition, the confusion matrix permits the assessment of the model's capacity to distinguish between different sign language gestures. Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class. By analyzing the confusion matrix, it is possible to identify patterns of misclassification and gain insights into which gestures the model struggles to accurately recognize.

This information is invaluable for refining the model's architecture, training data, and algorithms to improve overall performance. Additionally, the confusion matrix serves as a valuable tool for stakeholders to understand the model's strengths and weaknesses and make informed decisions about its deployment and further development.[15]

A review of the confusion matrix for the model test set in Figure 18 reveals that the model is relatively adept at correctly identifying the letter in question. Upon examination of the confusion matrix, it becomes evident that certain letters, such as A and M, are prone to misclassification. This is exemplified in Figure 19 where it can be observed that the two letters share a similar shape. This observation provides insight into the potential reasons behind the model's confusion, as an A may be perceived as a repeated M.

Figure 13 Confusion Matrix



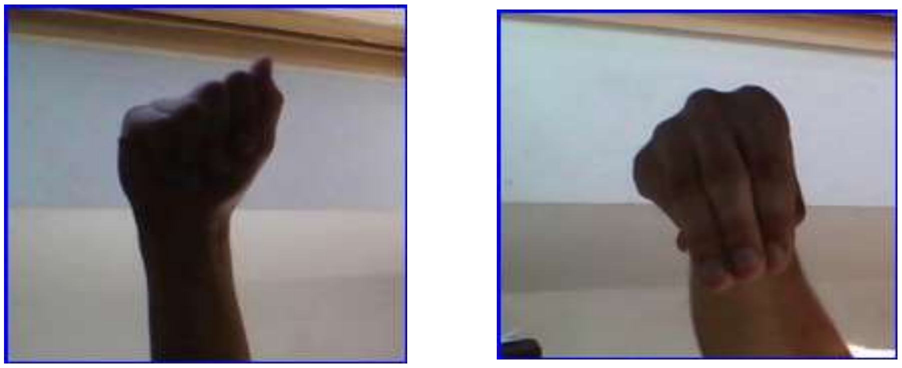


Figure 14 Showcasing similarity between the ASL gesture for A and M

3.4 Model deployment:

3.4.1 Flask Framework:

Flask offers a robust and flexible framework for developing web applications, rendering it an optimal choice for deploying ASL recognition models. With Flask, developers can effortlessly define routes, process HTTP requests, and render dynamic HTML content. The lightweight nature of Flask facilitates rapid prototyping and scalability, rendering it suitable for both small-scale projects and large-scale deployments. Furthermore, Flask's extensive ecosystem of extensions and plugins provides additional functionality, such as database integration, authentication, and session management, thereby further enhancing the capabilities of ASL recognition applications.

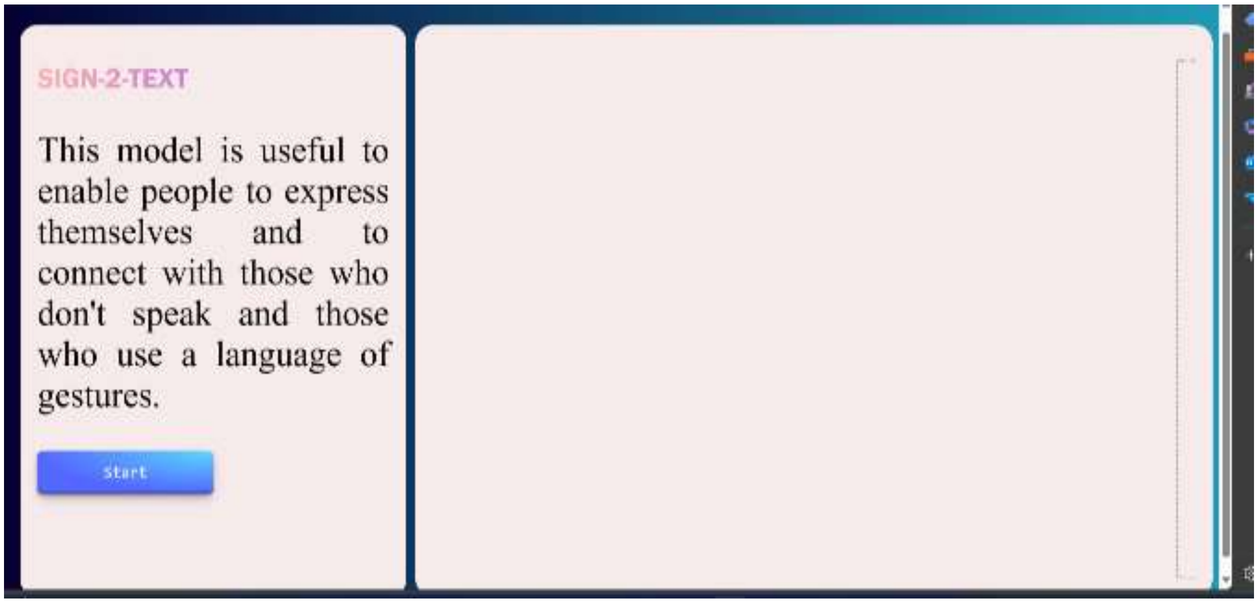
3.4.2 Load the Model

The loading of the ASL recognition model into the Flask application represents a pivotal step that requires meticulous attention to detail. It is of paramount importance that developers ensure that the trained model is stored in a format compatible with the chosen machine learning framework, such as TensorFlow or PyTorch. This typically involves the saving of the model's architecture, weights, and configuration to a file format that can be easily loaded during runtime. Once the model has been saved, Flask routes can be configured to load the model into memory when the application starts up. It is of the utmost importance to optimize the loading process in order to minimize the startup time and memory overhead, particularly in the case of larger models with complex architectures.

Moreover, it is of the utmost importance to incorporate version control and model management practices in order to maintain the integrity and reliability of the deployed ASL recognition system. This encompasses the implementation of mechanisms for the tracking of model versions, the management of dependencies, and the handling of updates in a seamless manner. The process of versioning a model ensures that changes can be tracked over time, allowing for the simple rollback to previous versions in the event that this is necessary. Furthermore, developers should implement protocols for monitoring model performance and conducting regular evaluations to identify potential issues or degradation in accuracy. By implementing robust model management practices, developers can ensure the continued effectiveness and reliability of the ASL recognition system throughout its lifecycle.[14]

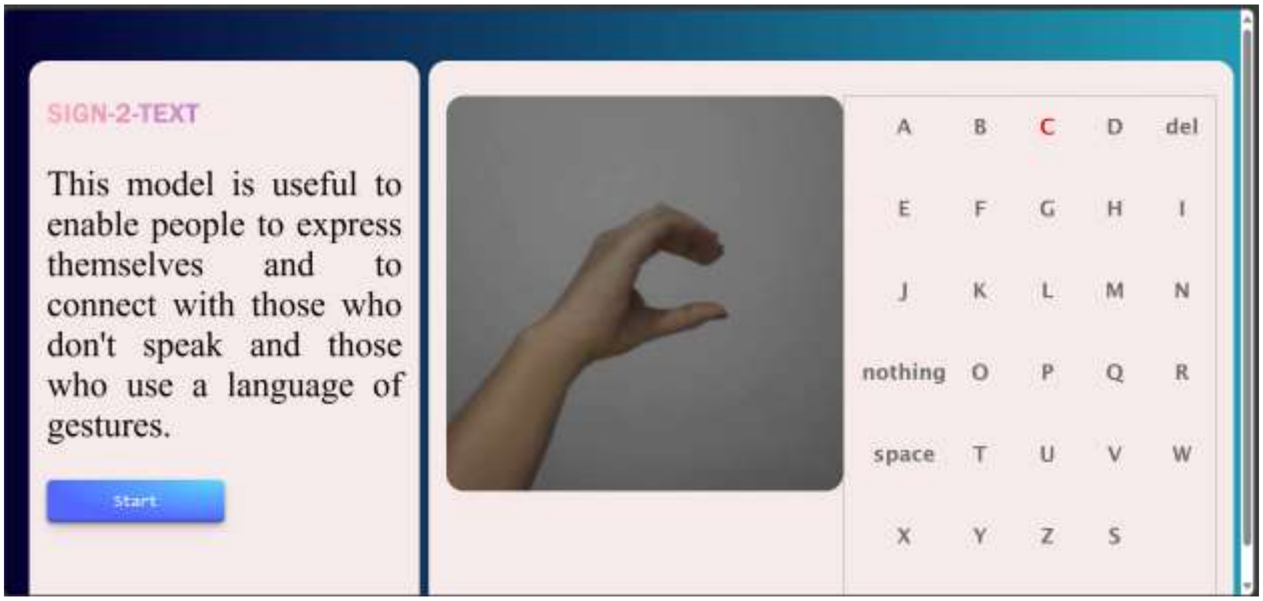
3.4.3 User Interface Development:

The development of the user interface for the ASL recognition application necessitates the creation of an intuitive and visually appealing interface that enables users to interact with the model in an effective manner. HTML, CSS, and JavaScript are frequently employed to design and style the interface elements, including input forms, buttons, and result displays. With Flask, developers can integrate HTML templates into their applications in a seamless manner, allowing for the generation and rendering of dynamic content. The process of streamlining the user interface design entails an understanding of the user's needs, the incorporation of accessibility features, and the conduction of usability testing to ensure an optimal user experience. Furthermore, it is recommended that responsive design principles be employed in order to ensure compatibility across different devices and screen sizes. This will enable users to access the ASL recognition application from various platforms.



SIGN-2-TEXT

This model is useful to enable people to express themselves and to connect with those who don't speak and those who use a language of gestures.



SIGN-2-TEXT

This model is useful to enable people to express themselves and to connect with those who don't speak and those who use a language of gestures.

Figure 15 the user interface for the ASL recognition application

3.5 Future Steps

- Integration with Assistive Technologies:

The integration of ASL recognition technology with other assistive technologies offers a multitude of potential avenues for enhancing accessibility and inclusivity. For instance, the integration of ASL recognition technology with text-to-speech systems enables the conversion of recognized ASL gestures into spoken language, providing individuals with auditory feedback in addition to visual recognition. Similarly, the integration of ASL recognition technology with sign language avatars allows users to visualize ASL gestures in animated form, facilitating communication for both deaf and hearing individuals. Furthermore, the integration of ASL recognition technology with wearable devices, such as smart glasses or haptic feedback devices, can extend the technology's reach and enable hands-free interaction and communication in various contexts.

- Deployment and Scalability:

The deployment and scalability of the system are of paramount importance.

It is of paramount importance to deploy the ASL recognition application to a scalable and reliable production environment in order to ensure the widest possible accessibility and usability. One potential avenue for achieving scalability and reliability is to explore options for cloud deployment, such as leveraging platforms like AWS, Azure, or Google Cloud. This approach can provide the necessary infrastructure to handle varying levels of usage and maintain performance under load. The use of containerization technologies, such as Docker and Kubernetes, can facilitate the streamlining of deployment and management processes. Additionally, automated scaling mechanisms can enable the dynamic adjustment of resources based on user demand. Moreover, the implementation of robust monitoring and alerting systems ensures the proactive identification and resolution of performance issues, thereby guaranteeing a seamless user experience even during periods of high traffic. [16]

- Community Engagement

It is of paramount importance to engage with the ASL community, educators, researchers, and other stakeholders in order to foster collaboration, gather valuable insights, and drive the adoption of ASL recognition technology. The establishment of partnerships with ASL organizations, educational institutions, and advocacy groups can facilitate the provision of valuable feedback and real-world use cases, thereby enabling the refinement and improvement of the technology. Participation in community events, conferences, and workshops dedicated to ASL and assistive technologies provides opportunities for networking and knowledge sharing. Moreover, the use of social media platforms, online forums, and user feedback channels can facilitate ongoing communication and collaboration with the broader ASL community, ensuring that the technology meets the evolving needs and preferences of its users. [17]

3.6 Conclusion

In conclusion, this chapter presents a comprehensive analysis of our ASL recognition model's results and implications. Through a thorough discussion of performance metrics and future directions, we aim to pave the way for the development of more robust and accurate ASL recognition systems. Furthermore, by exploring deployment strategies and outlining future steps, we strive to facilitate the seamless integration of ASL recognition technology into various applications, ultimately enhancing accessibility and inclusivity for individuals within the deaf and hard-of-hearing community.

Conclusion

Although not a global language, sign language is an essential tool for the deaf community. Communication between these communities and the hearing population is severely hampered by this, as human-based interpretation can be both costly and time-consuming. In this rapport, we present a real-time American Sign Language (ASL) generation and recognition system that makes use of Convolutional Neural Networks and deep learning (CNNs). Despite differences in lighting, skin tones, and backdrops, our technology is capable of correctly identifying and generating ASL signs. To achieve this level of accuracy, we trained our model on a large dataset of ASL signs. Our findings demonstrate that our system achieves high accuracy rates in both training and validation, with accuracy rates of 88.59%. Our approach leverages the strengths of convolutional neural networks (CNNs) to rapidly and accurately identify individual letters and words, making it particularly effective for sign fingerspelling recognition. We believe that our technology has the potential to transform communication between the hearing community and the deaf and hard-of-hearing communities by providing a reliable and cost-effective solution for sign language interpretation. The method could assist individuals who utilize sign language in communicating more effectively and residing more comfortably in a variety of settings, including educational institutions, medical facilities, and public spaces.

Bibliography

[1] Vermont Legislative Research Shop American Sign Language as a Foreign Language

[2]A Survey of Advancements in Real-Time Sign Language Translators: Integration with IoT Technology

[3]Sign Language: A Visual Language of Communication

[4]Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions

[5]GeeksforGeeks

[6]Pytorch vs Tensorflow: A Complete Breakdown

[7]Python Libraries to be used for Numerical Predictions

[8]A improved pooling method for convolutional neural networks

[9]Arabic Sign Language Recognition using Faster R-CNN

[10]A Hybrid Image Augmentation Technique for User- and Environment-Independent Hand Gesture Recognition Based on Deep Learning

[11]Maria Deprez, Emma C. Robinson, in Machine Learning for Biomedical Applications, 2024

[12]https://www.linkedin.com/pulse/convolutional-neural-networks-bi-consult-datamind

[13]Review of deep learning: concepts, CNN architectures, challenges, applications, future directions

[14]Python Web Applications: Deploy Your Script as a Flask App

[15]https://medium.com/@kennymiyasato/confused-about-the-confusion-matrix-b98a1afb00af

[16]The future of infrastructure will be containerized

[17] 15 ways to use social media for education