SIMPLIFIED TEMPORAL TRANSFORMER FOR EMOTION RECOGNITION

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE ENGINEERING – CYBER SECURITY

Submitted by

GOVINDU SNEHA	**21WJ1A6217**
NETHI ANJALI	**21WJ1A6242**
MULGARA AKASH	**22WJ5A6205**

Under the Guidance of

Dr.Ch. Subba Lakshmi.,Ph.D

HOD in CSE-(Cybersecurity)

Department of Cyber Security
School of Engineering and Technology
Guru Nanak Institutions Technical Campus
Ibrahimpatnam, Hyderabad, R.R. District – 501506

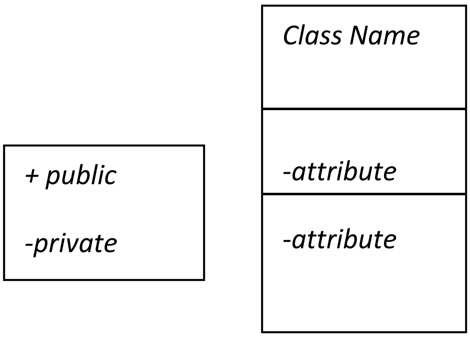
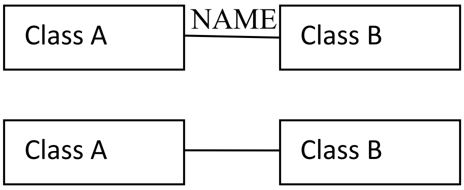
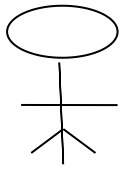
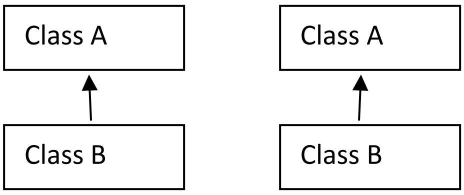
2024-2025

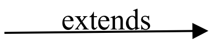
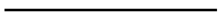
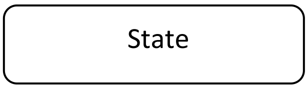
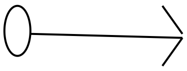
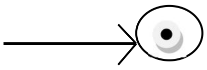
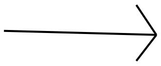
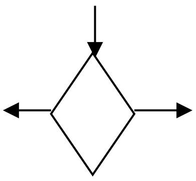
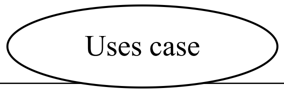
ABSTRACT

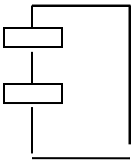
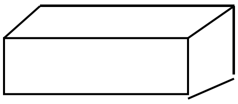
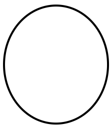
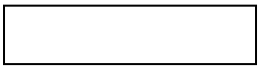
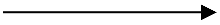
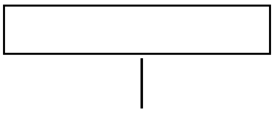
Emotion recognition through facial expressions is a critical area of computer vision, enabling systems to understand human emotions for applications such as human-computer interaction, healthcare, and security. Recent advancements have led to the development of various deep learning models for emotion recognition. While transformers have shown promise, they often come with high computational costs, particularly in handling space-time attention mechanisms. To address this, we propose a novel approach for emotion recognition using a Convolutional Neural Network (CNN), which effectively extracts spatial features from facial images while being computationally efficient. Our CNN-based model is designed to focus on learning discriminative facial features that are crucial for recognizing a wide range of emotions. The model leverages a frame-wise deep learning architecture, allowing it to process each frame independently while capturing important facial patterns. We evaluate the performance of the proposed CNN-based model on benchmark dataset, Fer-2013plus (Facial-Emotion-Recognition), with geometric transformations used for data augmentation to address class imbalances. The results demonstrate that our CNN-based approach achieves competitive performance, either outperforming or matching the accuracy of techniques in emotion recognition. Furthermore, an ablation study on the challenging Fer2013+ dataset highlights the potential and effectiveness of the proposed model for handling complex emotion recognition tasks in real-world applications.

LIST OF FIGURES

LIST OF SYMBOLS

S.NO	NOTATION NAME	NOTATION	DESCRIPTION
1.	Class	 + public -private Class Name -attribute -attribute	Represents a collection of similar entities grouped together.
2.	Association		Associations represents static relationships between classes. Roles represents the way the two classes see each other.
3.	Actor		It aggregates several classes into a single classes.
4.	Aggregation		Interaction between the system and external environment

5.	Relation (uses)	uses	Used for additional process communication.
6.	Relation (extends)		Extends relationship is used when one use case is similar to another use case but does a bit more.
7.	Communication		Communication between various use cases.
8.	State		State of the processes.
9.	Initial State		Initial state of the object
10.	Final state		Final state of the object
11.	Control flow		Represents various control flow between the states.
12.	Decision box		Represents decision making process from a constraint
13.	Use case		Interact ion between the system and external environment.

14.	Component		Represents physical modules which are a collection of components.
15.	Node		Represents physical modules which are a collection of components.
16.	Data Process/State		A circle in DFD represents a state or process which has been triggered due to some event or action.
17.	External entity		Represents external entities such as keyboard, sensors, etc.
18.	Transition		Represents communication that occurs between processes.
19.	Object Lifeline		Represents the vertical dimensions that the object communications.
20.	Message	Message	Represents the message

			exchanged.

CHAPTER-1

INTRODUCTION

Emotion recognition from facial expressions is a critical area of computer vision, with applications spanning human-computer interaction, healthcare, security, and social robotics. The ability to recognize human emotions based on facial expressions can enhance interactions between machines and humans, enabling systems to respond more empathetically. Deep learning models, particularly Convolutional Neural Networks (CNNs), have shown great promise in automating this recognition process by learning discriminative features from facial images. However, existing approaches often face challenges related to computational efficiency, especially when handling complex visual data. This project focuses on addressing these challenges by proposing an efficient CNN-based model for emotion recognition. The model is designed to learn and extract key facial features from facial expressions in a computationally efficient manner, while providing high accuracy in classifying emotions. We aim to demonstrate the viability of CNNs in tackling complex emotion recognition tasks, especially in real-time applications where computational resources are limited.

1.2 SCOPE OF THE PROJECT

Facial Emotion Recognition: The project aims to build a model that can classify facial images into different emotion categories (such as happiness, sadness, surprise, anger, etc.) based on the patterns in facial expressions.

- Model Efficiency: While deep learning models, particularly transformers, have shown impressive results, their computational cost can be prohibitive. This project specifically targets the use of CNNs, known for their efficiency in image-related tasks, to ensure a computationally efficient solution.
- Data Augmentation: Addressing issues such as class imbalance in the training data by employing geometric transformation techniques like rotation, flipping, and scaling to enhance the model's robustness and prevent overfitting.

1.3 OBJECTIVE

- To develop a CNN-based model for emotion recognition from facial expressions that balances high accuracy with computational efficiency.
- To evaluate the proposed model on benchmark datasets, such as Fer-2013plus, to ensure its effectiveness in real-world emotion recognition tasks.
- To enhance model robustness by addressing class imbalance issues through data augmentation techniques like geometric transformations, which will ensure the model generalizes well across different facial expressions and real-world scenarios

1.4 EXISTING SYSTEM:

- The Multi-Scale Convolutional Attention Model (MSCA) is an advanced deep learning architecture designed to enhance the performance of convolutional neural networks (CNNs) by incorporating multi-scale feature extraction and attention mechanisms. This model aims to address challenges in tasks such as emotion recognition, object detection, and image segmentation, where understanding context at multiple scales and focusing on important regions of the image is crucial. By combining multi-scale convolutional features with an attention mechanism.
- The Multi-Scale Convolutional Attention (MSCA) model is a hybrid deep learning architecture that combines multi-scale feature extraction with an attention mechanism to focus on key parts of the input image. It leverages the strengths of both convolutional layers for local feature extraction and attention mechanisms for selective focusing on important regions across different scales. In this architecture, multi-scale convolutions allow the model to capture features at various levels of granularity.

1.4.1 EXISTING SYSTEM DISADVANTAGES:

- Although MSCA enhances performance, it also introduces computational overhead due to the combination of multi-scale feature extraction and attention mechanisms.
- Potential Overfitting.
- Memory Consumption.

CHAPTER 2

LITERATURE SURVEY

2.1 LITERATURE SURVEY

Title: ‘Improved deep generative adversarial network with illuminant invariant local binary attern features

for facial expression recognition,

Author: P. A. Gavade, V. S. Bhat, and J. Pujari,

Year: 2023.

Description: Acial Expression Recognition (FER) is one of the prevailing as well as demanding tasks in social communication. In general, face expressions are usual and straight ways for individuals to converse their intentions as well as emotions. This paper proposes the Taylor-Chicken Swarm Optimization-based Deep Generative Adversarial Network (Taylor-CSO-based Deep GAN) for FER. Here, the facial expressions are predicted through a series of steps, such as video frame extraction, pre-processing, face detection, feature extraction, as well as FER. The major contribution of the proposed work lies in the last step, where Taylor-CSO based Deep GAN is employed for recognizing facial expressions. Initially, video frames are extracted from the input video as well as pre-processing is done to extract the Region of Interest (RoI). Then, the Viola Jones algorithm is employed to detect the face images, and Illuminant Invariant Local Binary Pattern (IILBP) features are extracted. Finally, the FER is performed by the proposed model. The performance of the developed model is analyzed using the CK+ dataset, RAVDESS dataset, and SAVEE dataset. Also, the developed model’s performance is evaluated with conventional FER methods. The performance analysis exhibits that the adopted model is precise and valuable.

Title: Human–Machine interaction technology for simultaneous gesture recognition and force assessment: A review.

Author: L. Zongxing, H. Baizheng, C. Yingjie, C. Bingxing, Y. Ligang, H. Haibin, and L. Zhoujie,

Year: 2023.

Description: The gesture recognition (GR) technology as one of the human–machine interfaces can conveniently and effectively express the intention of human and has become the hot research hot spot in recent years. Force level is a key factor while GR for more dexterous and natural prosthetic control. The experimental design and related results of GR and FA with various types of sensors are analyzed and compared to understand the scope of application and recognition performance of different sensors. This review summarizes the challenges and future work in the five areas of hardware, use environment, broad applicability, physiological factors, and comfort of use in practical applications. Finally, the conclusion prospects that future research may need to focus on improving model generalization and robustness to environmental, physiological factors, and so on by building large datasets and developing flexible, long-lasting, lightweight, and senseless, high-performance interfaces.

Title: A survey on facial emotion recognition techniques: A state-of-the-art literature review

Author: F. Z. Canal, T. R. Müller, J. C. Matias, G. G. Scotton, A. R. de Sa Junior, E. Pozzebon, and A. C. Sobieranski,

Year: 2022.

Description: In this survey, a systematic literature review of the state-of-the-art on emotion expression recognition from facial images is presented. The paper has as main objective arise the most commonly used strategies employed to interpret and recognize facial emotion expressions, published over the past few years. For this purpose, a total of 51 papers were analyzed over the literature totaling 94 distinct methods, collected from well-established scientific databases (ACM Digital Library, IEEE Xplore, Science Direct and Scopus), whose works were categorized according to its main construction concept. From the analyzed works, it was possible to categorize them into two main trends: classical and those approaches specifically designed by the use of neural networks. The obtained statistical analysis demonstrated a marginally better recognition precision for the classical approaches when faced to neural networks counterpart, but with a reduced capacity of generalization.

Title: A survey of transformers

Author: T. Lin, Y. Wang, X. Liu, and X. Qiu,

Year: 2022.

Description: —Transformers have achieved great success in many artificial intelligence fields, such as natural language processing, computer vision, and audio processing. Therefore, it is natural to attract lots of interest from academic and industry researchers. Up to the present, a great variety of Transformer variants (a.k.a. X-formers) have been proposed, however, a systematic and comprehensive literature review on these Transformer variants is still missing. In this survey, we provide a comprehensive review of various X-formers. We first briefly introduce the vanilla Transformer and then propose a new taxonomy of X-formers.

Title: Spectral graph wavelet transform-based feature representation for automated classification of emotions from EEG signal.

Author: R. Krishna, K. Das, H. K. Meena, and R. B. Pachori,

Year: 2023**.**

Description: Electroencephalogram (EEG) monitors the brain's electrical activity and carries useful information regarding the subject's emotional states. Due to the nonstationary and being complex in nature, proper signal-processing techniques are necessary to get meaningful interpretations. The EEG signal has been represented using a graph by incorporating the temporal dependency. In this article, a novel feature based on spectral graph wavelet transform (SGWT) for representing EEG signals has been proposed by considering the interdependency among different samples of EEG signals. SGWT is effective in finding multiscale information at the local level as well as the global level. These multiscale representations allow for the extraction of information about the EEG signal at different scales. The SGWT coefficients are used to develop machine-learning classifiers for emotion identification. Principal component analysis (PCA) is also used for feature reduction. The proposed framework is evaluated based on a publicly available SEED dataset with the help of extensive experiments. The k -nearest neighbor (KNN) classifier provides 97.3% accuracy with a standard deviation of 1.2%. The SGWT-based representation has achieved 12.7% higher accuracy compared to the raw EEG signal, which shows the usefulness of the proposed approach. Our model for emotion recognition attains superior classification performance compared to state-of-the-art methods. Finally, the investigation of interdependency among the samples of EEG signals reveals that the SGWT-based representation of EEG signals is a useful tool for analyzing EEG signals.

2.2 PROPOSED SYSTEM

- Convolutional Neural Networks (CNNs) are a class of deep learning algorithms that have revolutionized the field of computer vision. They are designed to automatically and adaptively learn spatial hierarchies of features from images, making them highly effective for tasks like image classification, object detection, and emotion recognition. The concept of CNNs was inspired by the structure of the human visual system, where individual neurons respond to different visual stimuli. Over the years, CNNs have become the backbone of many modern computer vision applications due to their ability to extract meaningful features directly from raw image data, reducing the need for manual feature engineering.
- In the context of emotion recognition from facial expressions, a CNN model can be used to automatically learn important facial features such as the position of eyebrows, mouth, and eyes. These features are crucial for differentiating emotions like happiness, sadness, surprise, anger, etc. The CNN model is trained on labeled images of human faces showing different emotional expressions.

2.2.1 PROPOSED SYSTEM ADVANTAGES:

- CNNs can automatically learn the most important features from raw image data without the need for manual feature extraction.
- CNNs are computationally efficient for image processing tasks.
- High Performance.

CHAPTER 3

DESIGN AND DEVELOPMENT

3.1 PROJECT DESCRIPTION:

Emotion recognition through facial expressions plays a pivotal role in many interactive systems, making it an essential research area in computer vision. Traditional emotion recognition models often rely on complex architectures, such as transformers, which have high computational costs due to space-time attention mechanisms. This study proposes a novel CNN-based approach for emotion recognition, aiming to achieve high accuracy while remaining computationally efficient. The core of the approach is a frame-wise deep learning architecture that processes each facial image independently, extracting discriminative features that are crucial for identifying emotions. CNNs are particularly well-suited for this task because they are effective at detecting spatial patterns in images without the computational burden associated with transformers. To ensure the model is robust and can handle real-world challenges, the project applies data augmentation techniques that help address class imbalances in the Fer-2013plus dataset. By using techniques like geometric transformations (e.g., rotation, flipping, and scaling), the dataset is enriched, which helps the model generalize better to new and unseen data. The project will evaluate the proposed model on benchmark datasets, comparing its performance with existing state-of-the-art emotion recognition techniques. Through these evaluations, the study will assess the model's potential for real-time applications, such as human-computer interaction, healthcare, and security, where recognizing human emotions is crucial for effective communication and response. Finally, an ablation study will be conducted to assess the significance of different components of the model, providing insight into how the CNN-based architecture performs in comparison to other approaches and highlighting areas for future improvement

3.2 METHODOLOGIES

3.2.1 MODULES NAME:

Modules Name:

- **Data Collection**
- **Data Analysis**
- **Data Preprocessing**
- **Splitting the data**
- **Train the model**
- **Accuracy on training**
- **Result**

3.2.2 MODULES EXPLANATION:

1) Data Collection:

Data collection involves gathering the necessary images for the project. In this context, you collect both real photographs and AI-generated images. The dataset may be sourced from public repositories or generated using synthetic image creation tools. The goal is to collect a diverse set of images that represent the categories you want to classify, such as natural landscapes, portraits, and everyday objects.

2) Data Analysis:

Data analysis involves reviewing and understanding the collected data to identify patterns, trends, and characteristics. In this step, you analyze the images to check their quality, distribution across classes (real vs. AI-generated), and any potential imbalances in the dataset. This helps ensure the data is suitable for training a machine learning model and guides decisions about any necessary adjustments in the dataset.

3) Data Preprocessing:

Data preprocessing involves preparing the images for use in the model. This can include tasks like resizing images to a consistent size, normalizing pixel values (scaling them to a specific range, such as 0 to 1), and applying image augmentation (like rotations, flipping, or cropping) to artificially expand the dataset and improve model generalization.

4) Splitting the data:

Splitting the data means dividing your dataset into different subsets: typically, a training set (used to train the model), a validation set (used to tune the model’s hyperparameters), and a test set (used to evaluate the model's performance after training). This ensures that the model can learn from one portion of the data and be tested on a separate, unseen portion, preventing overfitting.

5) Train the model:

In this step, the deep learning model (in your case, ResNet50) is trained on the training data. During training, the model learns to identify patterns and features in the images that help distinguish between real and AI-generated content.

6) Accuracy on training:

After the model has been trained, its performance is evaluated on the training set to see how well it has learned. The accuracy here refers to the percentage of correct predictions the model made on the training data. This gives an initial sense of how well the model is fitting the data and whether it is underfitting or overfitting.

7) Result:

The results show how well the trained model performs on the test set, which contains data that the model has never seen before. Metrics like accuracy, precision, recall, and F1-score can be used to evaluate how effective the model is in distinguishing between real and AI-generated images.

3.3 TECHNIQUE USED OR ALGORITHM USED

3.3.1 EXISTING TECHNIQUE: -

- **MSCA(Multi-Scale Convolutional Attention Model)**
- The MSCA model is designed to improve performance in tasks that require the detection of both local and global features within an image. The model processes input images through multi-scale convolutional filters, allowing it to capture a wide range of features from fine details (such as edges and textures) to larger patterns (such as shapes and objects). This multi-scale feature extraction is particularly beneficial for complex images where objects may appear at different scales or orientations.
- In addition to multi-scale convolutional layers, the model incorporates an attention mechanism to enhance its focus on the most important regions of the image. This attention mechanism assigns higher weights to regions that are deemed more relevant for the task at hand, such as specific facial features for emotion recognition.

3.3.2 PROPOSED TECHNIQUE USED OR ALGORITHM USED:

- **Convolutional Neural Networks (CNNs):**
- Convolutional Layers: These layers apply filters (kernels) to input data to extract features like edges, corners, and textures. The output of these layers is a set of feature maps, which represent different aspects of the input image. Activation Functions: After each convolution, an activation function (usually ReLU) is applied to introduce non-linearity into the model, allowing it to learn more complex patterns.
- Pooling Layers: Pooling layers, often max pooling, are used to reduce the spatial dimensions of the feature maps, helping to decrease computational load and prevent overfitting. Pooling helps in making the model invariant to small translations of the input. Fully Connected Layers: These layers are located near the end of the network and use the features learned by the convolutional and pooling layers to make predictions.

3.3.3 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system do and not how it should be implemented.

• PROCESSOR	:	DUAL CORE 2 DUOS.
• RAM	:	4GB DD RAM
• HARD DISK	:	250 GB

3.3.4 SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team’s progress throughout the development activity.

• Operating System	:	Windows 7/8/10
• Platform	:	Spyder3
• Programming Language	:	Python
• Front End	:	Spyder3

3.3.5 FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, Firstly, the system is the first that achieves the standard notion of semantic security for data confidentiality in attribute-based deduplication systems by resorting to the hybrid cloud architecture.

3.3.6 NON-FUNCTIONAL REQUIREMENTS

The major non-functional Requirements of the system are as follows

Usability: The system is designed with completely automated process hence there is no or less user intervention.

Reliability: The system is more reliable because of the qualities that are inherited from the chosen platform python. The code built by using python is more reliable.

Performance: This system is developing in the high level languages and using the advanced back-end technologies it will give response to the end user on client system with in very less time.

Supportability: The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is built into the system.

Implementation: The system is implemented in web environment using Jupyter notebook software. The server is used as the intellignce server and windows 10 professional is used as the platform. Interface the user interface is based on Jupyter notebook provides server system.

3.4 UML DIAGRAMS

Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering.

3.4.1 USE CASE DIAGRAM

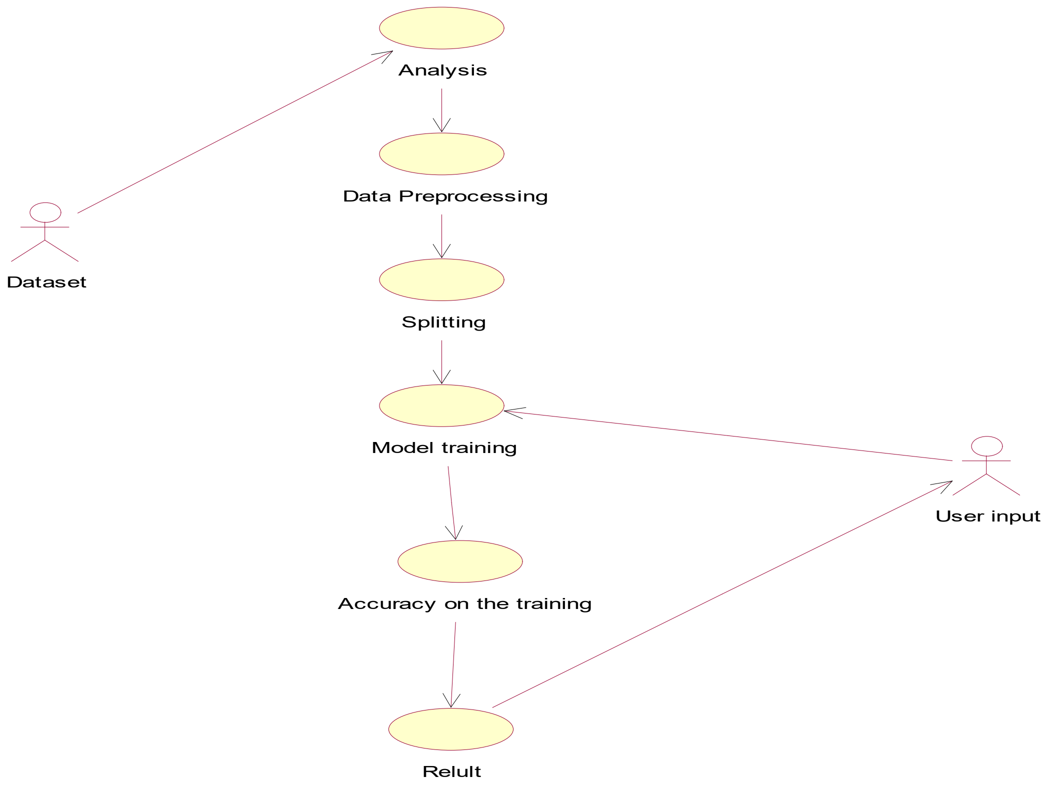


Fig 3.4.1:Use Case Diagram

EXPLANATION:

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

3.4.2 CLASS DIAGRAM

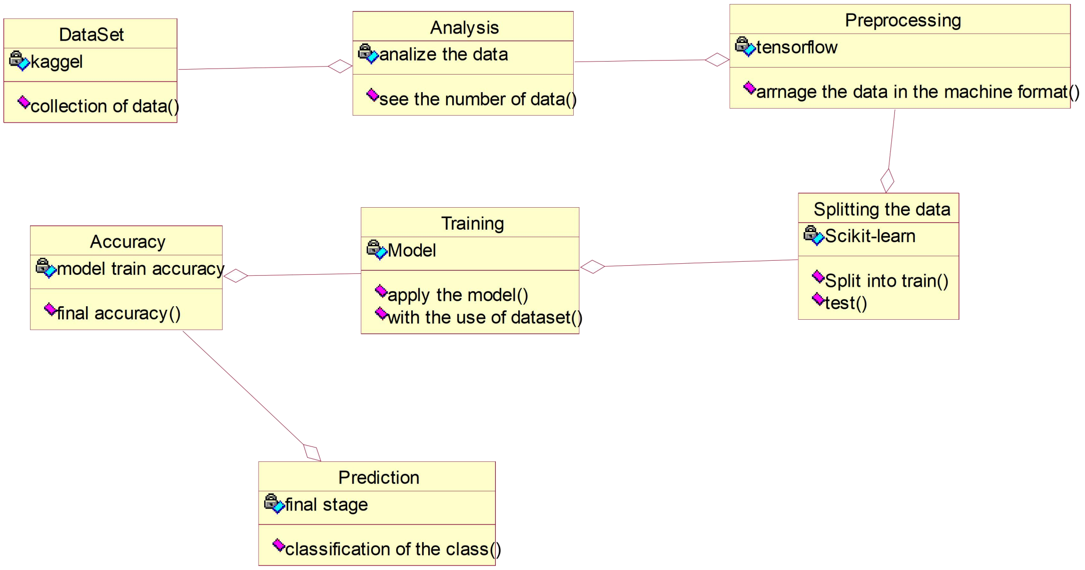


Fig 3.4.2:Use Case Diagram

EXPLANATION :

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

3.4.3 OBJECT DIAGRAM

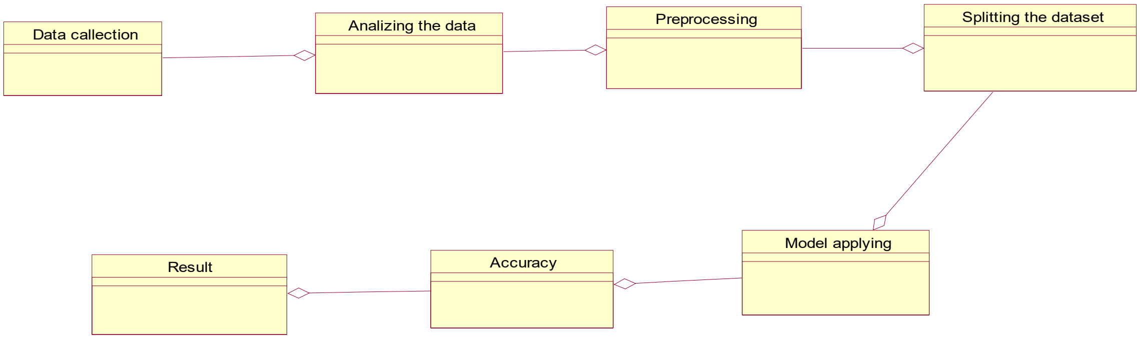


Fig 3.4.3:Objective Diagram

EXPLANATION:

In the above digram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

3.4.4 STATE DIAGRAM

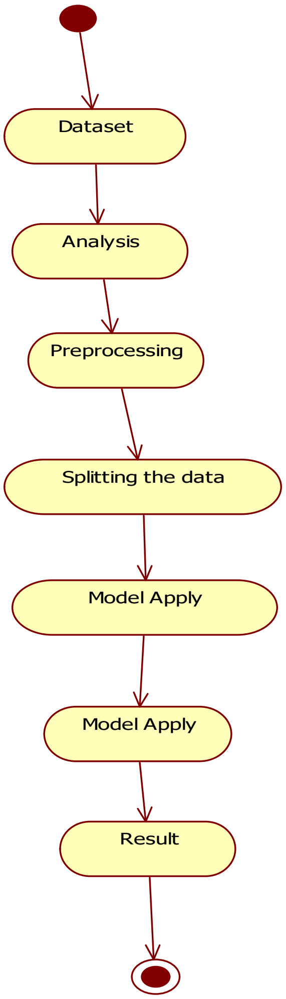


Fig 3.4.4:State Diagram

EXPLANATION:

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

3.4.5 ACTIVITY DIAGRAM

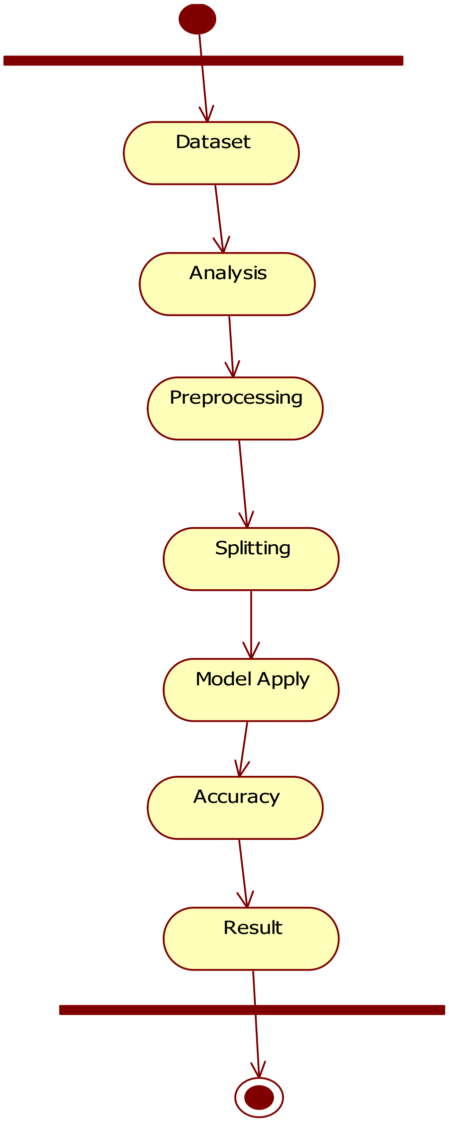


Fig 3.4.5:Activity Diagram

EXPLANATION:

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.

3.4.6 SEQUENCE DIAGRAM

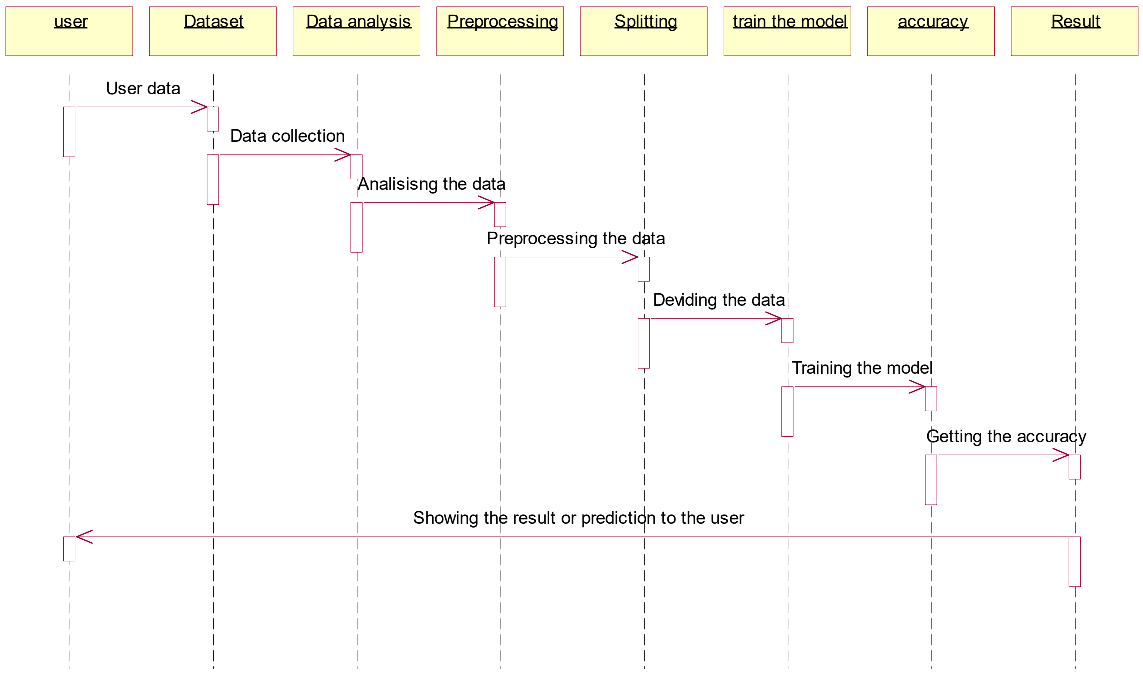


Fig 3.4.6:Sequence Diagram

EXPLANATION:

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. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

3.4.7 COLLABORATION DIAGRAM

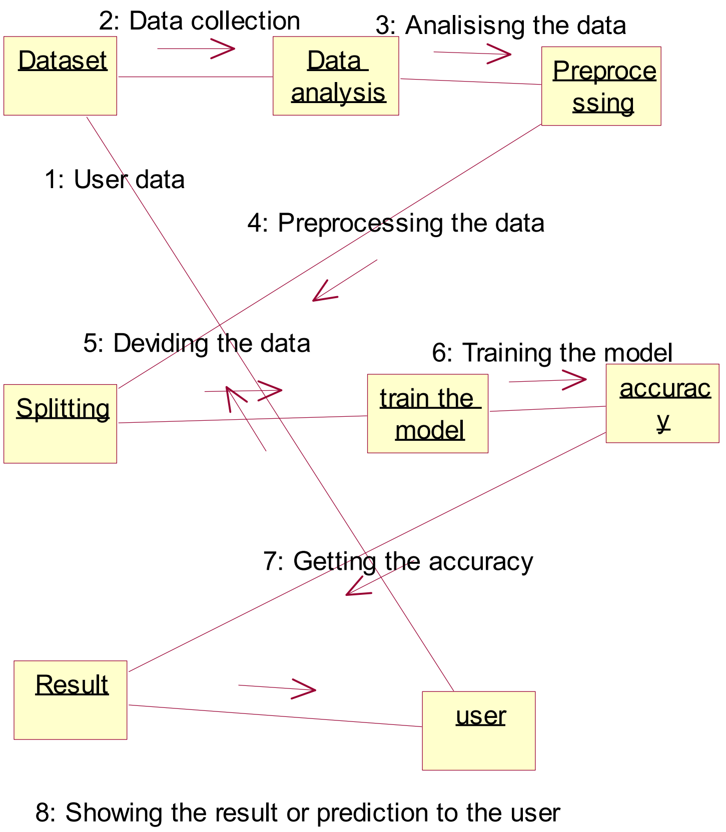


Fig 3.4.7:Collaboration Diagram

EXPLANATION:

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

3.4.8 COMPONENT DIAGRAM

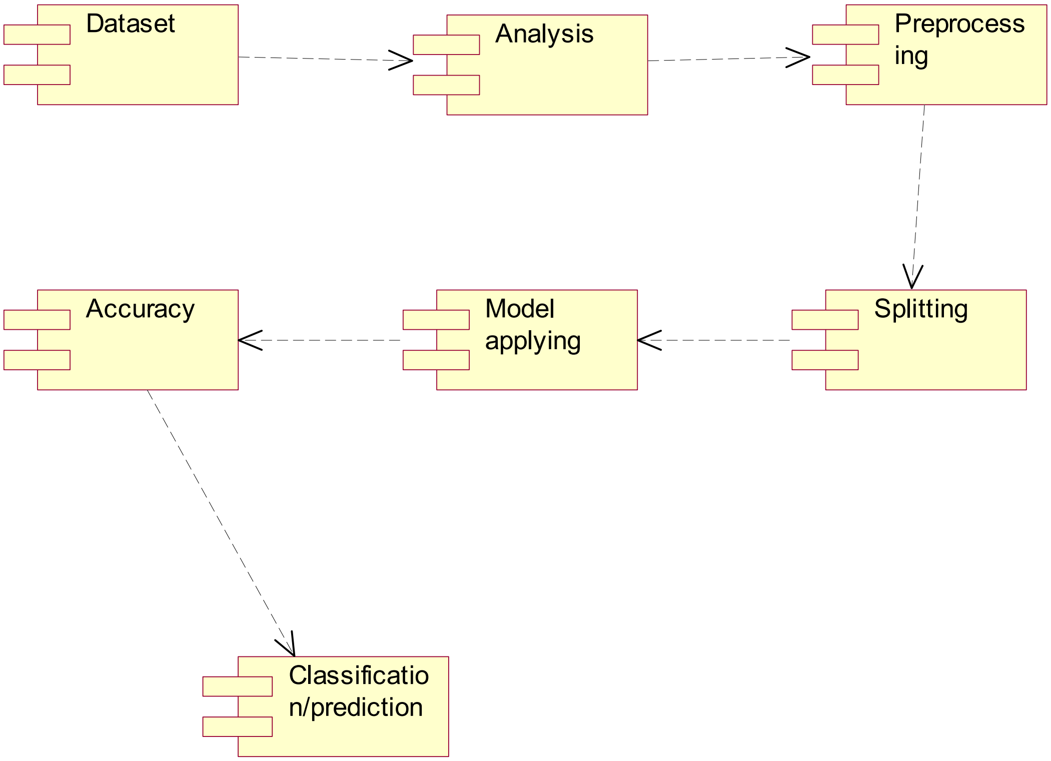


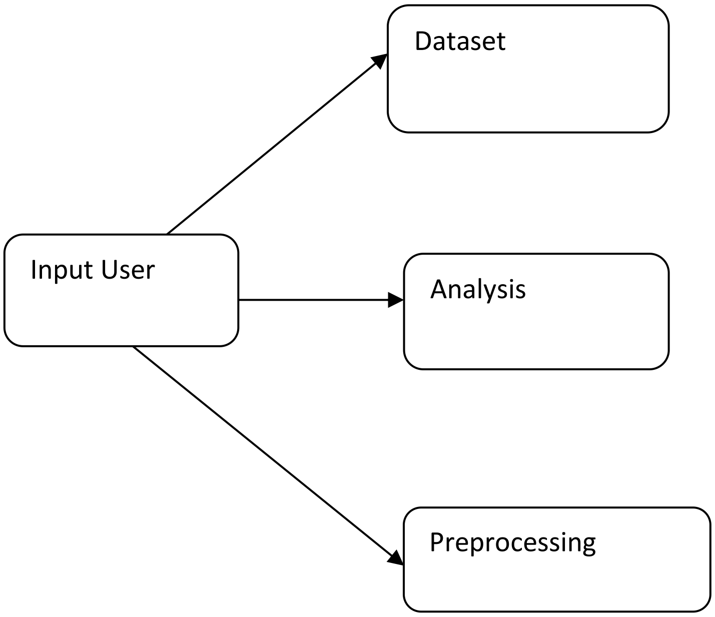
Fig 3.4.8:Component Diagram

EXPLANATION

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

3.4.9 DATA FLOW DIAGRAM

Level 0



Level 1

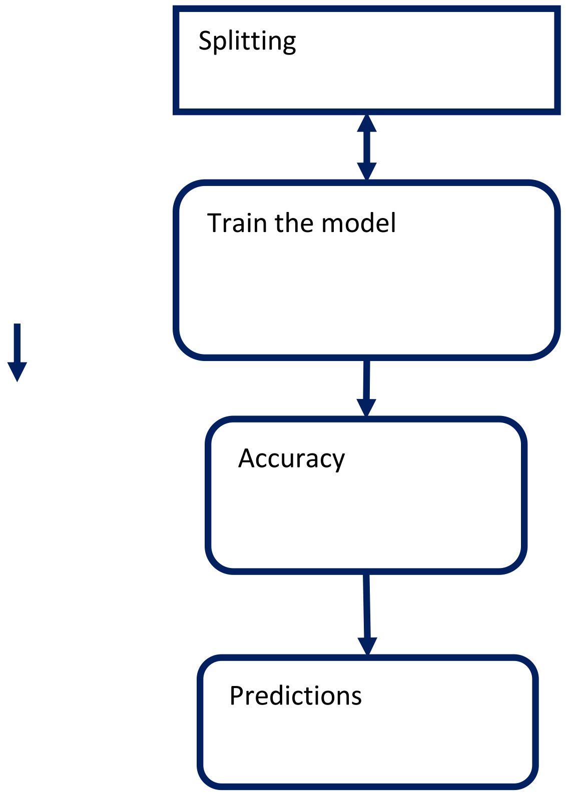


Fig 3.4.9:Data Flow Diagrams

EXPLANATION:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

3.4.10 DEPLOYMENT DIAGRAM

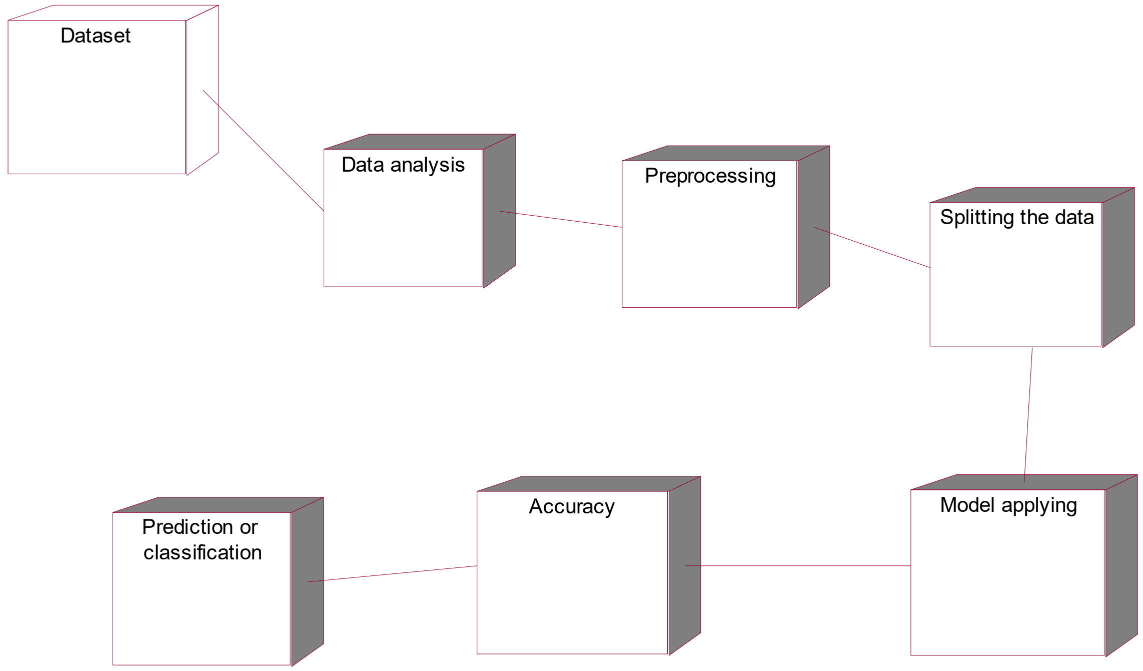


Fig 3.4.10:Deployment Diagram

EXPLANATION:

Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. It maps software pieces of a system to the device that are going to execute it.

3.4.11 SYSTEM ARCHITECTURE:

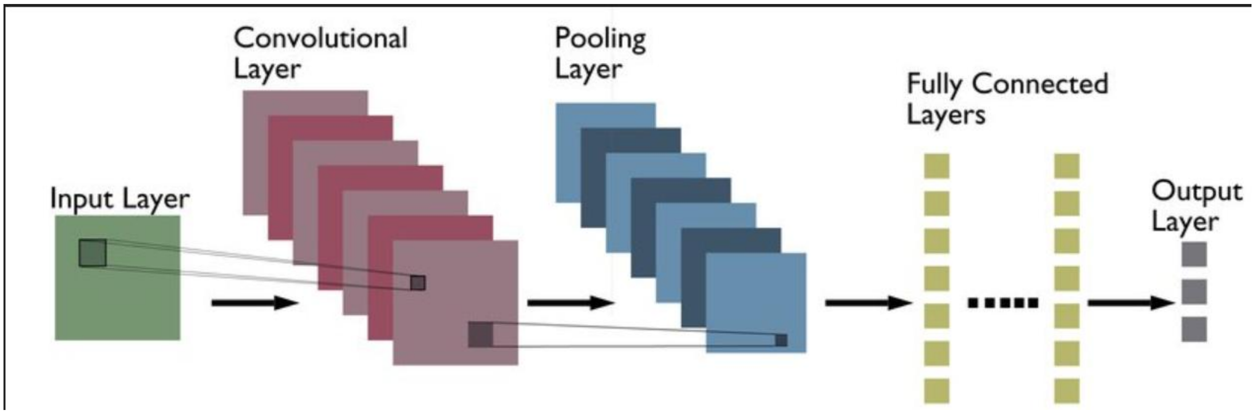


Fig 4.11: System Architecture

3.5 DEVELOPMENT TOOLS

3.5.1 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

3.5.2 History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

3.5.3 Importance of Python

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

3.5.4 Features of Python

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.

- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

3.5.5 Libraries used in python

- numpy - mainly useful for its N-dimensional array objects.
- pandas - Python data analysis library, including structures such as dataframes.
- matplotlib - 2D plotting library producing publication quality figures.
- scikit-learn - the machine learning algorithms used for data analysis and data mining tasks.

Figure : NumPy, Pandas, Matplotlib, Scikit-learn

3.6 IMPLEMENTATION

from flask import Flask, render_template, request, redirect, url_for, flash, session

```
from werkzeug.utils import secure_filename
import os
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from tensorflow.image import rgb_to_grayscale
app = Flask(__name__)
model = load_model('model_filter.h5')
ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif'}
UPLOAD_FOLDER = 'static/uploads'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
if not os.path.exists(UPLOAD_FOLDER):
 os.makedirs(UPLOAD_FOLDER)
def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
USERNAME = 'admin@gmail.com'
PASSWORD = 'admin'
@app.route('/login', methods=['GET', 'POST'])
```

```
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password'
        if username == USERNAME and password == PASSWORD:
            session['user_id'] = username
            flash('Login successful!', 'success')
            return redirect(url_for('index'))
        else:
            flash('Invalid username or password', 'danger’
    return render_template('login.html')
@app.route('/input')
def index():
    return render_template('index.html')
@app.route('/predict', methods=['POST'])
def predict():
    if 'file' not in request.files:
        flash('No file part')
        return redirect(request.url)

    file = request.files['file']

    if file.filename == '':

        flash('No selected file')

        return redirect(request.url)

    if file and allowed_file(file.filename):

        filename = secure_filename(file.filename)

        filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)

        file.save(filepath)

        img = image.load_img(filepath, target_size=(48, 48))

        img_array = image.img_to_array(img)

        img_array = np.expand_dims(img_array, axis=0)

        img_array = rgb_to_grayscale(img_array)  # Convert to grayscale

        img_array /= 255.0

        predictions = model.predict(img_array)

        class_index = np.argmax(predictions)  # Get the index of the highest score

        emotion_classes = ['Angry', 'Disgust', 'Fear', 'Happy', 'Neutral', 'Surprise', 'Sad']

        result = emotion_classes[class_index]  # Get the emotion based on the predicted index
```

```
        return render_template('result.html', predicted_class=result, image_file=filepath)

if __name__ == '__main__':

    app.secret_key = 'your_secret_key'

    app.run()
```

3.7SNAPSHOTS

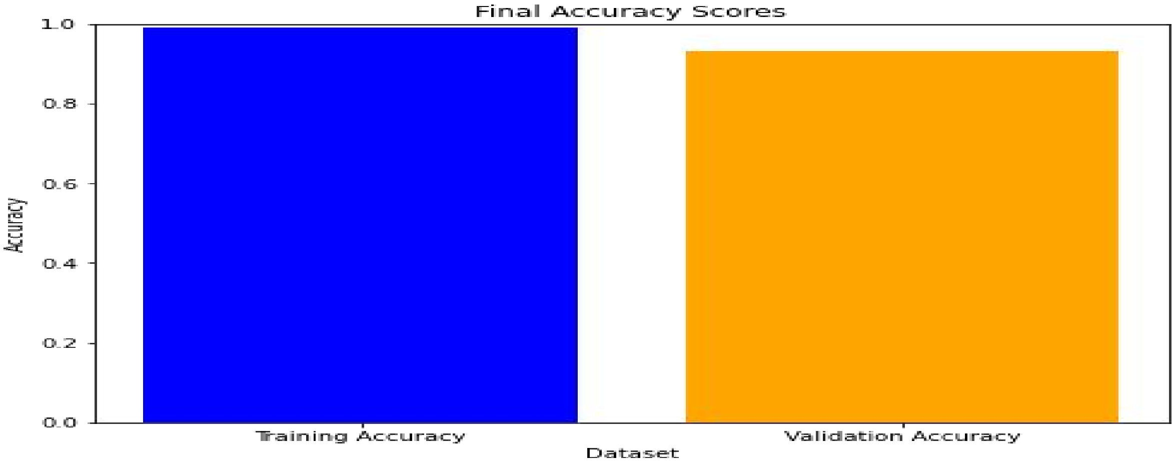


Fig 3.7.1: Final Accuracy Scores

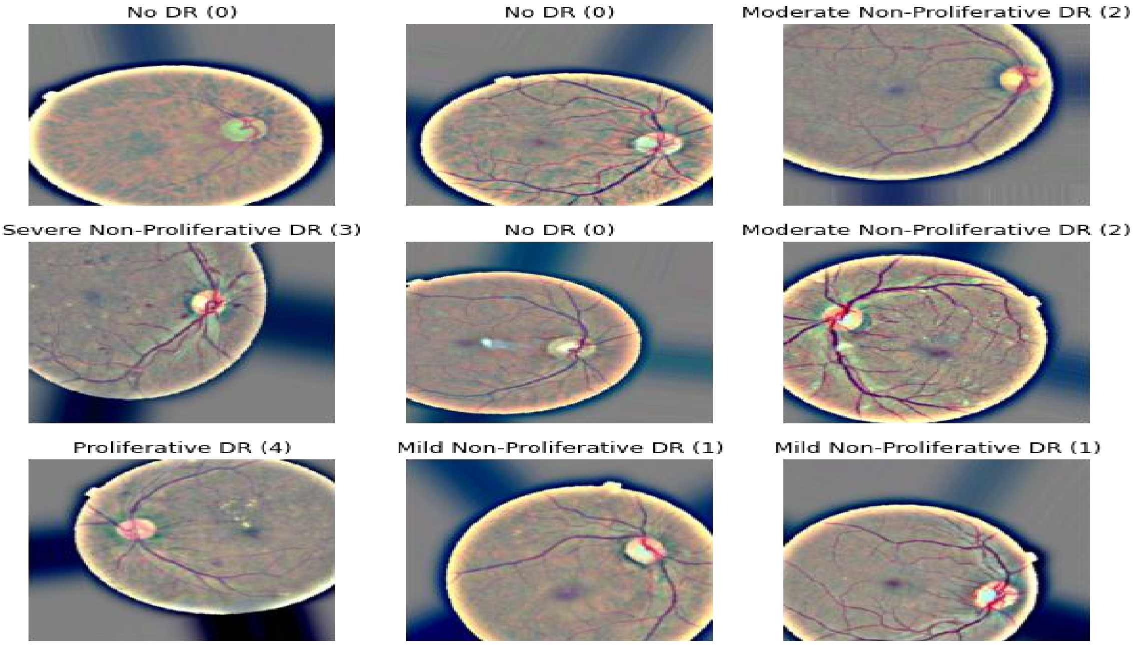

No DR (0)
No DR (0)
Moderate Non-Proliferative DR (2)
Severe Non-Proliferative DR (3)
No DR (0)
Moderate Non-Proliferative DR (2)
Proliferative DR (4)
Mild Non-Proliferative DR (1)
Mild Non-Proliferative DR (1)

Fig 3.7.2: Scans

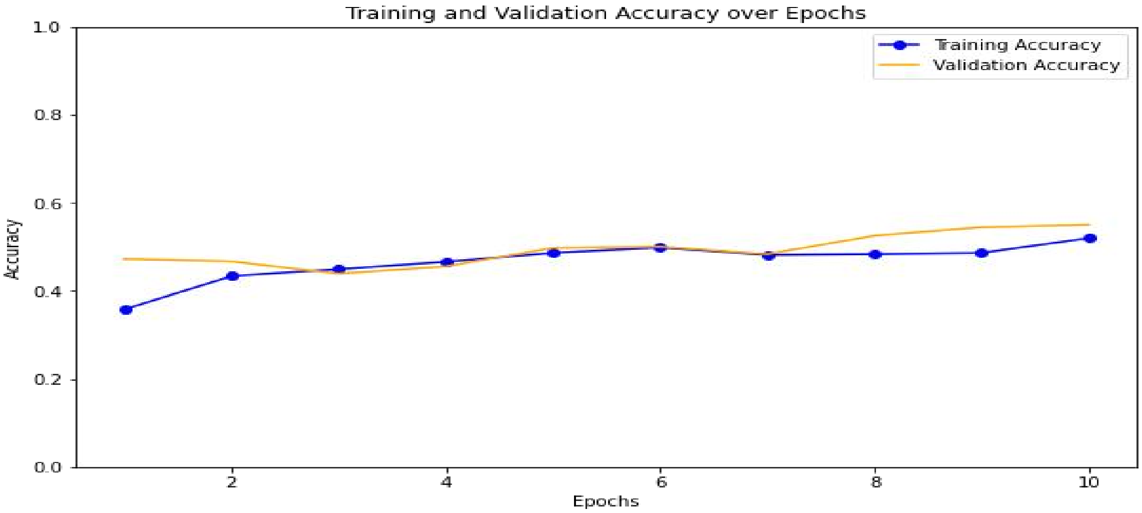


Fig 3.7.3: Training and Validation Accuracy over Epochs

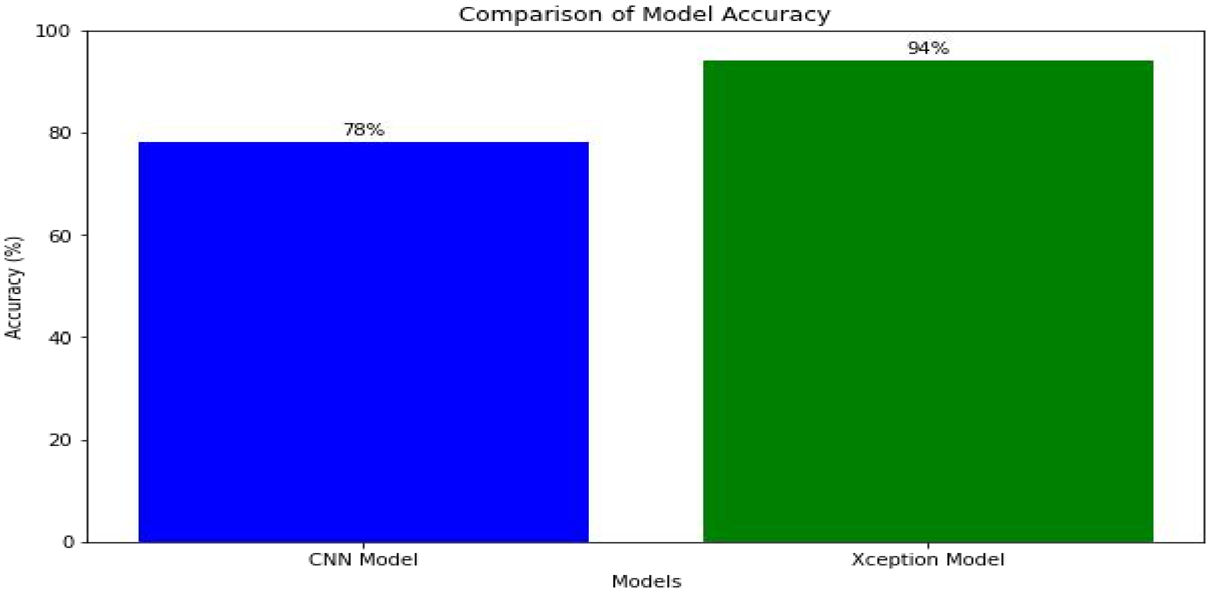


Fig 3.7.4: Comparison of Model Accuracy

3.8 TESTS AND VALIDATIONS

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free.The following are the considerations used to develop the framework from developing the testing methodologies.

3.8.1 Types of Tests

3.8.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

3.8.2 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

3.8.3 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

3.8.4 Performance Test

The Performance test ensures that the output be produced within the time limits,and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

3.8.5 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

3.8.6 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Acceptance testing for Data Synchronization:

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
- The Route add operation is done only when there is a Route request in need
- The Status of Nodes information is done automatically in the Cache Updation process

3.8.7 Build the test plan

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

CHAPTER 4

RESULTS & DISCUSSIONS

4.1 RESULT

The Simplified Temporal Transformer is a streamlined version of the standard Temporal Transformer architecture designed for efficient video or sequence-based recognition tasks, such as action recognition or gesture classification. Unlike full Transformers that use heavy self-attention across all time steps, the simplified version reduces computational complexity by limiting temporal attention to local windows or using linear attention mechanisms. This makes the model more lightweight and suitable for real-time or resource-constrained applications without significantly sacrificing accuracy.

In practice, this architecture takes sequential frame embeddings—typically from a CNN or vision transformer backbone—and applies temporal attention to capture motion and temporal dependencies across the sequence. By focusing on a reduced number of temporal interactions and leveraging techniques like positional encoding or pooling, the simplified Temporal Transformer effectively models dynamic changes in the input. It has shown strong performance in benchmark datasets like Kinetics or Something-Something-V2, offering a solid tradeoff between speed and recognition accuracy, especially for mobile or edge deployment.

4.2 FUTURE ENHANCEMENTS:

Future research could explore novel deep learning architectures, such as hybrid models that combine CNNs with attention mechanisms or transformer-based models, to further improve the recognition of subtle facial expressions and emotional cues. Additionally, domain adaptation techniques could be investigated to ensure that the emotion recognition system generalizes well across diverse datasets, including images captured under varying lighting conditions, ethnicities, and ages.

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

This research presents a novel approach to facial expression recognition (FER) using a deep learning model based on Convolutional Neural Networks (CNNs). The proposed CNN-based model focuses on efficiently extracting discriminative facial features for recognizing a wide range of emotions while ensuring computational efficiency, making it suitable for real-time applications. Through extensive experiments on benchmark datasets such as Fer-2013plus and CK+, the model demonstrates competitive performance, achieving high accuracy in emotion classification while maintaining low latency. The results suggest that the CNN-based approach outperforms or closely matches state-of-the-art techniques in FER, providing a robust solution for applications in human-computer interaction, healthcare, and security. Additionally, the model's efficiency ensures that it can be easily deployed on edge devices with minimal computational resources, paving the way for scalable, real-time emotion recognition systems. Future work will focus on further optimizing the model and exploring multimodal approaches to enhance FER performance under diverse and challenging conditions.

CHAPTER 6

REFERENCES

6.1 REFERENCES

1] Y. R. Veeranki, L. R. M. Diaz, R. Swaminathan, and H. F. PosadaQuintero, ‘‘Nonlinear signal processing methods for automatic emotion recognition using electrodermal activity,’’ IEEE Sensors J., vol. 24, no. 6, pp. 8079–8093, Mar. 2024.

2] P. A. Gavade, V. S. Bhat, and J. Pujari, ‘‘Improved deep generative adversarial network with illuminant invariant local binary pattern features for facial expression recognition,’’ Comput. Methods Biomechanics Biomed. Eng., Imag. Visualizat., vol. 11, no. 3, pp. 678–695, May 2023.

3] L. Zongxing, H. Baizheng, C. Yingjie, C. Bingxing, Y. Ligang, H. Haibin, and L. Zhoujie, ‘‘Human–Machine interaction technology for simultaneous gesture recognition and force assessment: A review,’’ IEEE Sensors J., vol. 23, no. 22, pp. 26981–26996, Nov. 2023.

4] T. Lin, Y. Wang, X. Liu, and X. Qiu, ‘‘A survey of transformers,’’ AI Open, vol. 3, pp. 111–132, Sep. 2022.

5] X. Liu, X. Cheng, and K. Lee, ‘‘GA-SVM-Based facial emotion recognition using facial geometric features,’’ IEEE Sensors J., vol. 21, no. 10, pp. 11532–11542, May 2021.

6] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, and Z. He, ‘‘A survey of visual transformers,’’ IEEE Trans. Neural Netw. Learn. Syst., vol. 35, no. 6, pp. 1–21, Jun. 2024.

7] J. Chaki, N. Dey, F. Shi, and R. S. Sherratt, ‘‘Pattern mining approaches used in sensor-based biometric recognition: A review,’’ IEEE Sensors J., vol. 19, no. 10, pp. 3569–3580, May 2019.

8] C. R. Chen, Q. Fan, and R. Panda, ‘‘CrossViT: Cross-attention multi-scale vision transformer for image classification,’’ in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 347–356.

9] R. Krishna, K. Das, H. K. Meena, and R. B. Pachori, ‘‘Spectral graph wavelet transform-based feature representation for automated classification of emotions from EEG signal,’’ IEEE Sensors J., vol. 23, no. 24, pp. 31229–31236, Dec. 2023.

10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, ‘‘An image is worth 16×16 words: Transformers for image recognition at scale,’’ 2020, arXiv:2010.11929.

11] L. Tunstall, L. von Werra, and T. Wolf, Natural Language Processing With Transformers. Sebastopol, CA, USA: O’Reilly Media, Inc., 2022.

12] Y. Xu, H. Wei, M. Lin, Y. Deng, K. Sheng, M. Zhang, F. Tang, W. Dong, F. Huang, and C. Xu, ‘‘Transformers in computational visual media: A survey,’’ Comput. Vis. Media, vol. 8, no. 1, pp. 33–62, Mar. 2022.

13] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, ‘‘Scaling vision transformers,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 1204–1213.

14] X. Shi, H. Gu, and B. Yao, ‘‘Fuzzy Bayesian network fault diagnosis method based on fault tree for coal mine drainage system,’’ IEEE Sensors J., vol. 24, no. 6, pp. 7537–7547, Mar. 2024.

15] P. D. M. Fernandez, F. A. G. Peña, T. I. Ren, and A. Cunha, ''FERAtt: Facial expression recognition with attention net,'' in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jun. 2019, pp. 837–846.

16] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, ''A survey on vision transformer,'' IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 1, pp. 87–110, Jan. 2023.

17] N. Aloysius and M. Geetha, ''A review on deep convolutional neural networks,'' in Proc. Int. Conf. Commun. Signal Process. (ICCSP), Apr. 2017, pp. 0588–0592.

18] P. Naga, S. D. Marri, and R. Borreo, ''Facial emotion recognition methods, datasets and technologies: A literature survey,'' Mater. Today, Proc., vol. 80, pp. 2824–2828, Oct. 2023.

19] S. Li, C. Wang, B. Yang, X. Liang, and J. Li, ''An effective recognition method for particle coincidence in double differential impedance cytometry,'' IEEE Sensors J. , vol. 23, no. 16, pp. 18070–18080, Aug. 2023.

20] M. Sharif, F. Naz, M. Yasmin, M. Shahid, and A. Rehman, ''Face recognition: A survey,'' J. Eng. Sci. Technol. Rev., vol. 10, no. 2, pp. 166–177, 2017.

21] Z. Song, ''Facial expression emotion recognition model integrating philosophy and machine learning theory,'' Frontiers Psychol., vol. 12, pp. 1–19, Sep. 2021.

22] J. Mahata and A. Phadikar, ''Recent advances in human behavior understanding: A survey,'' Devices Integr. Circuit., vol. 1, no. 1, pp. 751–755, Mar. 2017.

23] D. D. Sawat and R. S. Hegadi, ''Unconstrained face detection: A deep learning and machine learning combined approach,'' CSI Trans. ICT, vol. 5, no. 2, pp. 195–199, Jun. 2017.

24] M. H. Siddiqi, R. Ali, A. M. Khan, E. S. Kim, G. J. Kim, and S. Lee, ‘‘Facial expression recognition using active contour-based face detection, facial movement-based feature extraction, and non-linear feature selection,’’ Multimedia Syst., vol. 21, no. 6, pp. 541–555, Nov. 2015.

25] Y. Huang, F. Chen, S. Lv, and X. Wang, ‘‘Facial expression recognition: A survey,’’ Symmetry, vol. 11, no. 10, p. 1189, Sep. 2019.

<< Include Paper Publications under the References >>