VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018, KARNATAKA

A Project Report
on

Sign Language Recognition using Machine Learning

Submitted in partial fulfillment of the requirements for the VIII Semester of degree of ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi*

by

Ruchi	**Sameer Kumar Singh**	**Saurav Kumar**	**Tejas V Kangod**
1RN20IS125	**1RN20IS130**	**1RN20IS140**	**1RN20IS172**

Under the Guidance of
Ms. Harshitha P
Assistant Professor
Department of ISE

Department of Information Science and Engineering

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar Post,
Channasandra, Bengaluru – 560 098

2023-2024

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar Post
Channasandra, Bengaluru – 560 098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled ***Sign Language Recognition using Machine Learning*** has been successfully completed by **Ruchi (1RN20IS125), Sameer Kumar Singh (1RN20IS130), Saurav Kumar (1RN20IS140) and Tejas V Kangod (1RN20IS172)** bonafide students of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements for the award of degree **Bachelor of Engineering** in **Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year **2023-2024**. The project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Ms. Harshitha P
Project Guide

Dr. Suresh L
Professor and HOD

Dr. Ramesh Babu H S
Principal

External Viva

Name of the Examiners — **Signature with Date**

1. ____________ **1.** ____________

2. ____________ **2.** ____________

DECLARATION

We, **Ruchi (1RN20IS125), Sameer Kumar Singh (1RN20IS130), Saurav Kumar (1RN20IS140)** and **Tejas V Kangod (1RN20IS172)** students of VIII Semester B.E.in Information Science and Engineering, RNS Institute of Technology hereby declare that the Project entitled ***Sign Language Recognition using Machine Learning*** has been carried out by us and submitted in partial fulfillment of the requirements for the *VIII Semester of degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2023-2024.

Place: Bengaluru
Date:

RUCHI	**[1RN20IS125]**
SAMEER KUMAR SINGH	**[1RN20IS130]**
SAURAV KUMAR	**[1RN20IS140]**
TEJAS V KANGOD	**[1RN20IS172]**

ABSTRACT

Sign languages are employed by many individuals with hearing impairments worldwide, and these languages exhibit a diversity of regional variations. The automated translation of sign languages is therefore viewed as essential for enhancing communication and fostering inclusion for these communities. Despite this, the heterogeneity of sign language across different regions presents significant challenges, making the development of a universal translation system complex. Yet, advancements in sign language recognition technologies hold promise for improving services for the deaf and hard of hearing by facilitating better communication and promoting social well-being. The project titled 'Real-Time Hand Gesture Detection for Sign Language Recognition using Python' seeks to create a real-time system that interprets sign language gestures into written text. Utilizing computer vision methods, the system identifies and tracks hand movements, while machine learning models, particularly the Xception architecture, are used to classify these gestures. This project will employ the Python programming language and leverage the capabilities of the OpenCV library for computer vision tasks. In our approach, we divide the project into two segments: one that uses the Xception model to process static images of hand gestures, and another that employs OpenCV for real-time detection through a webcam. The Xception model in our setup has shown impressive performance, achieving a training accuracy of 99.34% and a validation accuracy of 99.00%. The training of the hand gesture recognition model will utilize a dataset comprised of hand gesture images collected via camera. The completed system is designed to be user-friendly, enhancing interactions between sign language practitioners and those unfamiliar with sign language, which could greatly benefit communication in diverse environments such as educational institutions, workplaces, and public areas.

ACKNOWLEDGMENT

The fulfillment and rapture that go with the fruitful finishing of any assignment would be inadequate without the specifying the people who made it conceivable, whose steady direction and support delegated the endeavors with success.

We would like to profoundly thank **Management** of **RNS Institute of Technology** for providing such a healthy environment to carry out this project work.

We would like to express my thanks to our Director, **Dr. M K Venkatesha** and Principal, **Dr. Ramesh Babu H S** for his support and inspired me towards the attainment of knowledge.

We wish to place on record my words of gratitude to **Dr. Suresh L,** Professor and Head of the Department, Information Science and Engineering, for being the enzyme and master mind behind my project work.

We would like to express my profound and cordial gratitude to my Guide **Ms. Harshitha P,** Assistant Professor, Department of Information Science and engineering for their valuable guidance, constructive comments and continuous encouragement throughout the project work.

We would like to express my profound and cordial gratitude to my Coordinator **Dr. Bhagyashree Ambore,** Associate Professor, Department of Information Science and Engineering for valuable guidance in project work.

We would like to thank all other teaching and non-teaching staff of Information Science & Engineering who have directly or indirectly helped me to carry out the project work.

And lastly, we would hereby acknowledge and thank my parents who have been a source of inspiration and also instrumental in carrying out this project work.

RUCHI	**[1RN20IS125]**
SAMEER KUMAR SINGH	**[1RN20IS130]**
SAURAV KUMAR	**[1RN20IS140]**
TEJAS V KANGOD	**[1RN20IS172]**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	-Artificial Intelligence
ASL	- American Sign Language
CNN	- Convolutional Neural Network
DFD	-Data Flow Diagrams
FCLs	-Fully Connected Layers
ML	-Machine learning
NLP	- Natural Language Processing
ROI	-Region Of Interest
UML	-Unified Modelling Language
UI	- User Interface
VR	- Virtual/Virtual Reality

Chapter 1

INTRODUCTION

1.1 Deep Learning

Deep learning is a transformative technology that employs multi-layered computational models to learn data representations at various levels of abstraction. This approach has significantly enhanced capabilities in speech recognition, image recognition, object detection, and many other fields including healthcare and genomics. By utilizing backpropagation, deep learning adjusts internal parameters to transform raw data into increasingly abstract layers of representation. Deep learning excels at identifying complex patterns in large data sets, outperforming other methods in fields such as drug discovery, particle physics, brain research, and genomic analysis. It has also shown remarkable results in natural language tasks like sentiment analysis and translation. With ongoing advancements in algorithms and computing power, deep learning continues to minimize the need for manual engineering, promising further breakthroughs in various scientific and practical domains.

1.2 Supervised Learning

Supervised learning is a prevalent form of machine learning where a system is trained using a dataset of labeled examples, such as images categorized as houses, cars, people, or pets. During training, the system adjusts millions of internal parameters, or weights, to minimize the error between its predicted output and the actual category labels, using a method known as stochastic gradient descent. This involves computing a gradient that suggests how to adjust the weights to reduce the error, and iteratively updating the weights based on batches of examples. Deep learning, an advanced form of machine learning, utilizes multiple layers of non-linear processing for feature extraction and transformation, allowing the system to perform complex functions and make nuanced distinctions (like differentiating between similar animal breeds) while being robust to variations in input such as lighting or position.

1.3 Backpropagation to train multilayer architectures

From the early days of pattern recognition, the goal was to replace hand-engineered features with trainable multilayer networks, but it wasn’t until the mid-1980s that the method of training these architectures through backpropagation, a practical application of the chain rule, and the usage of the whole SLR as a subject come to a point and globally became widely understood. This involves calculating gradients for each layer in a network using stochastic gradient descent, starting from the output back to the input. Initially doubted due to fears of poor local minima, large networks have shown that such minima rarely hinder learning, with the real challenge being numerous saddle points that are manageable due to their similar objective values. Interest in deep learning surged again around 2006 with new techniques like unsupervised pre-training, helping to initialize networks effectively, particularly in areas with limited labeled data, leading to breakthroughs in speech recognition and other fields. This revival emphasized the robustness of deep feedforward networks, especially convolutional neural networks (ConvNets), which have since dominated practical applications in machine learning.

1.4 Convolutional Neural Network

Convolutional Neural Networks (ConvNets) are tailored to process multi-array data like images (2D arrays) or sequences (1D arrays). They capitalize on four key principles: local connections, shared weights, pooling, and multiple layers. In ConvNets, convolutional layers detect local patterns via filters that scan the input array, while pooling layers aggregate similar features to reduce dimensionality and enhance invariance to shifts and distortions. This architecture exploits the hierarchical nature of natural signals, where higher-level features are compositions of lower-level ones, mirroring concepts from visual neuroscience. Backpropagation efficiently trains ConvNets, and their performance aligns with neural activities observed in primate visual systems. ConvNets have a rich history, with early applications dating back to the 1990s in speech recognition, document reading, and object detection, showcasing their versatility and effectiveness across diverse domains.

1.5 Image understanding with deep convolutional networks

Convolutional Neural Networks (ConvNets) have revolutionized computer vision and image understanding, dominating recognition and detection tasks since their success in the 2012 ImageNet competition. Their efficient use of GPUs, ReLUs, and regularization techniques like dropout significantly improved performance, leading to their adoption by major tech companies for various applications such as face recognition, autonomous vehicles, and natural language processing. ConvNets boast architectures with multiple layers of ReLUs, millions of weights, and billions of connections, now trainable in a matter of hours thanks to advancements in hardware and software. Their success has spurred extensive research and deployment across industries, with efforts underway to implement efficient hardware solutions for real-time applications. Deep learning theories highlight ConvNets' exponential advantages in distributed representations, facilitating generalization to new data combinations and enabling efficient composition of layers for predictive modeling in language processing task.

1.6 Machine Learning

Machine Learning is a system of computer algorithms that can learn from example through self-improvement without being explicitly coded by a programmer. Machine learning is a part of artificial Intelligence which combines data with statistical tools to predict an output which can be used to make actionable insights. The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input and uses an algorithm to formulate answers. A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation. Machine learning is also used for a variety of tasks like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

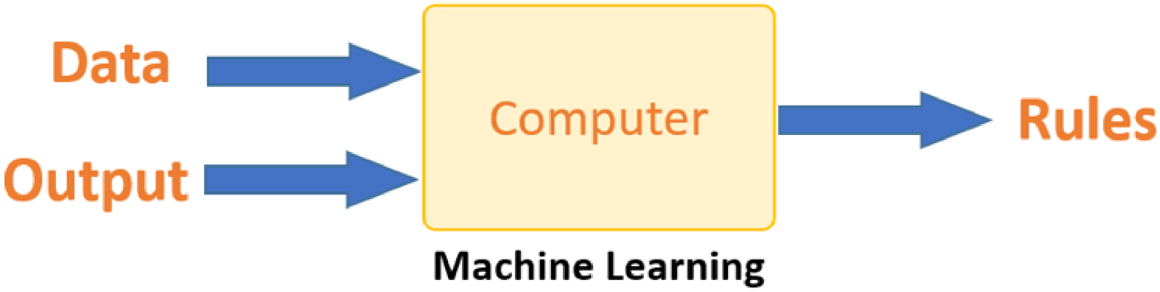


Fig 1.1 Login Machine Learning

1.7 Training and Testing

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it’s feed a previously unseen example, the machine has difficulties to predict. The core objective of machine learning is the learning and inference. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a feature vector.

Machine learning serves as the brain behind all learning processes within computational systems. It mimics the human learning process, which heavily relies on experience. Just as humans learn from exposure to various situations, machines learn from data. The more data they are exposed to, the better they become at making predictions. This concept mirrors the human experience: familiarity breeds proficiency, while unfamiliar situations pose challenges.

Training a machine learning model involves exposing it to examples of input data along with their corresponding output labels. Through this process, the machine identifies patterns and relationships within the data, enabling it to make accurate predictions. However, similar to humans, machines struggle when faced with previously unseen examples. This underscores the importance of robust training datasets that encompass a diverse range of scenarios.

The core objective of machine learning lies in its ability to learn from data and make inferences. This learning process hinges on the discovery of patterns within the provided data. Data scientists play a pivotal role in this process by curating and selecting relevant data to feed into the machine learning model. The selection of attributes, known as a feature vector, is crucial in addressing specific problem domains and optimizing model performance.

In essence, machine learning operates on the principle of pattern recognition and inference, akin to the way humans learn and make decisions based on past experiences. By leveraging large datasets and sophisticated algorithms, machine learning algorithms can unlock insights and predictions from complex data, ultimately driving innovation and problem-solving across various domains.

Training and testing are essential phases in the machine learning workflow. During the training phase, the machine learning model is exposed to a significant amount of labeled data to learn the underlying patterns and relationships. This process involves iterative adjustments to the model's parameters until it achieves satisfactory performance on the training dataset. However, to evaluate the model's generalization ability and assess its performance on unseen data, a separate testing phase is conducted. In the testing phase, the trained model is evaluated using a separate dataset called the testing dataset, which contains examples not seen during training. This allows for an unbiased assessment of the model's performance and its ability to make accurate predictions on new, unseen instances. Various metrics such as accuracy, precision, recall, and F1 score are commonly used to evaluate the model's performance during testing. We can after all the testing can use it into a model. Therefore, the learning stage is used to describe the data and summarize it into a model.

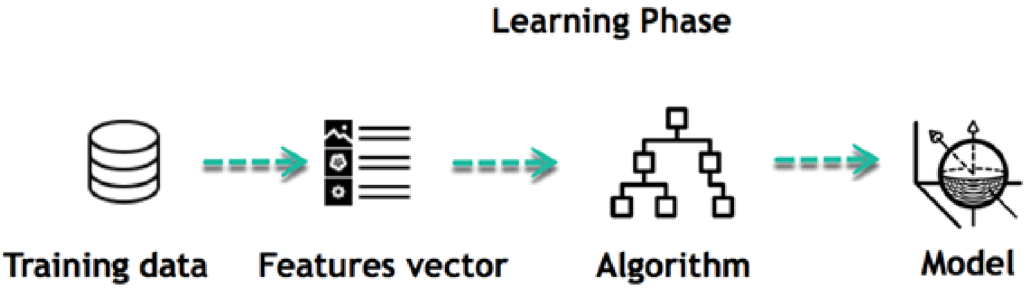


Fig 1.2 Training the Model

For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model

Inferring

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

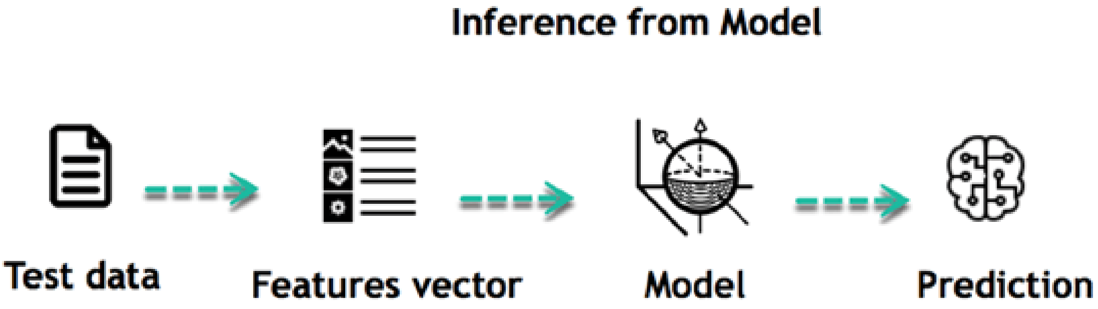


Fig 1.3 Testing the Model

The life of Machine Learning programs is straightforward and can be summarized in the following points:

1. Define a question
2. Collect data
3. Visualize data
4. Train algorithm
5. Test the Algorithm
6. Collect feedback
7. Refine the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction

Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data.

Consider a scenario where a machine learning model is tasked with understanding the relationship between an individual's wage and their likelihood of dining at a fancy restaurant. Through the training process, the model discovers a positive correlation between higher wages and the propensity to visit high-end dining establishments. This correlation forms the basis of the model's understanding of the relationship between wage and restaurant choice.

Once the model is trained, it can be tested on previously unseen data to evaluate its performance and predictive capabilities. New data points are transformed into feature vectors, which are then inputted into the trained model to generate predictions. This aspect of machine learning is particularly fascinating as it demonstrates the model's ability to generalize its learnings from the training phase to make accurate predictions on new, unseen instances. Importantly, there is no need to modify the model's rules or retrain it; the previously trained model can be utilized for inference on new data.

Once the model has completed its training, it stands ready to apply its newfound knowledge to real-world scenarios. When presented with fresh data, the model seamlessly transforms it into a format it can understand—feature vectors—before employing its learned patterns to make predictions. This ability to infer conclusions from unseen data exemplifies the power and versatility of machine learning. Unlike traditional rule-based systems that require constant updating, a well-trained machine learning model can autonomously apply its insights to new situations, making predictions with remarkable accuracy and efficiency.

The lifecycle of a machine learning program is akin to a continuous journey of discovery and refinement. It begins with a clear question or problem statement, driving the collection and exploration of relevant data. Through visualization and analysis, insights emerge, guiding the training of the machine learning algorithm. The testing phase serves as a crucial checkpoint, validating the model's performance on unseen data and providing invaluable feedback for further improvement.

Chapter 2

LITERATURE SURVEY

A literature survey or a literature review in a project report shows the various analyses and research made in the field of interest and the results already published, taking into account the various parameters of the project and the extent of the project. Literature survey is mainly carried out in order to analyze the background of the current project which helps tofind out flaws in the existing system & guides on which unsolved problems we can work out. So, the following topics not only illustrate the background of the project but also uncover the problems and flaws which motivated to propose solutions and work on this project.

A literature survey is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature reviews use secondary sources, and do not report new or original experimental work. Most often associated with academic-oriented literature, such as a thesis, dissertation or a peer-reviewed journal article, a literature review usually precedes the methodology and results sectional though this is not always the case. Literature reviews are also common in are search proposal or prospectus (the document that is approved before a student formally begins a dissertation or thesis). Its main goals areto situate the current study within the body of literature and to provide context for the particular reader. Literature reviews are a basis for researching nearly every academic field. demic field. A literature survey includes the following:

- Existing theories about the topic which are accepted universally.
- Books written on the topic, both generic and specific.
- Research done in the field usually in the order of oldest to latest.
- Challenges being faced and on-going work, if available.

Literature survey describes about the existing work on the given project. It deals with problem associated with the existing system and also gives user a clear knowledge on howto deal with the existing problems and how to provide solution to the existing problems sothat we can tackle the problem, resolve them in better way by implementing advanced techniques.

Objectives of Literature Survey

- Learning the definitions of the concepts.
- Access to latest approaches, methods and theories.
- Discovering research topics based on the existing research
- Concentrate on your own field of expertise– Even if another field uses thesame words, they usually mean completely.
- It improves the quality of the literature survey to exclude sidetracks–Remember to explicate what is excluded.

Before building our application, the following system is taken into consideration:

Paper Summaries

Title: Recent Advances and Resources in Vision-Based Hand Gesture Recognition: An Overview

Author: Rafiqul Zaman Khan ,Noor Adnan Ibraheem

Year: 2012

Abstract: Over the past twenty years, significant strides in hand gesture recognition research have facilitated advancements in human-computer interaction. However, persistent challenges such as accurate phase detection of gestures, variation in size, shape, and speed, along with occlusion issues, maintain the field’s dynamism and relevance. This review covers the algorithms for vision-based hand gesture recognition reported over the past 16 years, focusing on those using RGB and RGB-D cameras. It includes both quantitative and qualitative analyses of these algorithms based on 13 criteria derived from various attributes of the algorithms and their evaluation methods. The review underscores the importance of these criteria alongside algorithm accuracy to predict real-world applicability. Additionally, we discuss 26 publicly accessible hand gesture databases, providing URLs for their download.

Strengths: The paper offers detailed insights into the extraction methods, features extraction, and classification tools used in hand gesture recognition systems, providing a thorough understanding of the subject matter.

Weakness: Some drawbacks of the discussed methods include issues with orientation histograms, time-consuming neural network classifiers, and limitations related to environmental lighting changes.

Title: Hand Gesture Recognition System Using Camera

Author: Viraj Shinde, Tushar Bacchav, Jitendra Pawar, Mangesh Sanap.

Year: 2014

Abstract: This paper provides the first comprehensive review of sign language recognition studies from 2007 to 2017, addressing a significant gap in the academic literature. Out of 396 articles initially identified, 117 were thoroughly reviewed and classified across six dimensions, including data acquisition methods, sign types, signing mode, hand usage, classification techniques, and recognition accuracy. The findings offer valuable insights to inform future research in this important field.

Strengths: The research focuses on developing a real-time hand gesture recognition system based on adaptive color HSV model and motion history image (MHI), which can greatly improve the robustness of hand gesture recognition.

Weakness: The proposed system may face challenges in accurately detecting hand objects in digital images or videos, especially in real-world environments with unstable brightness, noise, poor resolution, and contrast.

Title: Exploring Techniques in Hand Gesture and Sign Language Recognition

Author: M.J. Cheok, Z. Omar, M.H. Jaward

Year: 2012

Abstract: Hand gesture recognition is pivotal for enhancing user convenience and overcoming challenges in human-machine interaction, particularly in the realm of sign language recognition. This paper exhaustively reviews cutting-edge methods in hand gesture and sign language recognition research. The approaches are organized by stages of data handling—from acquisition and preprocessing to segmentation, feature extraction, and classification. Each stage’s techniques are detailed, with a focus on their strengths. The paper also addresses the broader challenges in gesture recognition and specific issues related to sign language recognition. The overview aims to furnish a comprehensive foundation for new research in automated gesture and sign language recognition.

Strength: The inclusion of benchmark databases allows for direct comparison between different algorithms used in the research.

Weakness: Lack of Detailed Pre-Processing Information: The paper notes that many

papers lack detailed information on the pre-processing stage, which could limit the reproducibility and comparability of the research.

Title : Enhancing Hand Gesture Recognition with Key Frame Extraction and Feature Fusion Techniques

Author : Hao Tang, Wei Xiao, Hong Liu, Nicu Sebe

Year : 220

Abstract: Gesture recognition is an integral part of natural user interfaces in computer vision and pattern recognition, yet achieving both rapid and robust recognition remains challenging. This study introduces a novel approach that integrates image entropy and density clustering to identify key frames in gesture videos, enhancing processing speed and efficiency. Additionally, a feature fusion strategy is proposed to boost the robustness of feature representation. The effectiveness of these methods is demonstrated through experiments on the Northwestern University, Cambridge, Hand Gesture, and Action3D datasets, where our strategies show competitive performance.

Strength: This work combines image entropy and density clustering to exploit the key frames from hand gesture video for further feature extraction, which can improve the efficiency of recognition

Weakness: Fast and robust hand gesture recognition remains an open problem, since the existing methods have not well balanced the performance and the efficiency simultaneously

Title: Introducing the American Sign Language Lexicon Video Dataset

Author : V. Athitsos, C. Neidle, S. Sclaroff, J. Nash, A. Stefan, Q. Yuan, A. Thangali

Year : 2020

Abstract :- The absence of a written form for American Sign Language (ASL) complicates tasks like dictionary look-ups of unknown signs, which are typically organized by their closest English equivalent in printed dictionaries. This paper presents the ASL Lexicon Video Dataset, an extensive and growing public dataset featuring video sequences of thousands of distinct ASL signs, complete with annotations including start/end frames and class labels for each sign. Created to support the development of a computer vision system for ASL sign look-up, this dataset also serves as a benchmark for various computer vision and machine learning applications, particularly those focused on gesture and human communication analysis.

Strengths: The baseline experimental results, while simple, provide encouraging results and highlight the potential for improving upon them, motivating further research in the field.

Weakness: The motion energy images method used for sign retrieval does not perform very well over the entire test set, indicating the need for more advanced methods to improve accuracy.

Chapter 3

ANALYSIS

3.1 Problem Statement

Despite advancements in technology, communication barriers persist for individuals with hearing impairments, particularly in scenarios where sign language interpretation is necessary. Traditional methods of sign language recognition often lack real-time capabilities and struggle with accuracy, hindering effective communication between sign language users and non-sign language users. There is a need for a robust and efficient sign language recognition system that can accurately interpret hand gestures in real-time, bridging the communication gap and promoting inclusivity for individuals with hearing impairments

3.2 Aims of the Project

- The aim of the project is to create a comprehensive human-computer interface that enables users to interact with devices without requiring physical touch or traditional input methods.
- The project aims to achieve this by combining various technologies, such as virtual hand gestured keyboard, mouse, voice-based UI, and hand gesture-based UI, to create a seamle ss and intuitive interface for users.
- Mediapipe and deep learning are proposed to be used in the development of the hand gesture-based UI component, allowing for accurate and reliable detection of hand movements and gestures.
- OpenCV will also be utilized in the development of the system to provide robust computer vision capabilities, such as face detection and tracking, which canenhance the overall user experience.
- The ultimate goal of the project is to create a system that can improve accessibilityfor users with disabilities or mobility impairments, as well as enhance the user experience for all users by providing a more natural and intuitive interface.

3.3 Methodology

3.3.1 Existing System

2019 paper by Akshay V and Deepak Kumar, the focus was on detecting both static and dynamic gestures performed by individuals. They aimed to construct a Python-like syntax for various commands such as if and if-else statements, with minimal processing time. This was achieved by considering fundamental programming keywords and utilizing a custom dataset consisting of diverse images of gestures corresponding to different words to detect and interpret gestures from a sign language dictionary, they employed a Convolutional Neural Network (CNN) model. This model aimed to identify the words being communicated through the gestures. Their approach involved real-time management using Principal Component Analysis (PCA) for gesture extraction and pattern recognition with stored images. By combining computer vision and deep learning algorithms, including CNNs customized for the task, they aimed to bridge communication gaps for hearing-impaired individuals. Their primary focus was on tracking hand and finger movements to gather information about gestures.

The system utilized a camera-based setup to capture data about the gestures, which was proposed to be transitioned from a 2D to a 3D space for a more comprehensive understanding. A Greedy Algorithm was implemented to predict essential frames and eliminate redundant predictions, with the goal of enabling the model to generate code resembling if and loop structures. The model was trained on various images from a custom English-words dataset using some optimization techniques. Additionally, the system aimed to handle both one-handed and two-handed gestures within the same model. It was noted that a diverse dataset with more variations could enhance the model's robustness to different sign language variations and significantly improve performance.

3.3.2 Disadvantages of Existing System

- Limited Gesture Vocabulary: The system relies on a custom dataset of gestures for individual words, potentially leading to a restricted gesture vocabulary.
- Dependency on 2D to 3D Conversion: The proposed conversion process may introduce complexities and inaccuracies due to the challenge of estimating depth information from 2D images.
- Greedy Algorithm for Prediction: The use of a Greedy Algorithm may not always result in the most accurate predictions, impacting gesture recognition reliability.
- Limited Optimization Techniques: The system lacks specific details about optimization techniques employed, which could affect model performance.
- Potential Sensitivity to Variations: The system's effectiveness in handling various sign language variants and variations is not explicitly addressed.
- Limited Mention of System Usability: Information regarding the system's usability and user-friendliness is not provided.
- Lack of Comparative Performance Evaluation: The system does not compare its

performance with existing systems or benchmark datasets, making it challenging to assess its effectiveness.

3.3.3Proposed System

The proposed system aims to develop a real-time hand gesture detection system for sign language recognition using Python, leveraging computer vision techniques and machine learning algorithms. It will be developed using the Python programming language and the OpenCV library for computer vision tasks. Hand gesture recognition will be performed using the Xception architecture model, trained on a dataset of hand gestures captured via camera.

Key Features of the Proposed System:

- Utilization of the Xception architecture model for hand gesture detection and classification.
- Real-time hand gesture detection using computer vision techniques and OpenCV.
- High accuracy in recognizing gestures such as "Ok, Open Hand, Peace, Thumb, and No Hand Detected".
- Evaluation based on accuracy, real-time performance, and usability.

3.4 System Requirements

HARDWARE REQUIREMENTS

•	System	:	Pentium i3 Processor.
•	Hard Disk	:	500 GB.
•	Monitor	:	15’’ LED
•	Input Devices	:	Keyboard, Mouse
•	Ram	:	4 GB

SOFTWARE REQUIREMENTS:

•	Operating system	:	Windows 10.
•	Coding Language	:	Python.
•	Web Framework	:	Flask.

Chapter 4

SYSTEM DESIGN

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development.

Systems design implies a systematic approach to the design of a system. It may take a bottom-up or top-down approach, but either way, the process is systematic wherein it takes into account all related variables of the system that needs to be created—from the architecture, to the required hardware and software, right down to the data and how it travels and transforms throughout its travel through the system. Systems design then overlaps with systems analysis, systems engineering, and systems architecture.

4.1 System Architecture

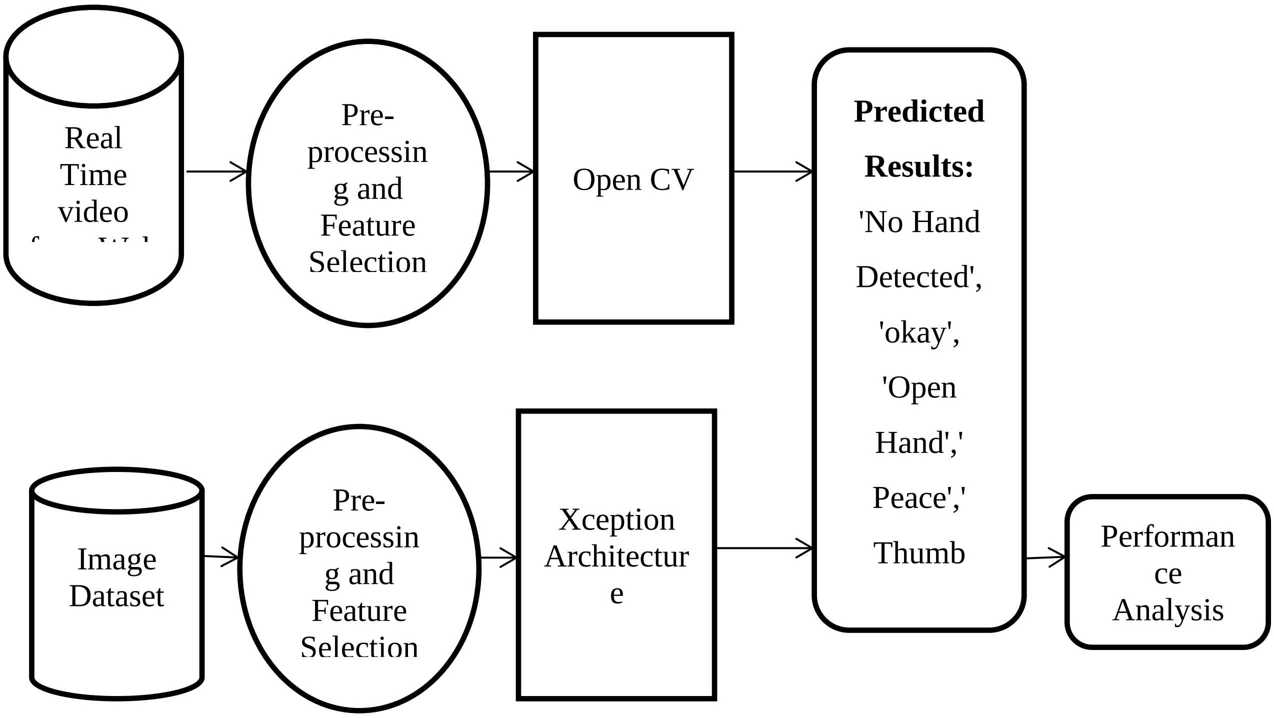

Real Time video
Pre-processing and Feature Selection
Open CV
Predicted Results: 'No Hand Detected', 'okay', 'Open Hand',' Peace',' Thumb
Image Dataset
Pre-processing and Feature Selection
Xception Architecture
Performance Analysis

Fig 4.1 System Architecture

4.2 Data Flow Diagram

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
- The data flow diagram (DFD) is one of the most important modeling tools. It is used

to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

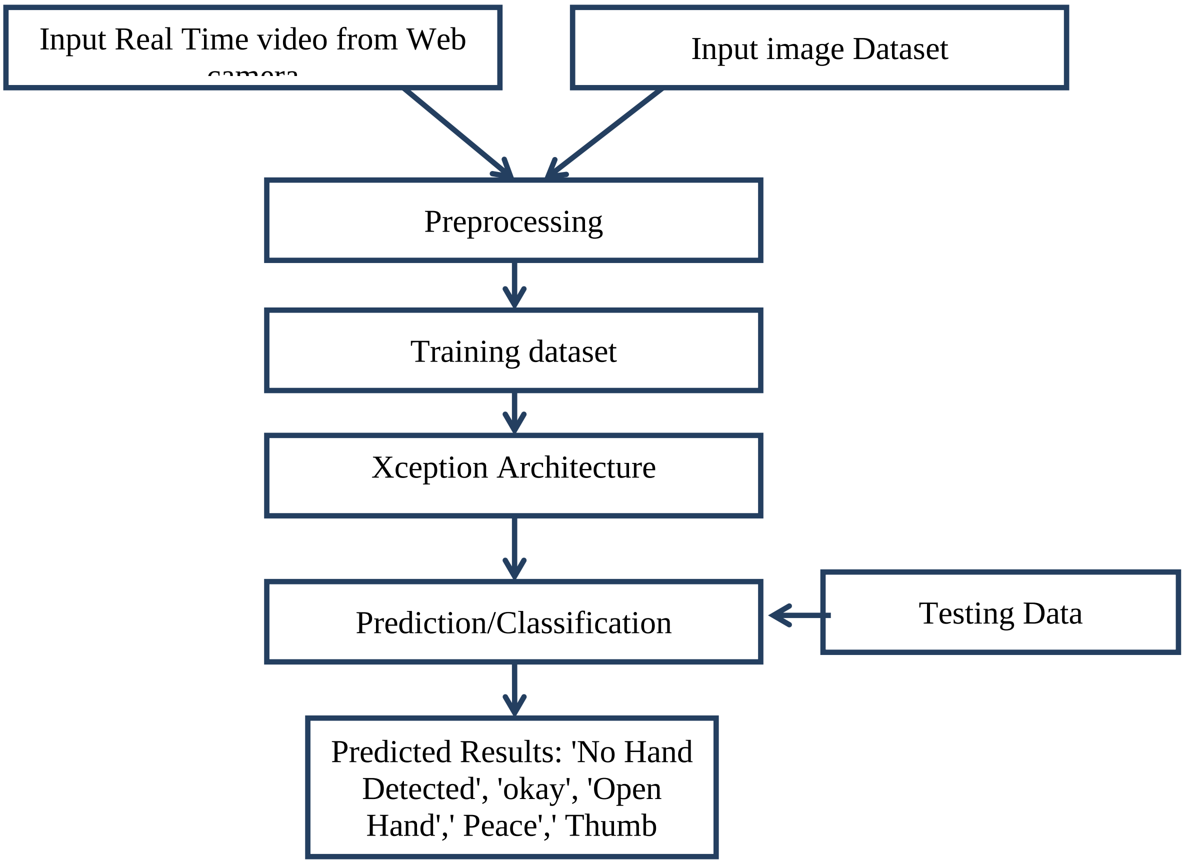

Input Real Time video from Web camera
Input image Dataset
Preprocessing
Training dataset
Xception Architecture
Prediction/Classification
Testing Data
Predicted Results: 'No Hand Detected', 'okay', 'Open Hand',' Peace',' Thumb

Fig 4.2 Dataflow diagram of System Architecture

4.3 UML Diagram

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.4 Goals:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

4.5 Use Case Diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor.

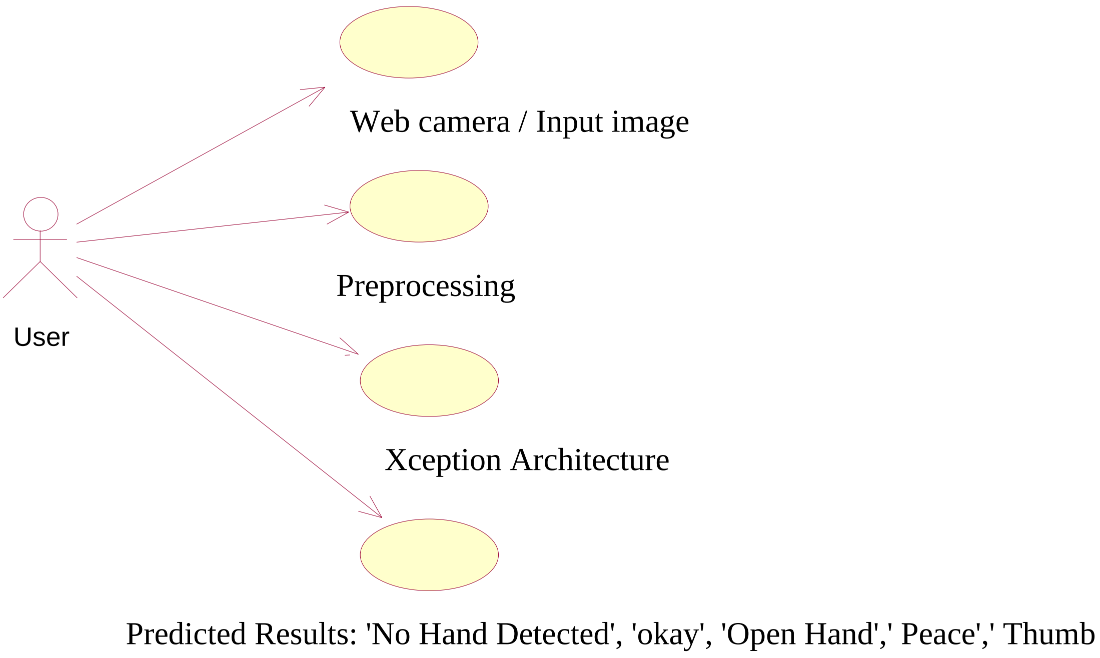

Web camera / Input image
Preprocessing
User
Xception Architecture
Predicted Results: 'No Hand Detected', 'okay', 'Open Hand',' Peace',' Thumb

Fig 4.3 Use case diagram

4.6 Class Diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

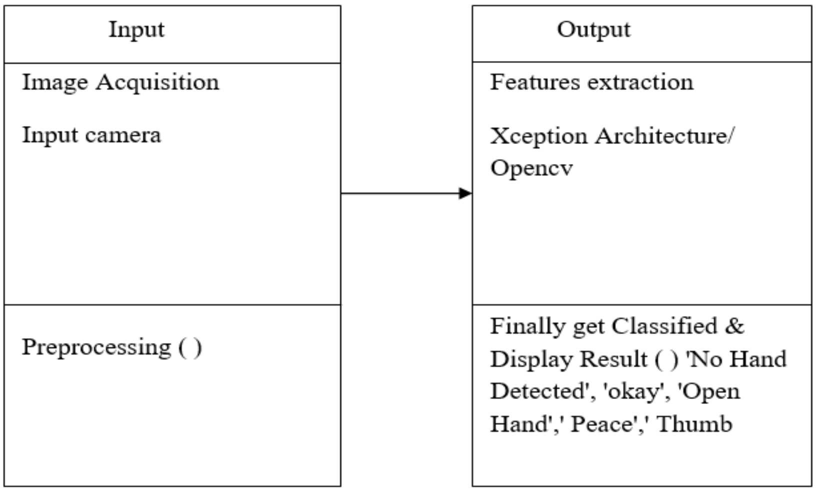

Input
Image Acquisition
Input camera
Preprocessing ( )
Output
Features extraction
Xception Architecture/ Opencv
Finally get Classified & Display Result ( ) 'No Hand Detected', 'okay', 'Open Hand',' Peace',' Thumb

Fig 4.4 Class Diagram

4.7 Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

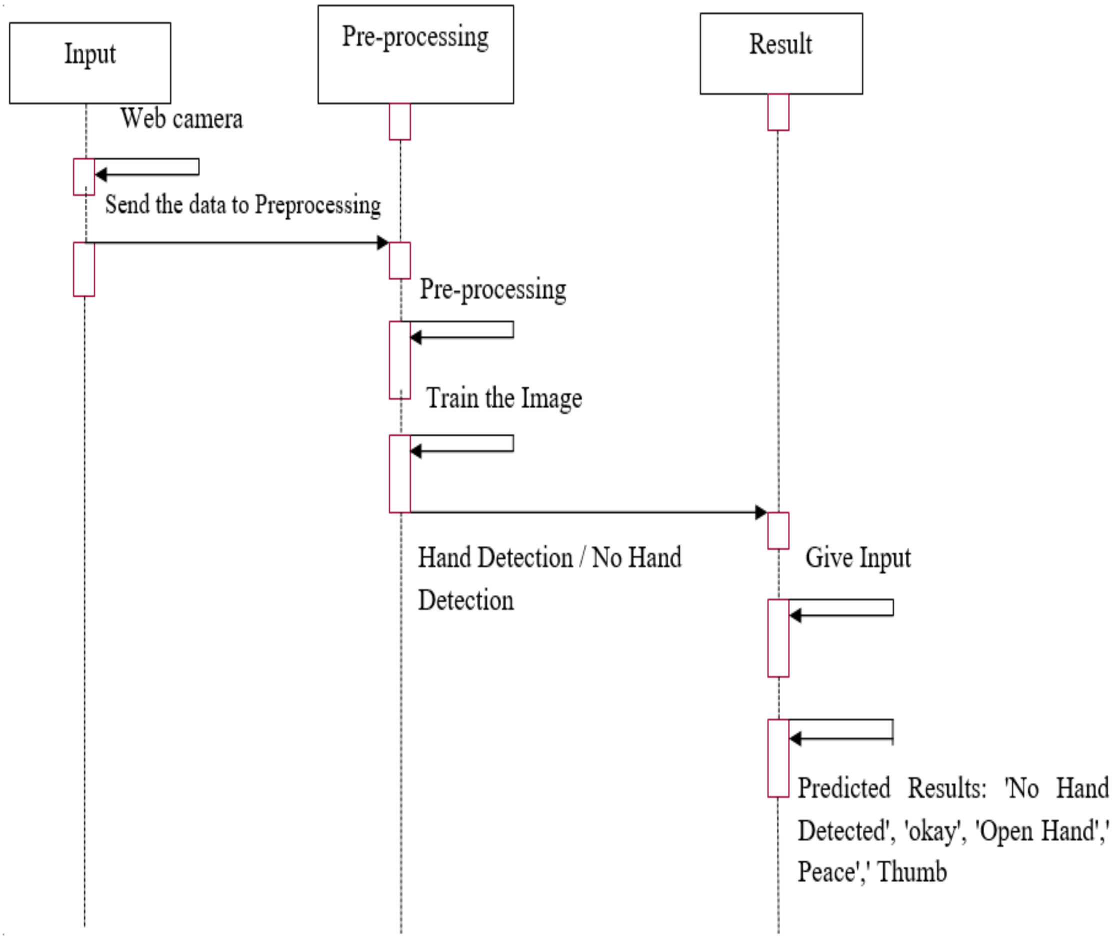

Input
Pre-processing
Result
Web camera
Send the data to Preprocessing
Pre-processing
Train the Image
Hand Detection / No Hand Detection
Give Input
Predicted Results: 'No Hand Detected', 'okay', 'Open Hand',' Peace',' Thumb

Fig 4.5 Sequence Diagram

4.8 Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

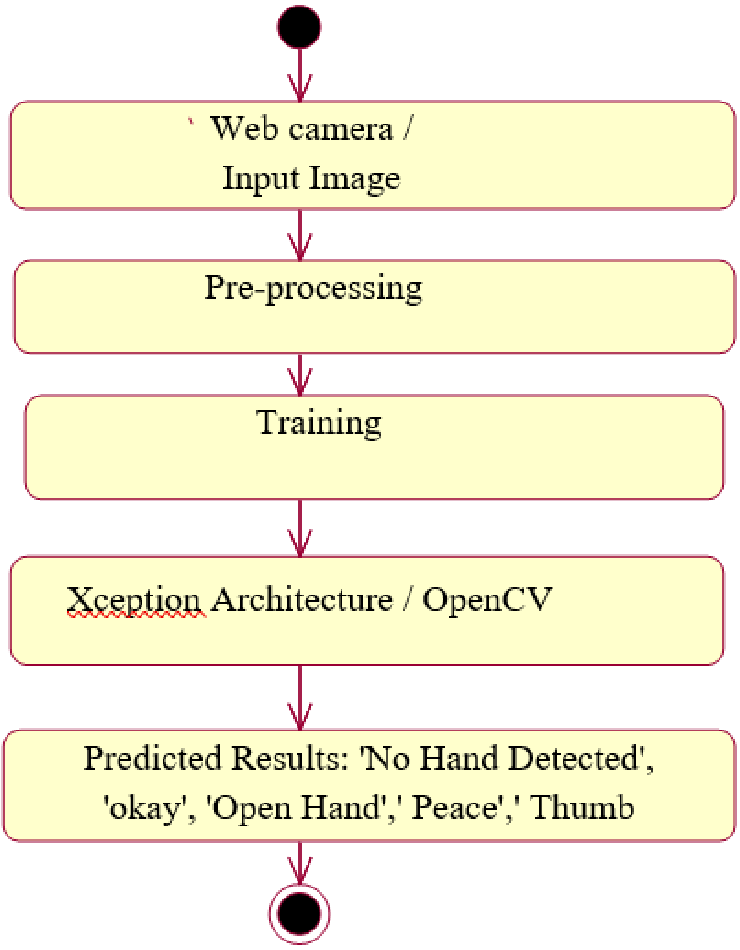

Web camera /
Input Image
Pre-processing
Training
Xception Architecture / OpenCV
Predicted Results: 'No Hand Detected',
'okay', 'Open Hand',' Peace',' Thumb

Fig 4.6 Activity Diagram

4.9. Design Of Xception Model

The objective of this system is to take an image of a gesture as input and then identify what it means. Currently the model can correctly identify 5 signs

- Open Hand
- OK
- Peace
- No hand detected
- Thumbs Up

A Transferable training approach is used to train the model. This means that the weights of the nodes are updated every time a model is applied on the neural network. This is done to adjust the nodes' weights to perform desired function. A Transferable training approach is used to train the model. This means that the weights nodes are updated every time a model is applied on the neural network. This is done to adjust the nodes' weights to perform desired function. This is carried out by two algorithms.

- **MobileNet**: This model is initially applied on the network. Other than itself, it consists of 2 more layers which are
- **GlobalAveragePooling2D** : This layer takes a list of values as input and averages
- output a single value. This is done to prevent overfitting
- **Dense Layer**: This layer performs the classification to classify the hand gesture to the appropriate class
- **vgg_16 model:** The VGG-16 model is a deep convolutional neural network (CNN) known for its simplicity and depth. Developed by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group (VGG) at the University of Oxford, It consists of 16 weighted layers consisting of input layer, convolutional layer, pooling layer, fully connected layer and output layer.

Finally, the Xception architecture is used as the base model. This is added to the network using the MobileNet while allows network layers to be built sequentially or layer by layer. This is responsible for processing the image and detect the correct gesture.

4.10. Open Cv Model

This model is used to detect sign language gestures in real time. It consists of the following systems:

- **OpenCV**: This library of python is used to utilize the web camera of the system to capture video of the environment. OpenCV provides a wide range of tools and algorithms that can be used for tasks such as image and video analysis, face recognition, motion tracking, object detection, and augmented reality. The video captured is used as the input.
- **MediaPipe**: This library of python is used for processing the images captured by the camera. The Hands Module of MediaPipe is used to detect hand of person from the image. Once this is done, The utils and Connections library is used to draw specific pattern on the hand. This is done by making use of nodes and edges. Different patterns are drawn based on the hand gesture.
- **KeyPointClassifier**: This model is used to classify the pattern drawn by MediaPipe to appropriate class of hand gesture. A dictionary is used to store index values (0,1,2,…) as keys and hand gestures are mapped them. Once a pattern is detected, It finds the respective class of the hand gesture and displays it as output.

Chapter 5

IMPLEMENTATION

5.1 Dataset:

In the first module, we developed the system to get the input dataset for the training and testing purpose. Dataset is given in the model folder. The dataset consists of 13,407 Hand gesture images. The dataset is referred from the popular dataset repository called Kaggle. The following is the link of the dataset:

Kaggle Dataset Link: https://www.kaggle.com/datasets/jayaprakashpondy/hand-gesture-dataset

5.2 Importing the necessary libraries:

This module involves importing the necessary libraries such as NumPy, TensorFlow to implement the model. We will be using Python language for this. First we will import the necessary libraries such as keras for building the main model, sklearn for splitting the training and test data, PIL for converting the images into array of numbers and other libraries such as pandas, numpy, matplotlib and tensorflow.

5.3 Retrieving the images:

We will retrieve the images and their labels. Then resize the images to (224,224) as all images should have same size for recognition. Then convert the images into numpy array.

5.4 Splitting the dataset:

This module involves splitting the preprocessed dataset into training and testing sets for model development and evaluation. Split the dataset into train and test. 80% train data and 20% test data.

5.5 Building the model:

5.5.1 Xception model

- Inspired by Google's Inception model
- Xception is based on an 'extreme' interpretation of the Inception model
- The Xception architecture is a linear stack of depth wise separable convolution layers

5.5.2 Depth wise Separable Convolution

Regular Convolutions:

- look at both channel & spatial correlations simultaneously

Depth wise separable convolution:

- look at channel & spatial correlations independently in successive steps
- spatial convolution: 3x3 convolutions for each channel
- depth wise convolution: 1x1 convolutions on concatenated channels

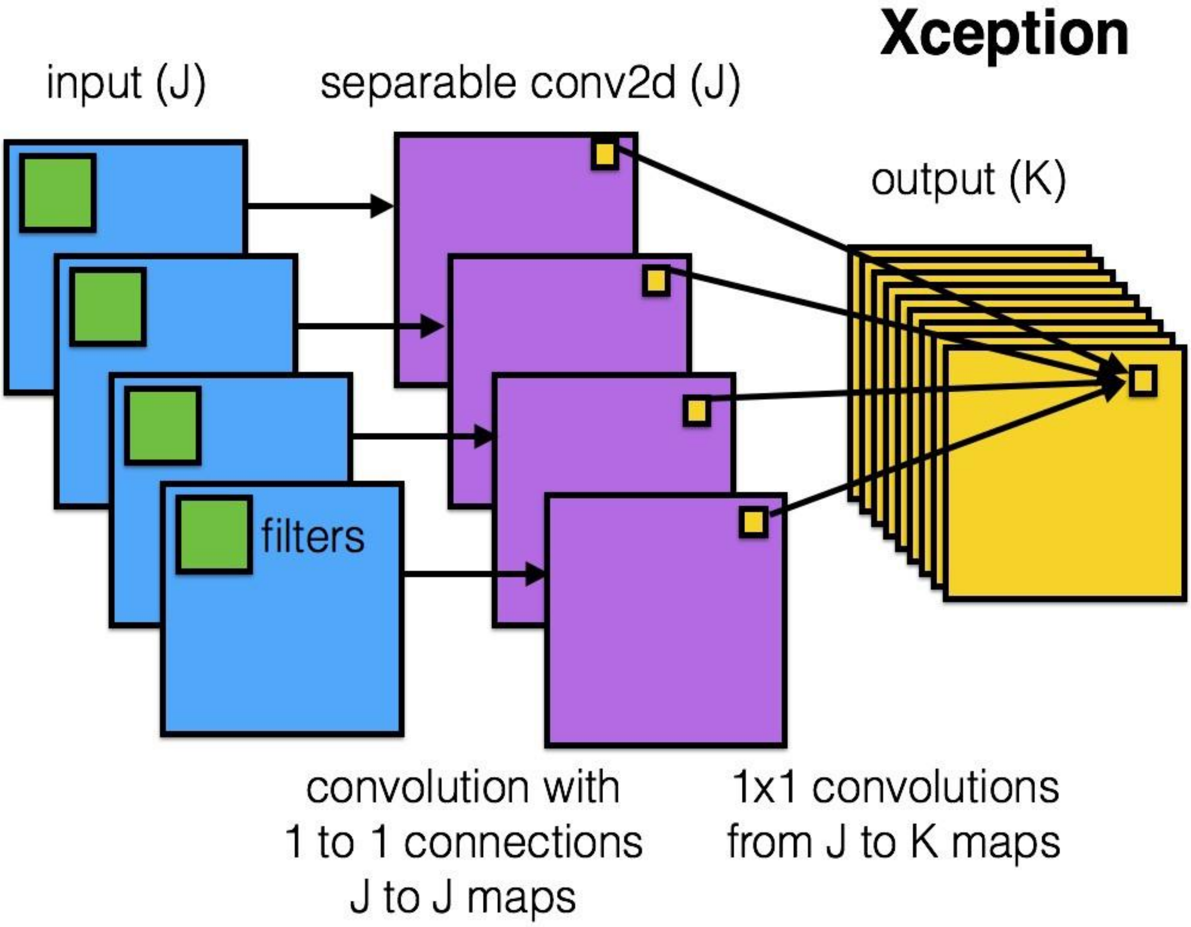


Fig 5.1 Xception Model

Example: take 3x3 convolutional layer on 16 input channels and 32 output channels.

- Regular convolution: 16x32x3x3 = 4608 parameters.
- Easy to map data.
- Depth wise separable convolution: (spatial conv + depth wise conv) = (16x3x3 + 16x32x1x1) = 656 parameters

- Greatly reduced parameter count
- More efficient complexity
- Maintains cross-channel features

5.5.3 Inception Model

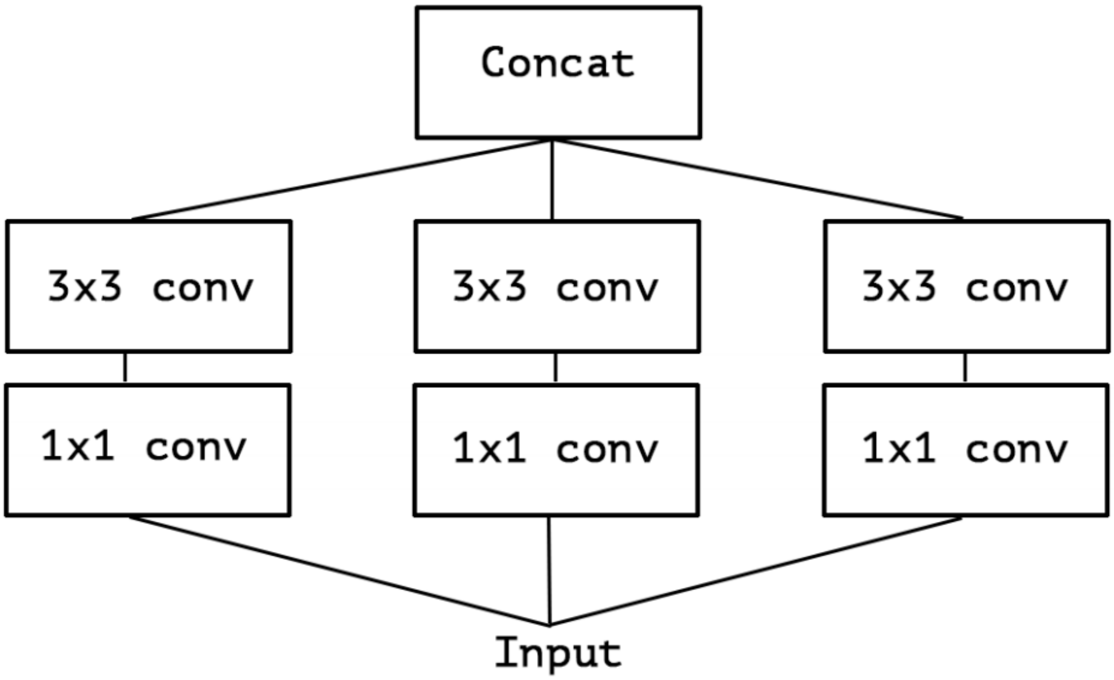


Fig 5.2 Inception Model

- Fundamental hypothesis: cross-channel correlations and spatial correlations are sufficiently decoupled
- First looks at cross channel correlations via a set of 1x1 convolutions.
- Then acts as a “multi-level feature extractor” by computing 1×1, 3×3, and 5×5 convolutions
- Output feature maps are stacked along the channel dimension

5.5.4 “extreme” version of Inception module:

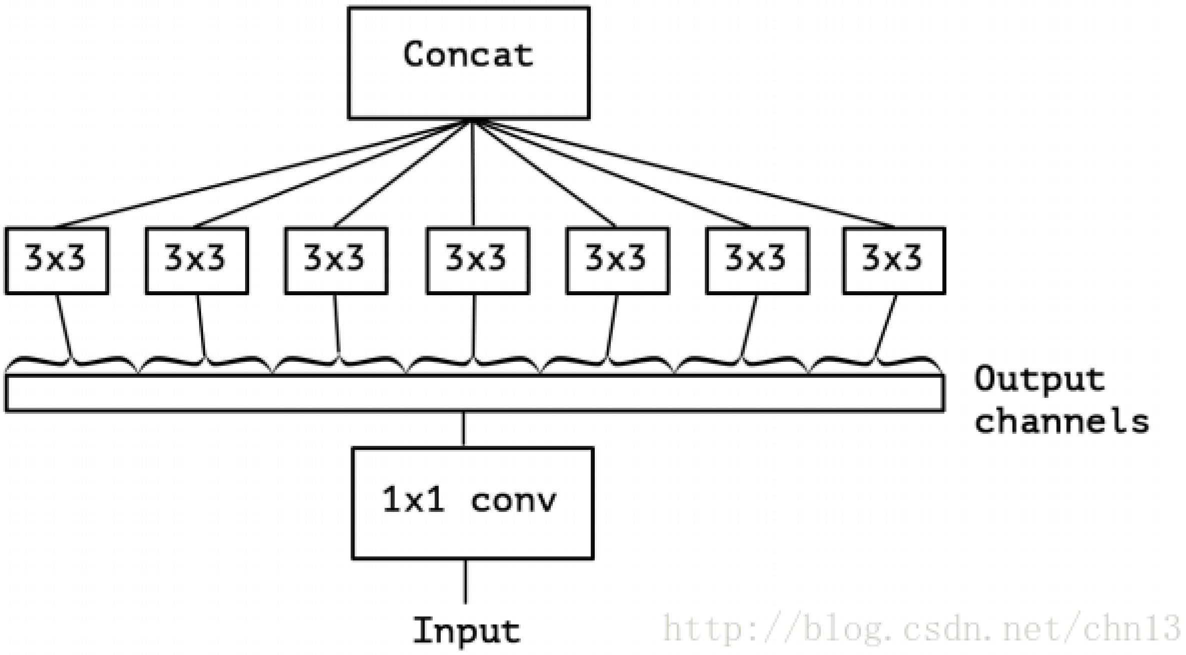


Fig 5.3 Extreme version of Inception Model

- First use a 1x1 convolution to map cross-channel correlations
- Then separately map the spatial correlations of every output channel (instead of just 3-4 partitions)
- Similar to depth wise separable convolution

5.5.5 What makes depth wise separable convolution different

Two differences between and "extreme" version of an Inception module and a depth wise separable convolution

The order of the operations:

- Depth wise separable convolutions perform first channel-wise spatial convolution and then perform 1x1 depth wise convolution on the output.
- Non-linearity after the first operation:
- In Inception both operations are followed by a ReLU non-linearity
- In depth wise separable convolutions don't have intermediate non-linearities

5.5.6 Xception architecture

- convolutional neural network architecture based entirely on depth wise separable convolution layers.

- fundamental hypothesis: mapping of cross-channels correlations and spatial correlations can be entirely decoupled.
- composed of 36 convolutional layers forming the feature extraction base of the network
- structured into 14 modules, all of which have linear residual connections around them, except for the first and last modules.

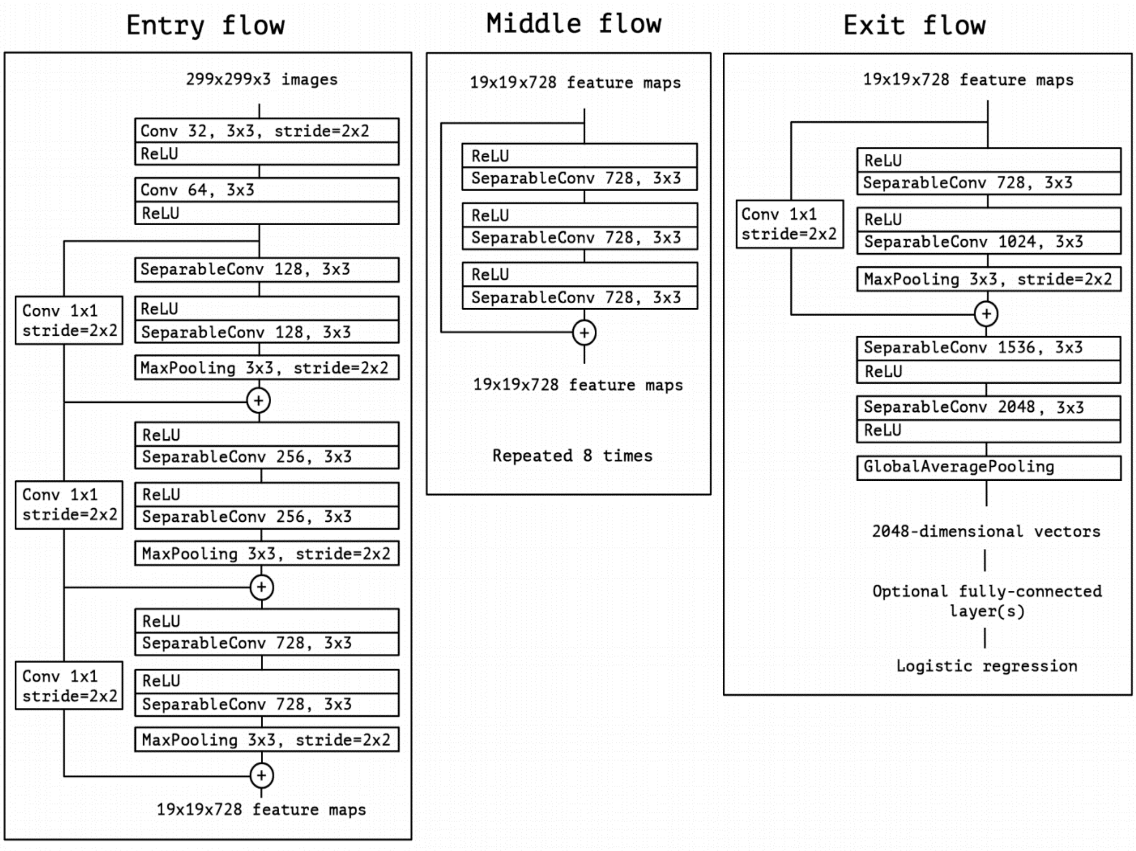


Fig 5.4 Xception Architecture

5.6 Apply the model and plot the graphs for accuracy and loss:

This module involves applying the selected model to predict the hand gestures and plotting the graphs for accuracy and loss during training. We will compile the model and apply it using fit function. The batch size will be 1. Then we will plot the graphs for accuracy and loss. We got average validation accuracy of 99.00% and average training accuracy of 99.00%.

5.7 Accuracy on test set:

After training and evaluating the model on the validation set, the accuracy of the model will be assessed on the test set. The accuracy on the test set will be an important metric for evaluating the model's performance. We got an accuracy of 99.00% on test set.

5.8 Saving the Trained Model:

This module involves saving the trained CNN model for future use, such as deploying the model as a web application using Flask framework so that can be accessed by any users to predict the likelihood of hand gesture using image.

Once you're confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or .pkl file using a library like pickle . Make sure you have pickle installed in your environment.

Next, let's import the module and dump the model into .h5 file

5.9 Real-time Hand Gesture Recognition using TensorFlow & OpenCV

Gesture recognition is an active research field in Human-Computer Interaction technology. It has many applications in virtual environment control and sign language translation, robot control, or music creation. In this machine learning project on Hand Gesture Recognition, we are going to make a real-time Hand Gesture Recognizer using the MediaPipe framework and Tensorflow in OpenCV and Python.

OpenCV is a real-time Computer vision and image-processing framework built on C/C++. But we'll use it on python via the OpenCV-python package.

MediaPipe is a customizable machine learning solutions framework developed by Google. It is an open-source and cross-platform framework, and it is very lightweight. MediaPipe comes with some pre-trained ML solutions such as face detection, pose estimation, hand recognition, object detection, etc.

TensorFlow is an open-source library for machine learning and deep learning developed by the Google brains team. It can be used across a range of tasks but has a particular focus on deep neural networks.

Neural Networks are also known as artificial neural networks. It is a subset of machine learning and the heart of deep learning algorithms. The concept of Neural networks is inspired by the human brain. It mimics the way that biological neurons send signals to one another. Neural networks are composed of node layers, containing an input layer, one or more hidden layers, and an output layer.

We’ll first use MediaPipe to recognize the hand and the hand key points. MediaPipe returns a total of 21 key points for each detected hand.

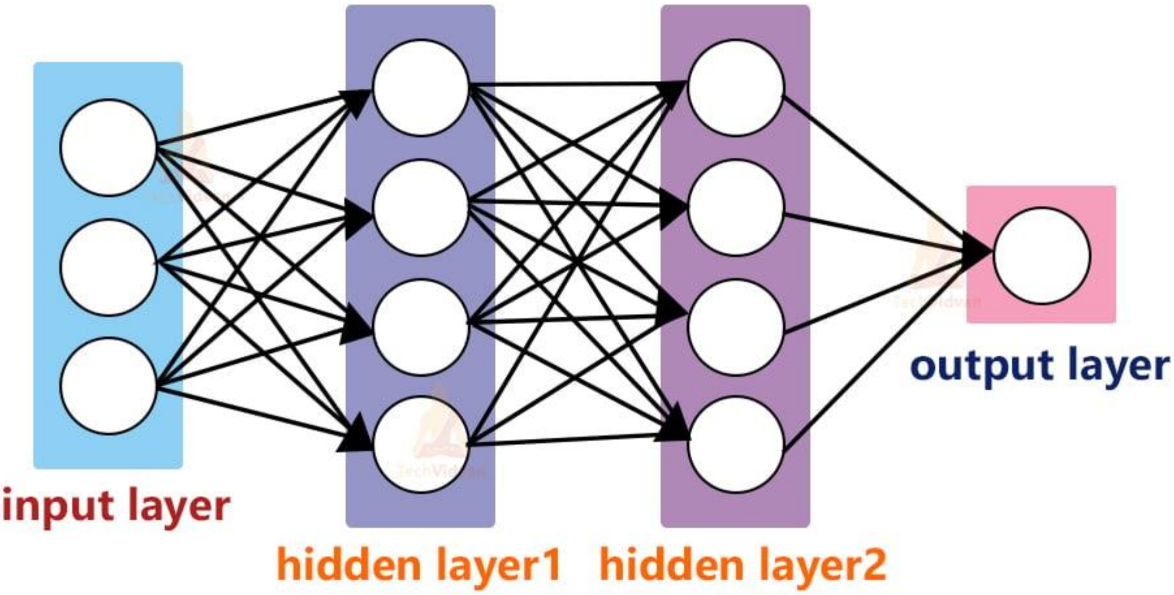


Fig 5.5 MediaPipe Architecture

We’ll first use MediaPipe to recognize the hand and the hand key points. MediaPipe returns a total of 21 key points for each detected hand.

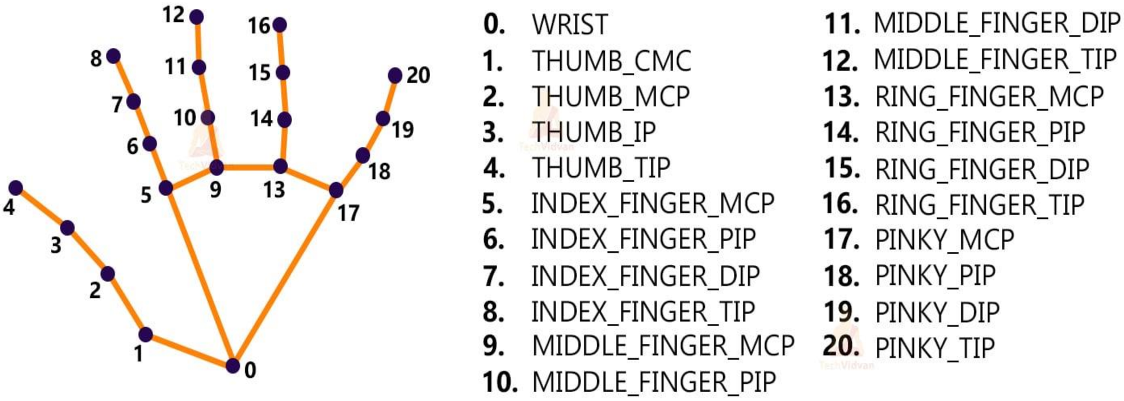


Fig 5.6 Node placements of MediaPipe

These key points will be fed into a pre-trained gesture recognizer network to recognize the hand pose.

5.10 Steps to solve the project

Step 1 – Import necessary packages:

To build this Hand Gesture Recognition project, we’ll need four packages. So first import these.

Step 2 – Initialize models:

Mp.solution.hands module performs the hand recognition algorithm. So we create the object and store it in mpHands.

Using mpHands.Hands method we configured the model. The first argument is max_num_hands, that means the maximum number of hand will be detected by the model in a single frame. MediaPipe can detect multiple hands in a single frame, but we’ll detect only one hand at a time in this project.

Mp.solutions.drawing_utils will draw the detected key points for us so that we don’t have to draw them manually.

Initialize Tensorflow:

Using the load_model function we load the TensorFlow pre-trained model. Gesture.names file contains the name of the gesture classes. So first we **open** the file

using python’s inbuilt open function and then read the file. After that, we read the file using the read() function.

Output :

[‘okay’, ‘peace’, ‘thumbs up’, ’no hand detection’, ‘open hand’]

The model can recognize 10 different gestures.

Step 3 – Read frames from a webcam:

We create a VideoCapture object and pass an argument ‘0’. It is the camera ID of the system. In this case, we have 1 webcam connected with the system. If you have multiple webcams then change the argument according to your camera ID. Otherwise, leave it default.

The cap.read() function reads each frame from the webcam.

- cv2.flip() function flips the frame.
- cv2.imshow() shows frame on a new openCV window.
- The cv2.waitKey() function keeps the window open until the key ‘q’ is pressed.

Step 4 – Detect hand key points:

MediaPipe works with RGB images but OpenCV reads images in BGR format. So, using cv2.cvtCOLOR() function we convert the frame to RGB format. The process function takes an RGB frame and returns a result class. Then we check if any hand is detected or not, using result.multi_hand_landmarks method. After that, we loop through each detection and store the coordinate on a list called landmarks. Here image height (y) and image width(x) are multiplied with the result because the model returns a normalized result. This means each value in the result is between 0 and 1. And finally using mpDraw.draw_landmarks() function we draw all the landmarks in the frame.

Step 5 – Recognize hand gestures:

- The model.predict() function takes a list of landmarks and returns an array contains 10 prediction classes for each landmark.
- The output looks like this- [[2.0691623e-18 1.9585415e-27 9.9990010e-01 9.7559416e-05 1.6617223e-06 1.0814080e-18 1.1070732e-27 4.4744065e-16 6.6466129e-07 4.9615162e-21]]
- Np.argmax() returns the index of the maximum value in the list.
- After getting the index we can simply take the class name from the classNames list.
- Then using the cv2.putText function we show the detected gesture into the frame.

5.11 Implementation Support

There are many features in Python, some of which are discussed below as follows:

1. Easy to code: Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, JavaScript, Java, etc. It is very easyto code in python language and anybody can learn python basics in a few hours or days. It isalso a developer-friendly language.

2. Free and Open Source: Python language is freely available at the official website and you can download it from the given download link below click on the Download Python keyword. Download Python Since it is open- source, this means that source code is also available to the public. So, you can download it as, use it as well as share it.

3. High-Level Language: Python is a high-level language. When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.

4. Extensible feature: Python is an Extensible language. We can write some Python code into C or C++ language and also, we can compile that code in C/C++ language.

5. Python is Portable language: Python language is also a portable language. For example,if we have python code for windows and if we want to run this code on other platforms suchas Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.

Basic syntax of Python

Python syntax can be executed by writing directly in the Command Line:

```
>>> print(“Hello, World!”)Hello, World!
>>> print(3+3)6
```

Or ,by creating a python file on the server, using the .py file extension, and running it in the Command Line:

```
C:\Users\Your Name>python myfile.py C:\Desktop>python aibasedui.py
```

Python Indentation

Indentation refers to the spaces at the beginning of a code line where in other programming languages the indentation in code is for readability only, the indentation in Python is very important. Python uses indentation to indicate a block of code.

Example:

```
if 5 > 2:
print(“Five is greater than two!”)
```

Python will give you an error if you skip the indentation.Output:

```
Five is greater than two!
```

Python Variables

In Python, variables are created when you assign a value to it.Example:

```
x = 5
y = “Hello, World!”
```

Python has no command for declaring a variable.

Chapter 6

TESTING

Testing can be stated as the process of verifying and validating whether a software or application is bug-free, meets the technical requirements as guided by its design and development, and meets the user requirements effectively and efficiently by handling all the exceptional and boundary cases. The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy, and usability. It mainly aims at measuring the specification, functionality, and performance of a software program or application, so that it may help in fixing the bugs.

Test Case is a set of actions executed to verify a particular feature or functionality of your software application. A Test Case contains test steps, test data, precondition, postcondition developed for specific test scenario to verify any requirement. The test case includes specific variables or conditions, using which a testing engineer can compare expected and actual results to determine whether a software product is functioning as per the requirements.

6.1 Unit Testing

Unit Testing is a software testing technique by means of which individual units of software i.e. group of computer program modules, usage procedures, and operating procedures are tested to determine whether they are suitable for use or not. Unit Testing is defined as a type of software testing where individual components of a software are tested. Unit Testing of the software product is carried out during the development of an application so that the errors can be fixed easily.

Table 6.1 Unit Test cases for whole project

Test Case No.	Description	Actual Output	Expected Output	Pass/Fail
1	Ensure that the OpenCV model is correctly capturing the environment	The environment correctly displayed	The environment correctly displayed	Pass
2	Ensure that MediaPipe is drawing correct pattern on the hand	Correct pattern drawn	Correct pattern drawn	Pass

3	Ensure output is “No Hand Detected” if no hand shown in camera	No Hand detected is seen as output	No Hand detected is seen as output	Pass
4	Ensure that the website is allowing login for the correct user	Login Success	Login Fail	Pass
5	Test if the image is able to be uploaded to the website and taken as input by system	Image accepted	Image accepted	Pass
6	Ensure KeyPoint Classifier is correctly classifying	Correctly predicted the gesture	Correctly predicted the gesture	Pass

6.2 System Testing

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested. System Testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both. System testing tests the design and behavior of the system and also the expectations. System testing evaluates the compliance of the complete integrated system with its corresponding requirements, ensuring that it meets specified criteria. It builds upon integration testing by assessing the interactions between integrated units and detecting irregularities. The observed behavior during system testing provides insights into the system's design, behavior, and alignment with both system and functional requirements.

And of the customer. It is performed to test the system beyond the bounds mentioned in the software requirements specification (SRS).

System Testing is basically performed by a testing team that is independent of the development team that helps to test the quality of the system impartial. It has both functional and non-functional testing.

System testing for a project involving virtual hand gestured keyboard, mouse, voice-based UI, and hand gesture-based UI would involve testing the overall functionality of the entire system, including its integration with other components. Here are the system testing scenarios for each of the components:

1. Correctly predict the gesture done by the user on camera: The user will do a particular gesture on the camera. The system successfully classified the gesture accordingly- No Hand Detected, Open Hand, Thumbs Up, OK, Peace.
2. Correctly predict the gesture displayed in the image: This system was able to correctly process the image and using the Xception Model, it classified it correctly.
3. Connect to the Web using Flask and allow user Login: This system made use of Flask to connect the website. The user was able to login by giving correct username and password to access the features of website.

6.3Validation Testing

Validation is an essential process in software development that involves checking whether the software product is up to the mark and meets the high-level requirements. It is a crucial step in ensuring that the developed software is the right product and aligns with the user's needs. The process of validation involves checking the actual product against the expected product, ensuring that it meets the intended functionality and specifications. The primary goal of validation is to determine whether the developed software satisfies the intended use and the customer's requirements.

Validation is a dynamic testing process that checks the behavior of the software in a dynamic environment. It involves executing the software under test with various inputs and scenarios and observing the system's behavior and responses. It is an iterative processthat involves multiple rounds of testing to ensure that the software satisfies the requirements and meets the user's expectations. Validation testing aims to identify defectsin the software product and fix them before the product is released to the market. By ensuring that the software meets the intended use, validation testing helps in reducing the risk of delivering a defective product and enhances the quality of the software.

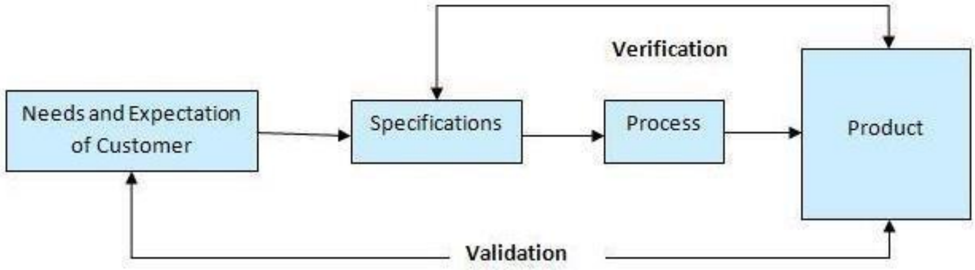


Fig 6.1 Software verification and validation

Validation testing is the process of evaluating the performance of a system to ensure that it meets the desired requirements and specifications. In the case Sign Language Recognition System, validation testing can be carried out to ensure that the system meets the following requirements:

1. Accuracy: The level of correctness with which the system was able to classify the hand gestures to the respective categories.
2. Responsiveness: Ensure there is no lag in the capturing of video from the camera. Also, once the user uploads the image, the result is displayed almost instantaneously
3. Robustness: If the user inputs an image not in the database, the system should classify the image correctly with high accuracy.
4. Usability: The system should be user-friendly and intuitive, with clear instructions and feedback, to ensure that users can easily learn how to use it and perform the desired actions.

To carry out validation testing, the system can be tested using a variety of different hand gestures and voice commands, with different users and in different environments, to ensure that it performs accurately, responsively, and robustly under a range of conditions. The system be tested using different types of input devices, such as a webcam and website.

Chapter 7

RESULTS

This section discusses the results from each module implemented.

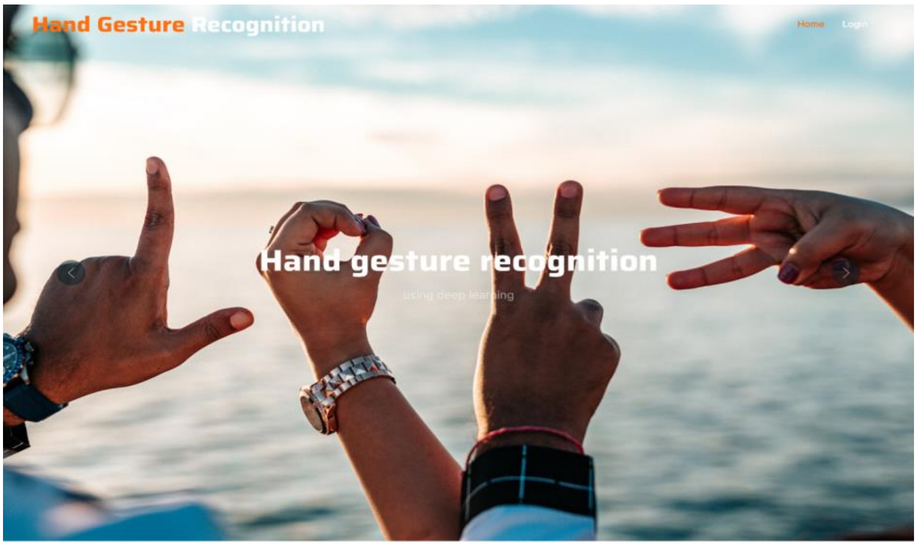
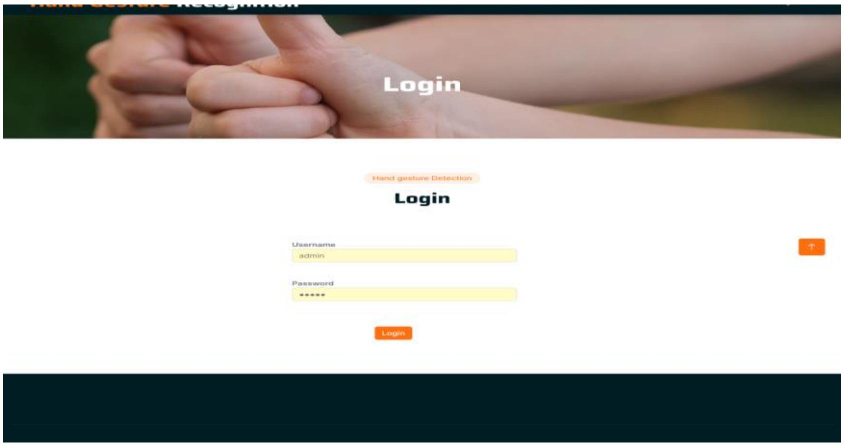


Fig 7.1 Website page

Fig 7.1 represents the first page the user will see once he visits our website. This is created such that the user will quickly understand the tools available for him/her to use.



Username
admin

Password
*****

Fig 7.2 Login Page

Fig 7.2 represents the login page in our website. The user can login into his/her account in order to use our website for sign language detection. Once the user enters the login username or password

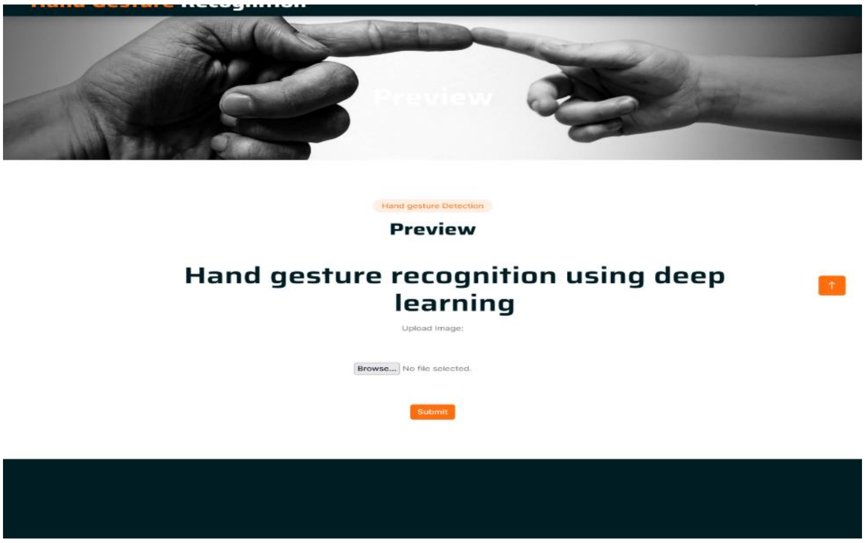

Preview
Hand gesture Detection
Preview
Hand gesture recognition using deep learning
Upload Image:
Browse... No file selected.
Submit

Fig 7.3 UI to enable user to upload image

Fig 7.3 represents the webpage the user sees which allows him to upload an image. Click on the Browse button to select an image from system and then Submit in order to get the prediction.

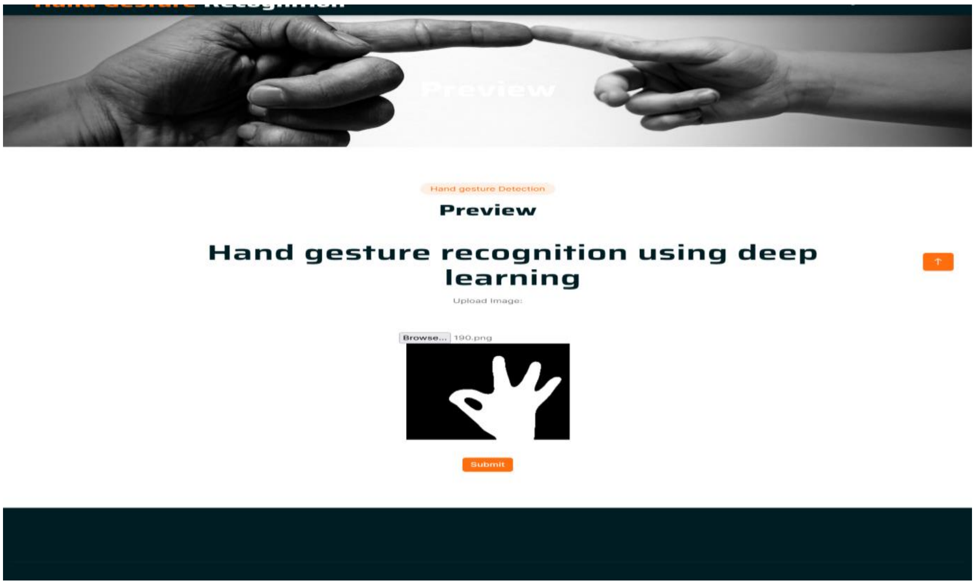
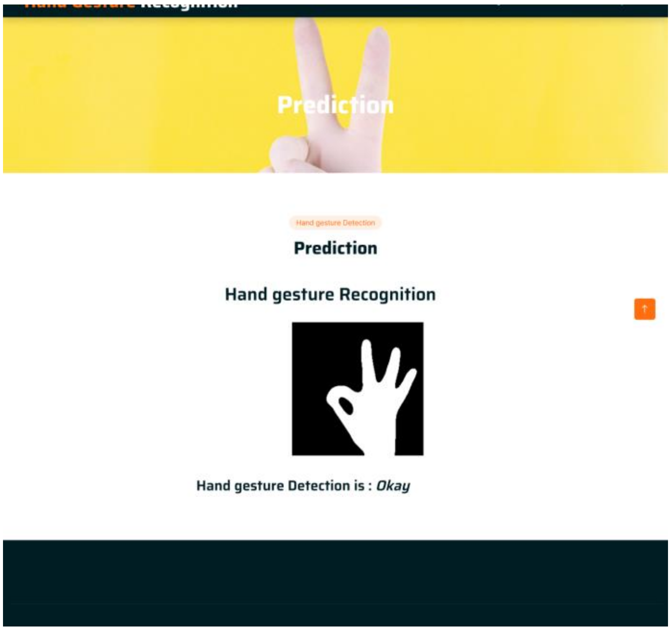


Fig 7.4 Upload Image

Fig 7.4 represents the webpage the user sees once he uploads the image. Here the user has uploaded image showing OK



Hand gesture Detection

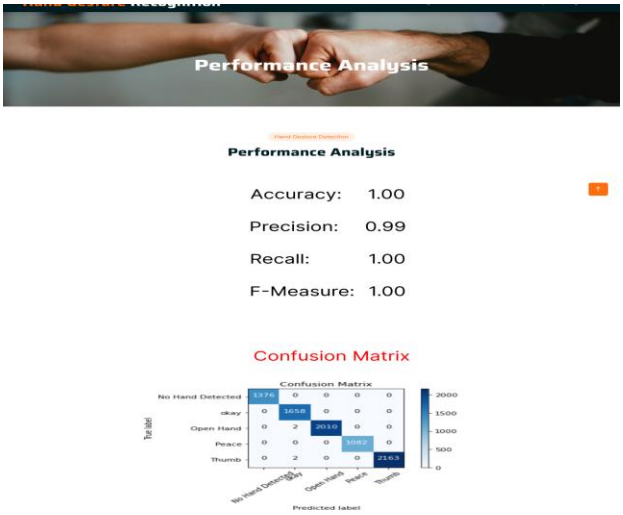
Prediction

Hand gesture Recognition

Hand gesture Detection is : *Okay*

Fig 7.5 After submitting image

Fig 7.5 shows the webpage after the user has submitted the image. Here we can see that the system has been able to predict that sign represented in the Image is OK



Hand Gesture Detection

Performance Analysis

Accuracy:	1.00
Precision:	0.99
Recall:	1.00
F-Measure:	1.00

Confusion Matrix

	No Hand Detected	okay	Open Hand	Peace	Thumb
No Hand Detected	1376	0	0	0	0
okay	0	1658	0	0	0
Open Hand	0	2	2010	0	0
Peace	0	0	0	1082	0
Thumb	0	2	0	0	2163

Fig 7.6 VR Performance Analysis

This figure represents the performance of the system. It shows the performance of all time, that is the figure is calculated based on all images uploaded till date,

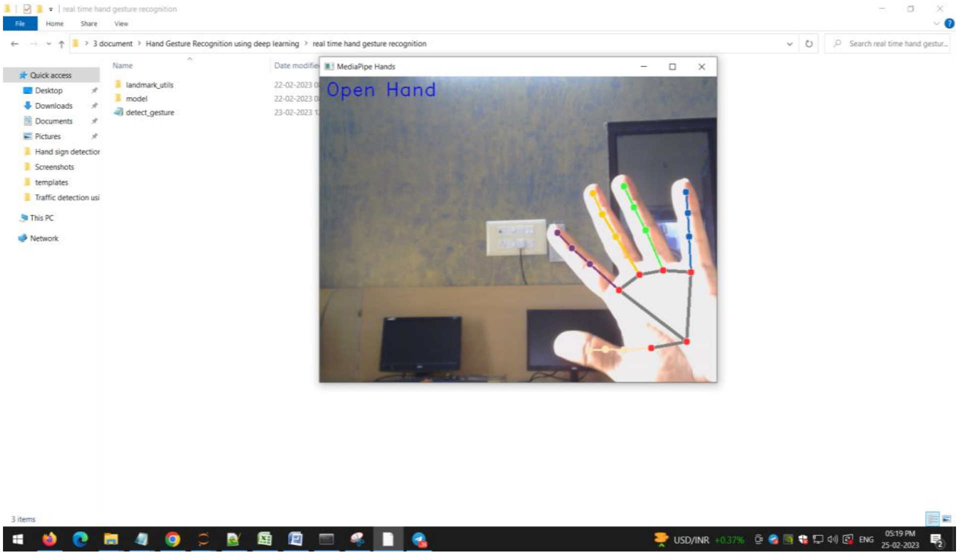


Fig 7.7 VR Rel time sign language detection

Fig 7.7 shows real time hand detection. Here the user has placed an Open hand in front of the camera. The system recognizes the sign shows and displays the gesture displayed via text as seen in the figure

7.1 Summary

The project utilizes cutting-edge technologies such as MediaPipe and deep learning with OpenCV to recognize sign language. There are 2 models in the project. Firstly, the Xception model recognizes the sign gesture represented in an image. This system has a website to ensure an easy-to-use interface for the user. Secondly, the real time sign language detection system makes use of OpenCV and MediaPipe to capture a pattern which represents the hand gesture and uses the KeyPoint Classifier to classify it to the appropriate category. Thus, this project gives the user 2 methods to communicate in sign language.

Chapter 8

CONCLUSION AND FUTURE WORK

8.1 Conclusion

In summary, the project "Real-Time Hand Gesture Detection for Sign Language Recognition using Python" offers an effective approach to facilitate communication for individuals with hearing impairments. By utilizing computer vision and machine learning algorithms, this system detects and recognizes hand gestures in real time, potentially translating them into text. The implementation of the Xception architecture has shown high levels of accuracy in both training and validation phases, underscoring the system's effectiveness. This technology holds promise for enhancing inclusivity and communication in environments such as classrooms, workplaces, and public areas. It provides a user-friendly interface that supports seamless interaction between sign language users and those unfamiliar with sign language. Moreover, the automated gesture recognition facilitated by this system could lead to quicker and more effective communication for sign language users.

However, it's important to acknowledge the challenges associated with sign language recognition, including regional variations in sign language and the need for standardization. The current system also faces limitations like a restricted gesture vocabulary, sensitivity to variations in gesture execution, and ongoing needs for optimization. Despite these challenges, the project represents a significant step forward in sign language recognition technology through real-time hand gesture detection. With further research and development, the system's accuracy, robustness, and overall usability could be greatly enhanced. Ultimately, the project "Real-Time Hand Gesture Detection for Sign Language Recognition using Python" stands as a promising tool for improving communication and fostering inclusivity .

8.2 Future Work

Several potential improvements could be considered for enhancing the "Real-Time Hand Gesture Detection for Sign Language Recognition using Python" project. These improvements are aimed at boosting the system's performance, user experience, and accessibility:

- **Enlarging Gesture Vocabulary:** Currently, the system recognizes a limited number
- of hand gestures. A valuable improvement would be to expand this vocabulary to

- encompass a wider array of sign language gestures, including those specific to different regions and more intricate gestures. Such an expansion would enhance the system's versatility and better meet the varied needs of sign language users.
- **Increasing Robustness to Environmental Variations:** To improve gesture recognition accuracy affected by individual differences and environmental factors like lighting and camera angles, enhancements could include data augmentation, feature normalization, and advanced machine learning techniques.
- **Incorporating Real-Time Translation:** The current system, which detects and recognizes gestures effectively, lacks a translation feature. Introducing real-time conversion of gestures into text or speech would enhance communication between sign language users and non-users.
- **Enhancing User Interface and Accessibility:** Improvements to the user interface and accessibility could make the system more user-friendly and accessible. This might involve designing an intuitive interface with easy navigation, providing user feedback, and optimizing the application for various devices, including smartphones and wearables. Implementing features like adjustable text size, color contrast options, and support for multiple sign language dialects could also be beneficial.
- **Real-World Testing and Feedback:** To assess the system's practicality, accuracy, and effectiveness, further testing and validation in real-world scenarios such as classrooms, offices, and public places are essential. Gathering feedback from users, particularly those with hearing disabilities, would help pinpoint necessary refinements and enhance system functionality.
- **Deployment on Edge Devices:** Implementing the system on edge devices like embedded systems or IoT devices could improve processing efficiency, reduce reliance on cloud services, and enhance both responsiveness and data privacy. Future developments should focus on optimizing the system's performance on these devices, considering constraints related to computational resources, energy consumption, and bandwidth.

REFERENCES

[1] Rafiqul Zaman Khan ,Noor Adnan Ibraheem. Hand Gesture Recognition: A Literature Review, 2012.

[2] Viraj Shinde, Tushar Bacchav, Jitendra Pawar, Mangesh Sanap.Hand Gesture Recognition System Using Camera , 2014.

[3] Siddharth S. Rautaraya , Anupam Agrawala. Real Time Gesture Recognition System with Dynamic Environment, 2012.

[4] Rina Damdoo, Kanak Kalyani , Jignyasa Sanghavi. Adaptive Hand Gesture Recognition System Using Machine Learning, 2020.

[5] Sumit Yadav, Chandradeep, Vishal Ku. Singh. Voice Recognition Techniques: A Review Paper, 2020.

[6] Ram Sethuraman , J.Selvin Paul Peter , Shanthan Reddy Middela. An Analysis of Automatic VR and Speaker Identification Algorithms and its Applications, 2018.

[7] David J. White, Andrew P. King, Shan D. Duncan. Voice recognition technology as a tool for behavioral research, 2002.

[8] Sang M. Lee , Yong K.Cho , David L.Oslon. VR: An examination of an evolving technology and its use in organizations, 1986.

[9] M. Naveenkumar , A. Vadivel. OpenCV for Computer Vision Applications Author, 2015.

[10] P Y Kumbhar, Mohammad Attaullah, Shubham Dhere, Shivkumar Hipparagi. Real Time Face Detection and Tracking Using OpenCV, 2017.

[11] Akhilesh Shukla, Dr. Devesh Katiyar, Mr. Gaurav Goel. Gesture Recognition-based AI Virtual Mouse, 2022.

[12] Mishaha MK, Manjusha MS. Virtual Mouse And Keyboard For Computer Interaction, 2022.

[13] Dipankar Gupta, Emam Hossain, Mohammed Sazzad Hossain, Mohammad Shahadat Hossain, and Karl Andersson. An Interactive Computer System with Gesture-Based Mouse and Keyboard, 2019.

[14] Dr. Kavitha. , Nachammai. , Ranjani. , Shifali. Speech Based Voice Recognition System for Natural Language Processing, 2019.

[15] Khin Myat Nwe Win, Zar Hnin , Yin Myo Kay Khing Thaw. Review and Perspectives of NLP for Speech Recognition, 2020.

[16] M.A.Anasuya , S.K.Katti. Speech Recognition by Machine: A Review, 2009.

A Survey on Sign Language Recognition: A Case Study

[1]Harshitha P, [2]Ruchi, [3]Sameer Kumar Singh, [4]Saurav Kumar, [5]Tejas V Kangod

[1]Guide, [2,3,4,5]Students
RNS Institute of Technology

Abstract- **The project focuses on creating a new language that combines the power of computer vision. Technologies such as image processing and machine learning**
were used to develop the systemIt allows extracting key points from the user's hands and body language. The system, imaging and designed to use technologies such as machine learning. It allows extracting important details from the user's hand and body language.

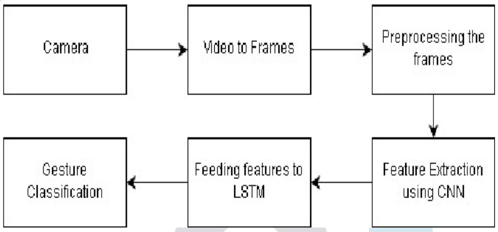


Figure 1 OpenCV process for recognition

Our project uses OpenCV to process video files and capture hand movements to provide instant feedback. This model is designed to learn and recognize alphabetic signs. We created a user-friendly graphical interface using the Tkinter library and made it accessible to many users. Users can engage .

Introduction:
Language is a language in which people
express meaning through language patterns through visual communication. It also includes facial and body language, which are called non-book signs and play an important role in understanding the truth of the signs. People who speak badly have difficulty communicating with others. Because even if family members learn the language, most people in the world cannot learn it.

This is because they do not need the language in their daily work. People from many countries signed it. It includes algorithms inspired by the brain's patterns and functions, including decision-making, called neural networks. Language checkbooks help people with special abilities. Therefore, they are not alone or isolated. Science couldn't have been developed so quickly if the ideas of people like Stephen Hawking had not been heard.
is about algorithms that respect decisions resulting from the structure and performance of the brain, called artificial neural networks. Sign language app will help special people communicate with all. Therefore they will not be separated or excluded.

1.1 Objectives:

The main purpose of creating this application is to bring people closer to each other. Language detection will be performed in real time to ensure rapid response and provide instant feedback for communication and learning. There should be no commerce in the exchange of information. It is important to have a user-friendly graphical interface (GUI) that is intuitive and accessible to users with different levels of expertise as well as users without expertise. Applications should be developed assuming end users have no technical skills. With this in mind, an application form should be created that allows users to practice and experience the application. The application should be able to convert sign language into different languages. Advanced systems that can accurately create 3D images and provide accurate outputs should be used

2. Related Work:

There has been a lot of research and development to create the best word analysis application. Tanuj Bohrab Contact should be established immediately. The system has hand detection, skin color segmentation, average blur and visual

detection features. These are used for images in the dataset to get better results. It was trained to use large dataset of 40 groups and can predict 17,600 image parameters with 99% accuracy in 14 seconds. Dr. Muthu Mariappan H. and Gomathi V. developed a fast annotation detection system using face, left hand and right hand detection and fuzzy C language algorithm to classify data books listed into categories. The system can reach a 75% accuracy rate. Suharjito implemented language perception with a 3D model for the first time using adaptive learning.
10-word LSA64 public dataset containing 500 videos. The data layer for training is split in the ratio 6:2:2.Public dataset LSA64 was used for 10 vocabularies with 500 videos. For training, the dataset is distributed in 6:2:2 ratio.

Figure 2 Real time detection

For checking the efficiency of the system 320+ videos is used for training, 110+ for validation and 110+ for testing set. It has very low-validation accuracy. Aditya Das trained a CNN using Inception v3 model.

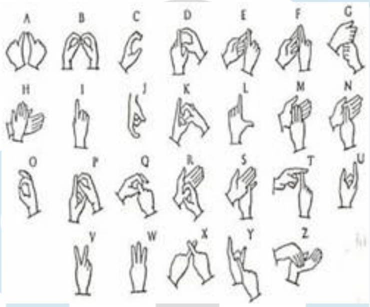


Figure 3 American Sign Lamguage

Before training, data augmentation is applied on the images to avoid overfitting. This model gives more than 90% accuracy on a dataset consisting of 24 class labels where each class has 100 images. The paper- Developed LSTM model for recognition of extended symbols using leap motion. A new framework for extended SLR using leap motion sensors is described. An improved LSTM architecture is also proposed the recognition of sign words and sentences. Average accuracy of 72.3% have been recorded on the signed sentences. For isolated sign-language words an accuracy of 89.5% was recorded. By increasing the training data, the performance increases.

3.1 Challenges in sign-language detection model

1. A single sign language is not following throughout the world. American Sign Language (ASL) , Indian Sign Language (ISL) are many other sign languages. So it is difficult guess the meaning of a particular sign
2. Achieving real-time processing in sign language detection is a formidable challenge. The system will have to analyse and precisely tell the meaning of sign and give output within a few seconds.
3. It is difficult to manually create the training dataset for signature recognition. There will be inconsistency in the existing training datasets.
4. The model will have to ignore the background objects which can include lighting, environment and people. The system will have to filter these out and give accurate output.

3.2 Dataset Generation

It is important for development of consistent Motion Gesture Library so that images captured by the system during communication can be interpreted correctly. Firstly, capture around 1000 images of each of the symbol for training purposes and around 400 images per symbol for testing purpose. After that, capture each frame shown by the webcam of the system. In each frame a region-of-interest (ROI) is defined which is denoted by a blue bounded square. From the whole image extract our ROI which is RGB and convert it into grey scale Image.

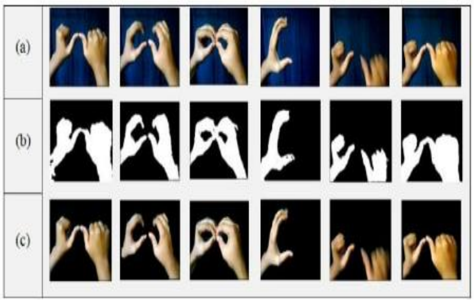


Figure 4 Sign Language Dataset

Finally, apply the gaussian blur filter to the dataset which helps to extract various features of the selected image.

3.3 Advantages

1. Basic Communication: The system allows for basic communication for people that don't know ASL to make it easier for visitors. Personalization: Custom signs can be tailored to an individual's unique communication needs, preferences, and even regional variations, enhancing the user experience.
2. Real-Time Recognition: The real-time recognition of custom signs from moving frames input provides a natural and efficient means of communication, enabling faster and more fluid interactions.
3. Compact and Accessible: Accessible through mobile and web applications, the system offers a convenient and portable means of communication, allowing users to communicate from various devices.
4. Simplified Model Training: Training models on a limited custom gesture dataset is simpler and requires less data compared to large, generic sign language datasets, reducing training time and computing resources.
5. Focused Vocabulary: The system's focused vocabulary enhances recognition accuracy, as it is specifically optimized for practical and commonly used words and phrases..

3.4 Disadvantages

1) Limited to Simple Phrases: The system's primary limitation is that it's designed for simple phrases and may not support the complexities of a full sign language, limiting its application scope.
2) Custom Gesture Collection: The process of collecting personalized custom gesture data can be time-consuming and may require user involvement, potentially posing a challenge.
3) Restricted Vocabulary: The system's vocabulary, while tailored to user needs, may not cover all communication requirements, potentially leading to gaps in expression.
4) Confusion Risk: With a small custom gesture vocabulary, there's a higher risk of confusion, especially if gestures are similar for different words or phrases.
5) Not Accessible to Non-Signers: This system primarily benefits some knowledgeable sign language, making it less accessible to individuals who only know spoken languages.

Conclusion:

This article describes the work done to date in research field of language analysis applications. Certain objectives and criteria have to be met for the application to be efficient. Explaining the functioning of existing systems and laying the foundation for the development of new systems. All the features of the sign-language detection system are described which would be implemented in the new system. The project's significance lies in its potential to foster inclusivity, understanding, and accessibility, creating a universe of effective communication.

REFERENCES:

1. Fig 3.1 by T. Bohra, S. Sompura, K. Parekh and P. Raut, "Real-Time Two Way Communication System for Speech and Hearing Impaired Using Computer Vision and Deep Learning," 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2019, pp. 734-739,doi:10.1109/ICSSIT46314.2019.8987908.
2. Fig 2 by H. Muthu Mariappan and V. Gomathi, "Real-Time Recognition of Indian Sign Language," 2019 in Data Science (ICCIDS), Chennai, India,2019,pp.1-6,doi: 10.1109/ICCIDS.2019.8862125.
3. Suharjito, Suharjito & Gunawan, Herman & Thiracitta, Narada & Nugroho, Ariadi. (2018). Sign Language Recognition Using Modified CNN Model. 1-5. 10.1109/INAPR.2018.8627014.
4. Fig 3 by Singha, Joyeeta & Das, Karen. (2013), "Recognition of Indian Sign Language in Live Video," International Journal of Computer Applications. 70. 10.5120/12174-7306.

5. Koller, Oscar & Zargaran, Sepehr & Ney, Hermann & Bowden,Richard. (2016). Deep Sign: Hybrid CNN-HMM for CSL Recognition. 10.5244/C.30.136
6. A. Mittal, P.Kumar, P.R Roy, R Balasubramanian, B.B. Chaudhuri, "A Modified-LSTM Model for Continuous Sign Language Recognition using Leap motion",2019. https://ieeexplore.ieee.org/abstract/document/8684245
7. Fig 2 by Elizabth Cody, "Deaftravel", http://www.deaftravel.co.uk/signs.php?id=27

JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

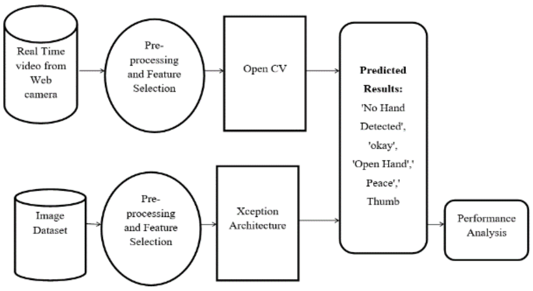
Sign Language Recognition: A Case Study

Harshitha, Ruchi, Sameer Kumar Singh, Saurav Kumar, Tejas V Kangod

ASSISTANT PROFESSOR, STUDENT
RNS INSTITUTE OF TECHNOLOGY

Abstract :

The "Real-Time Hand Gesture Detection for Sign Language Recognition using Python" project aims to develop a system for recognizing sign language gestures in real-time. It utilizes computer vision techniques and machine learning algorithms to detect and classify hand movements, achieving a training accuracy of 99.34% and validation accuracy of 99.00% with the Xception architecture model. The system, implemented in Python with OpenCV, detects gestures like "Ok," "Open Hand," "Peace," "Thumb," and "No Hand Detected." With a user-friendly interface, it facilitates non-sign language users, enhancing inclusivity in classrooms, workplaces, and public spaces.


Real Time video from Web camera
Pre-processing and Feature Selection
Open CV
Predicted Results: 'No Hand Detected', 'okay', 'Open Hand',' Peace',' Thumb
Image Dataset
Pre-processing and Feature Selection
Xception Architecture
Performance Analysis

Introduction:

Language is a language in which people express meaning through language patterns through visual communication. It also includes facial and body language, which are called non-book signs and play an important role in understanding the truth of the signs. People who speak badly have difficulty communicating with others. Because even if family members learn the language, most people in the world cannot learn it.

This is because they do not need the language in their daily work. People from many countries signed it. It includes algorithms inspired by the brain's patterns and functions, including decision-making, called neural networks. Language checkbooks help people with special abilities. Therefore, they are not alone or isolated. Science couldn't have been developed so quickly if the ideas of people like Stephen Hawking had not been heard. is about algorithms that respect decisions resulting from the structure and performance of the brain, called artificial neural networks. Sign language app will help special people communicate with all. Therefore, they will not be separated or excluded.

1.1 Objectives:

The main purpose of creating this application is to bring people closer to each other. Language detection will be performed in real time to ensure rapid response and provide instant feedback for communication and learning. There should be no commerce in the exchange of information. It is important to have a user-friendly graphical interface (GUI) that is intuitive and accessible to users with different levels of expertise as well as users without expertise. Applications should be developed assuming end users have no technical skills. With this in mind, an application form should be created that allows users to practice and experience the application. The application should be able to convert sign language into different languages. Advanced systems that can accurately create 3D images and provide accurate outputs should be used.

2. Related Work:

There has been a lot of research and development to create the best word analysis application. Tanuj Bohrab Contact should be established immediately. The system has hand detection, skin color segmentation, average blur and visual detection features. These are used for images in the dataset to get better results. It was trained to use large dataset of 40 groups and can predict 17,600 image parameters with 99% accuracy in 14 seconds. Dr. Muthu Mariappan H. and Gomathi V. developed a fast annotation detection system using face, left hand and right hand detection and fuzzy C language algorithm to classify data books listed into categories. The system can reach a 75% accuracy rate. Suharjito implemented language perception with a 3D model for the first time using adaptive learning.

10-word LSA64 public dataset containing 500 videos. The data layer for training is split in the ratio 6:2:2.Public dataset LSA64 was used for 10 vocabularies with 500 videos. For training, the dataset is distributed in 6:2:2 ratio.

Figure 2 Real time detection

For checking the efficiency of the system 320+ videos is used for training, 110+ for

validation and 110+ for testing set. It has very low-validation accuracy. Aditya Das traineda CNN using Inception v3 model.

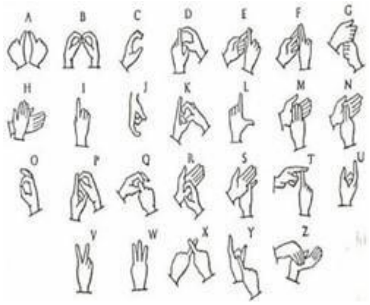


Figure 3 American Sign Lamguage

Before training, data augmentation is applied on the images to avoid overfitting. This model gives more than 90% accuracy on a dataset consisting of 24 class labels where each class has 100 images. The paper- Developed LSTM model for recognition of extended symbols using leap motion. A new framework for extended SLR using leap motion sensors is described. An improved LSTM architecture is also proposed the recognition of sign words and sentences. Average accuracy of 72.3% havebeen recorded on the signed sentences. For isolated sign-language words an accuracy of 89.5% was recorded. By increasing the training data, the performance increases.

3.1 Challenges in sign-language detection model

1. A single sign language is not following throughout the ASL, Indian Sign Language(ISL) are many other sign languages. It's difficult to guess the meaning.
2. Achieving real-time processing detection is a formidable challenge. The system will have to analyse and precisely tell the meaning ofsign and give output within a few seconds.
3. It is difficult to manually create the training dataset for signature. Recognition. There will be inconsistency in the existing training datasets.
4. The model will have to ignore the background objects which can include lighting, environment and people. The system will have to filter these out and give accurate output.

3.2 Dataset Generation

It’s imperative for development of consistent Dataset:

Kaggle Dataset Link:

https://www.kaggle.com/datasets/jayaprakashpondy/hand-gesture-dataset

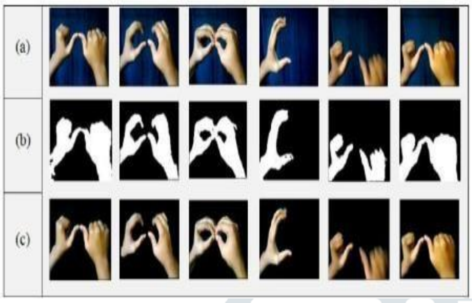


Figure 4 Sign Language Dataset

Finally, apply the gaussian blur filter to the dataset which helps to extract various features of the selected image.

3.3 Advantages

1. Basic Communication: The system allows for basic communication for people that don’t know ASL to make iteasier for visitors. Personalization: Custom signs can be tailored to an individual's unique communication needs, preferences, and even regional variations, enhancing the user experience.
2. Real-Time Recognition: The real-time recognition of custom signs from moving frames input provides a natural and efficient means of communication, enabling faster and more fluid interactions.
3. Compact and Accessible: Accessible through mobile and web applications, the system offers a convenient and portable means of communication, allowing users to communicate fromvarious devices.
4. Simplified Model Training: Training models on a limited custom gesture dataset is simpler and requires less datacompared to large, generic sign language datasets, reducing training time and computing resources.
5. Focused Vocabulary: The system's focused vocabulary enhances recognition accuracy, as it is specifically optimized for practical and commonly used words and phrases.

3.4 Disadvantages

1) Limited to Simple Phrases: Thesystem's primary limitation is that it's designed for simple phrases and may not support the complexities of a full sign language, limiting its application scope.
2) Custom Gesture Collection: The process of collecting personalized custom gesture data can be time- consuming and may require user involvement, potentially posing achallenge.
3) Restricted Vocabulary: The system's vocabulary, while tailored to user needs, may not cover all communication requirements,potentially leading to gaps inexpression.
4) Confusion Risk: With a small custom gesture vocabulary, there's a higher riskof confusion, especially if gestures are similar for different words or phrases.
5) Not Accessible to Non-Signers: This system primarily benefits some knowledgeable sign language, making it less accessible to individuals who only know spoken languages.

4 Methodology

4.1 Importing the necessary libraries:

This module involves importing the necessary libraries such as NumPy, TensorFlow to implement the model. We will be using Python language for this. First, we will import the necessary libraries such as keras for building the main model, sklearn for splitting the training and test data, PIL for converting the images into array of numbers and other

4.2 Importing the necessary libraries:

This module involves importing the necessary libraries such as NumPy, TensorFlow to implement the model. We will be using Python language for this. First we will import the necessary libraries such as keras for building the main model, sklearn for splitting the training and test data, PIL for converting the images into array of numbers and other libraries such as pandas, NumPy, matplotlib and TensorFlow.

4.3 Retrieving the images:

We will retrieve the images and their labels. Then resize the images to (224,224) as all images should have same size for recognition. Then convert the images into numpy array.

4.4 Splitting the dataset:

This module involves splitting the preprocessed 80% train data and 20% test data.

4.5 Building the model:

Xception model:

•Inspired by Google's Inception model
•Xception is based on an 'extreme' interpretation of the Inception model
•Simple and modular architecture

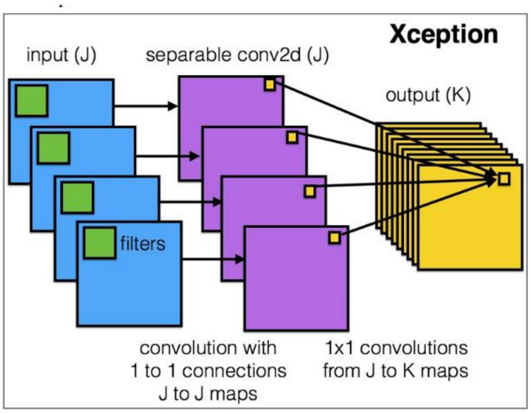
Depthwise Separable Convolution:

Regular Convolutions:

•Look at both channel & spatial correlations simultaneously

Depthwise separable convolution:

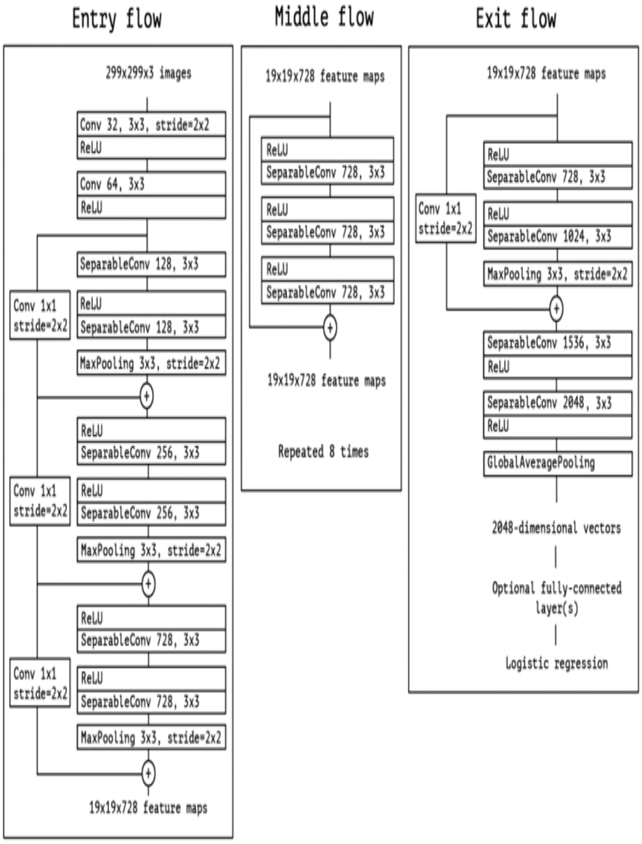
•Look at channel & spatial correlations independently in successive steps
•Spatial convolution: 3x3 convolutions for each channel
•Depthwise convolution: 1x1convolutions on concatenated channels


Xception
input (J)
separable conv2d (J)
output (K)
filters
convolution with 1 to 1 connections J to J maps
1x1 convolutions from J to K maps

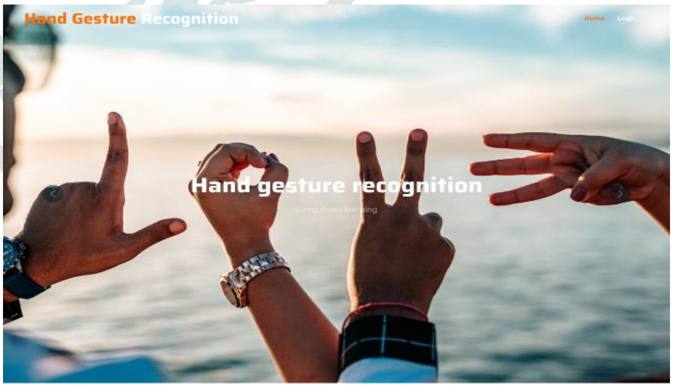
Example: take 3x3 convolutional layer on 16 input channels and 32 output channels.

•regular convolution: 16x32x3x3 = 4608 parameters
•depthwise separable convolution: (spatial conv + depthwise conv) = (16x3x3 + 16x32x1x1) = 656 parameters
•greatly reduced parameter count
•more efficient complexity.

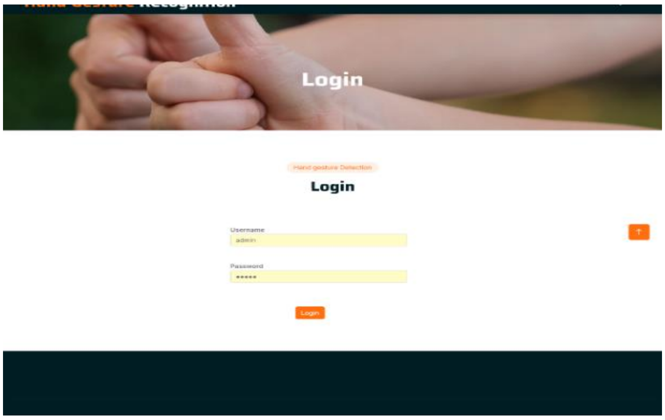
4.6 Xception architecture



5. RESULTS


Hand Gesture Recognition

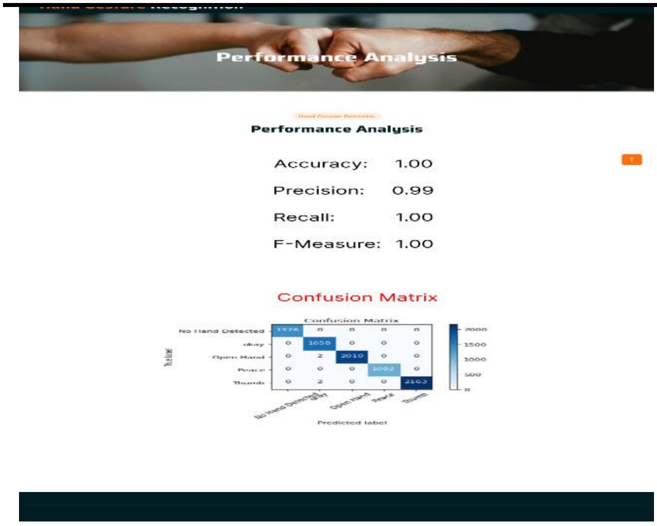
Hand gesture recognition
using deep learning


Hand Gesture Recognition
Login

Hand gesture Detection
Login

Username
admin

Password
•••••



Performance Analysis

Performance Analysis

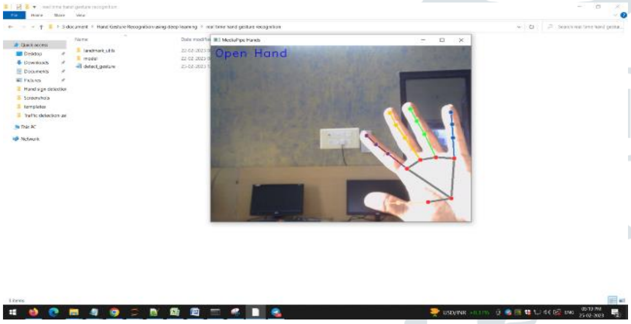
Accuracy: 1.00

Precision: 0.99

Recall: 1.00

F-Measure: 1.00

Confusion Matrix



Conclusion:

This article describes the work done to date in research field of language analysis applications. Certain objectives and criteria have to be met for the application to be efficient. Explaining the functioning of existing systems and laying the foundation for the development of new systems. All the features of the sign-language detection system are described which would be implemented in the new system. The project's significance lies in its potential to foster inclusivity, understanding, and accessibility,creating a universe of effective communication.

References

1. Fig 3.1 by T. Bohra, S. Sompura, K. Parekh and P. Raut, "Real-Time Speech and Hearing Impaired Using Computer Vision and Deep Learning," 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2019, pp. 734-739,doi:10.1109/ICSSIT46314.201 9.8987908.

2. Fig 2 by H. Muthu Mariappan and V. Gomathi, "Real-Time Recognition of Indian Sign Language," 2019 in Data Science (ICCIDS), Chennai,India,2019,pp.1-6,doi: 10.1109/ICCIDS.2019.8862125.

3.Suharjito, Suharjito & Gunawan, Herman & Thiracitta, Narada & Nugroho, Ariadi. (2018). CNN Model. 1-5. 10.1109/INAPR.2018.8627014.

4.Fig 3 by Singha, Joyeeta & Das, Karen. (2013), "Recognition of Indian Sign Language in Live Video," International Journal of Computer Applications. 70. 10.5120/12174-7306.

5. Koller, Oscar & Zargaran, Sepehr & Ney, Hermann & Bowden,Richard. (2016). Deep Sign: Hybrid CNN- HMM for CSL Recognition. 10.5244/C.30.136

6. A. Mittal, P.Kumar, P.R Roy, R Balasubramanian, B.B. Chaudhuri, "A Modified-LSTM Model for Continuous Sign Language Recognition using Leap motion",2019. https://ieeexplore.ieee.org/abstract/

7. A. Mittal, P.Kumar, P.R Roy, R Balasubramanian, B.B. Chaudhuri, "A Modified-LSTM Model for Continuous Sign Language Recognition using Leapmotion",2019. https://ieeexplore.ieee.org/abstract/document/8684245

8. Fig 2 by Elizabth Cody, "Deaftravel", http://www.deaftravel.co.uk/signs.php?id=27