Face Emotional Detection Using Machine Learning

[1]S. Jeniba, [2]S. Shalini, [3]A. Pettchi Shruthi, [4]K. Shameem Banu, [5]K.S. Varshini

[1]Professor, [2,3,4,5]Students
K.L.N. College of Engineering

***Abstract-* Facial expressions give important information about people emotions. Understanding facial expressions accurately is one of the challenging tasks for interpersonal relationships. Automatic emotion recognition using facial expressions is an area of interest within various fields as, for instance, computer science, healthcare, bio-engineering and psychology. In this paper we propose a method for automatic emotional recognition from facial expressions by exploiting machine learning techniques. The proposed method is able to classify a facial image under analysis in one of seven different emotions: angry, disgusted, fearful, happy, neutral, sad and surprised. Moreover, the proposed method is able to show on the facial images the areas symptomatic of a certain prediction, thus highlighting the characteristics of the face that are related to a particular emotion: in this way the model is able to explain the reason why it predicts a certain emotion.**

CHAPTER 1

INTRODUCTION

1.1General Introduction:

Human beings regularly have different moods and facial expressions changes consequently. Human emotion recognition performs a completely crucial role in social relations. The automated recognition of emotions has been an active analysis subject matter from early eras. In this deep learning system user's emotions using its facial expression can be detected. real-time detection of the face and deciphering different facial expressions like happy, sad, angry, afraid, surprise, disgust, and neutral. And many others. This system can locate six different human emotions. The trained model is capable to hit upon all of the noted emotions in real-time. An automated facial expression recognition system has to carry out detection and site of faces throughout a cluttered scene, facial feature extraction, and facial expression classification. This system has capability to monitor human beings emotions, to discriminate among emotions and label them accurately and use that emotion information to guide thinking and behaviour of specific individual.

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. There are two types of methods used for image processing namely,

- Analogue image processing
- Digital image processing

The two general phases that all types of data have to undergo while using digital technique are pre-processing, enhancement, and display, information extraction. The information obtained from Image processing is hopefully both new and useful.

Many encryption methods have been proposed in literature, and the most common way to protect large multimedia files is by using conventional encryption techniques, Private key bulk encryption algorithms, such as Triple DES, are not so suitable for transmission of images. Due to complexity of their internal structure, they are not particularly fast in terms of execution speed and cannot be applied for images in the real time scenario Also traditional cryptographic techniques such as DES cannot be applied to images due to intrinsic properties of images such as bulk data capacity, redundancy and high correlation among pixels. Image encryption algorithms can become an integral part of the image delivery process if they aim Many encryption methods have been proposed in literature, and the most common way to protect large multimedia files is by using conventional encryption techniques, Private key bulk encryption algorithms, such as Triple DES, are not so suitable for transmission of images. Due to complexity of their internal structure, they are not particularly fast in terms of execution speed and cannot be applied for images in the real time scenario Also traditional cryptographic techniques such as DES cannot be applied to images due to intrinsic properties of images such as bulk data capacity, redundancy and high correlation among pixels. Image encryption algorithms can become an integral part of the image delivery process if they aim

*1.2*Objectives:

The main objective of our project is,

- To improve their interaction with humans.
- Emotion detection can assess and measure a candidate's emotions through facial expressions.

CHAPTER 2

SYSTEM PROPOSAL

2.1 EXISTING SYSTEM:

- The classification algorithms are currently useful for the detection of emotions but the previous research models are less accurate because they usually focused on pre-processing techniques, data balancing, and various types of supervised and semi-supervised learning models.
- Therefore, it is required to find new technique with decision level fusion which would be able to integrate the accuracy of multiple deep learning algorithms with high disease detection accuracy.

Disadvantage:

- The loss value is very high when compared with proposed.
- Time consumption is high.
- Theoretical limits.

2.2 PROPOSED SYSTEM:

- In this proposed system, Machine learning is used with the help of sklearn, contains several Models. Among those models, the facial emotion of the human is classified.
- We propose a facial expression recognition method using a KNN model which extracts facial features effectively.
- Here emotions are classified as happy, sad, angry, surprise, disgusted, fearful and neutral with lots of images for the training dataset and huge images for testing.

Advantage

- It is efficient for large number of datasets.
- The experimental result is high when compared with existing system.
- Time consumption is low.
- Provide accurate prediction results.

2.3 LITERATURE SURVEY:

2.3.1 Facial emotion detection using deep learning, 2020

Author: Akriti Jaiswal, A. Krishnama Raju, Suman Deb

Methodology:

Human Emotion detection from image is one of the most powerful and challenging research task in social communication. Deep learning (DL) based emotion detection gives performance better than traditional methods with image processing. This paper presents the design of an artificial intelligence (AI) system capable of emotion detection through facial expressions. It discusses about the procedure of emotion detection, which includes basically three main steps: face detection, features extraction, and emotion classification.

Advantage:

- If it work conceivable catch any possible attack.
- Time consumption is low.

Disadvantage:

- Too Many False Negatives.
- Run to failure prediction is low.

2.3.2 Deep Facial Expression Recognition: A Survey, 2022

Author: Shan Li, Weihong Deng

Methodology:

With the transition of facial expression recognition (FER) from laboratory-controlled to challenging in-the-wild conditions and the recent success of deep learning techniques in various fields, deep neural networks have increasingly been leveraged to learn discriminative representations for automatic FER. In this survey, we provide a comprehensive review of deep FER, including datasets and algorithms that provide insights into these intrinsic problems.

Advantage:

- Monitor any data source, including user logs, devices, networks, and servers.
- Ability to predict changes.

Disadvantage:

- It can be intimidating.
- It can be intimidating.
- It can also be costly

2.3.3 Going deeper in facial expression recognition using deep neural networks, 2016

Author: David Chan, Mohammad H. Mahoor

Methodology:

Automated Facial Expression Recognition (FER) has remained a challenging and interesting problem in computer vision. Despite efforts made in developing various methods for FER, existing approaches lack generalizability when applied to unseen images or those that are captured in wild setting (i.e. the results are not significant). Most of the existing approaches are based on engineered features (e.g. HOG, LBPH, and Gabor) where the classifier's hyper-parameters are tuned to give best recognition accuracies across a single database, or a small collection of similar databases. This paper proposes a deep neural network architecture to address the FER problem across multiple well-known standard face datasets. Specifically, our network consists of two convolutional layers each followed by max pooling and then four Inception layers.

Advantage:
- Change of predicting unknown failure.
- Prediction more efficient than error value.

Disadvantage:
- Cannot use anomaly detection must be used with signature detection

2.3.4 **Hybrid deep neural networks for face emotion recognition, 2018**

Author: N Jain, S Kumar, A Kumar, P Shamsolmoali

Methodology

Deep Neural Networks (DNNs) outperform traditional models in numerous optical recognition missions containing Facial Expression Recognition (FER) which is an imperative process in next-generation Human-Machine Interaction (HMI) for clinical practice and behavioral description. Existing FER methods do not have high accuracy and are not sufficient practical in real-time applications. This work proposes a Hybrid Convolution-Recurrent Neural Network method for FER in Images. The proposed network architecture consists of Convolution layers followed by Recurrent Neural Network (RNN) which the combined model extracts the relations within facial images and by using the recurrent network the temporal dependencies which exist in the images can be considered during the classification. The proposed hybrid model is evaluated based on two public datasets and Promising experimental results have been obtained as compared to the state-of-the-art methods.

Advantage:

- Rate of missing report is low.
- Simple and Effective method.

Disadvantage:

- Needs to be trained, and trained model carefully otherwise tends to be false positive.

2.3.5 **A brief review of facial emotion recognition based on visual information, 2018**

Author: Byoung Chul Ko

Methodology:

Facial emotion recognition (FER) is an important topic in the fields of computer intelligence vision and artificial owing to its significant academic and commercial potential. Although FER can be conducted using multiple sensors, this review focuses on studies that exclusively use facial images, because visual expressions are one of the main information channels in interpersonal communication. This paper provides a brief review of researches in the field of FER conducted over the past decades. First, conventional FER approaches are described along with a summary of the representative categories of FER systems and their main algorithms. Deep-learning-based FER approaches using deep networks enabling “end-to-end” learning are then presented.

Advantage:
- Flexibility, fault tolerance, high sensing fidelity, low-cost and rapid deployment.

Disadvantage:
- Sensor nodes are prone to failures.

2.3.6 Simulations models of face-based emotion recognition, 2005

Author: AlvinI Goldman, Chandra Sekhar Sripada

Methodology: Recent studies of emotion mindreading reveal that for three emotions, fear, disgust, and anger, deficits in face-based recognition are paired with deficits in the production of the same emotion. What type of mindreading process would explain this pattern of paired deficits? The simulation approach and the theorizing approach are examined to determine their compatibility with the existing evidence. We conclude that the simulation approach offers the best explanation of the data. What computational steps might be used, however, in simulation-style emotion detection? Four alternative models are explored: a generate-and-test model, a reverse simulation model, a variant of the reverse simulation model that employs an “as if” loop, and an unmediated resonance model.

Advantage:

Simplest and Easiest Data Mining Approach.

Disadvantage:

Handling of failure prediction is difficult.

2.3.7 **State-dependent alteration in face emotion recognition in depression, 2011**
Author: Ian M. Anderson, Clare Shippen, Gabriella Juhasz
Methodology: Abnormalities in face emotion recognition differ between people with current depression and those in remission. Reduced discrimination in depressed participants may reflect withdrawal from the emotions of others, whereas the increased bias in those with a history of depression could contribute to vulnerability to relapse. The normal face emotion recognition seen in those taking medication may relate to the known effects of antidepressants on emotional processing and could contribute to their ability to protect against depressive relapse.
Advantage:

- May be more efficient.
- Flexibility.

Disadvantage:

- Accurate forecast is not possible.
- Failure implies unusual activity.

CHAPTER 3

SYSTEM DIAGRAMS

3.1 SYSTEM ARCHITECTURE:

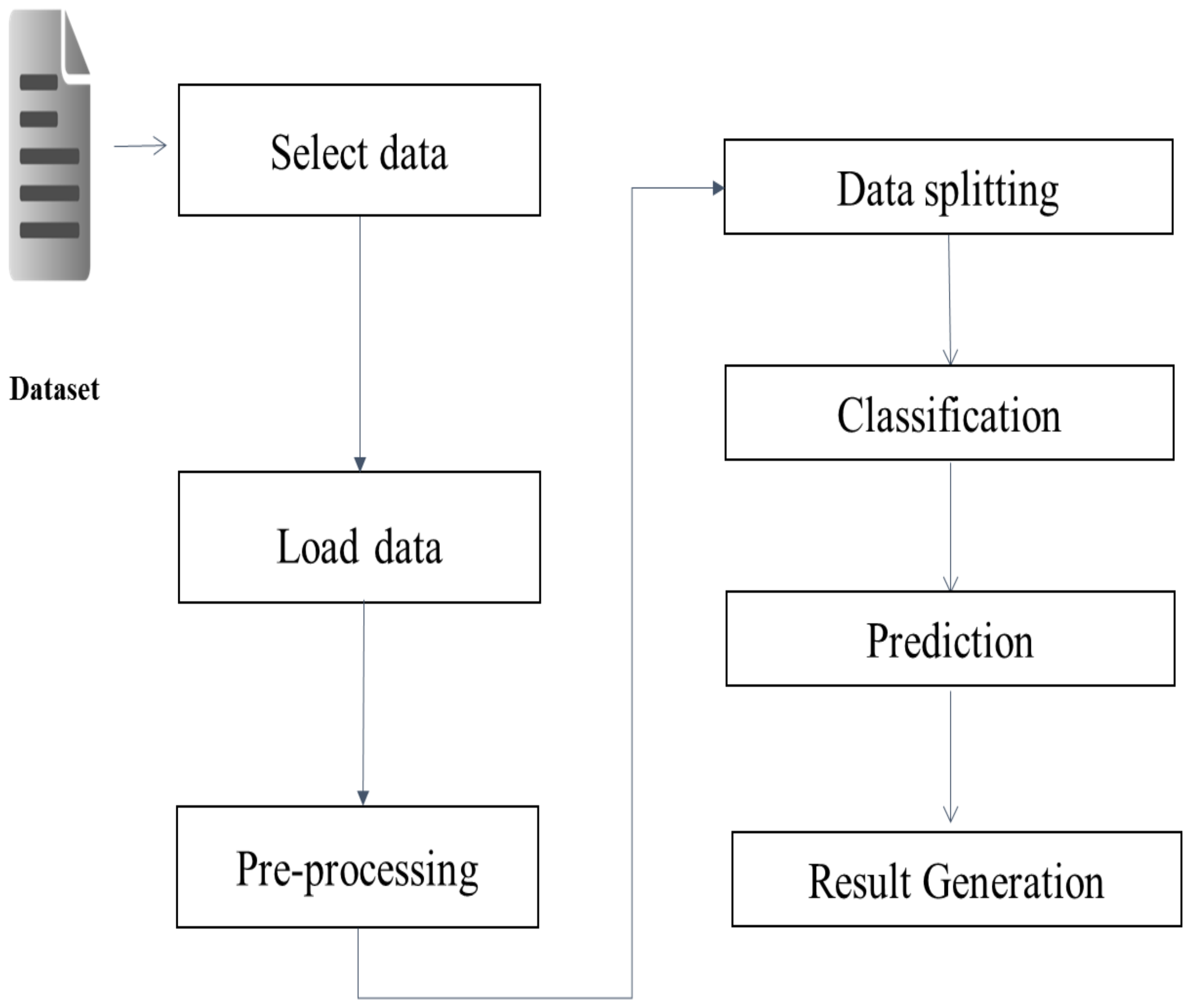


FIGURE 3.1: SYSTEM ARCHITECTURE

3.2 FLOW DIAGRAM

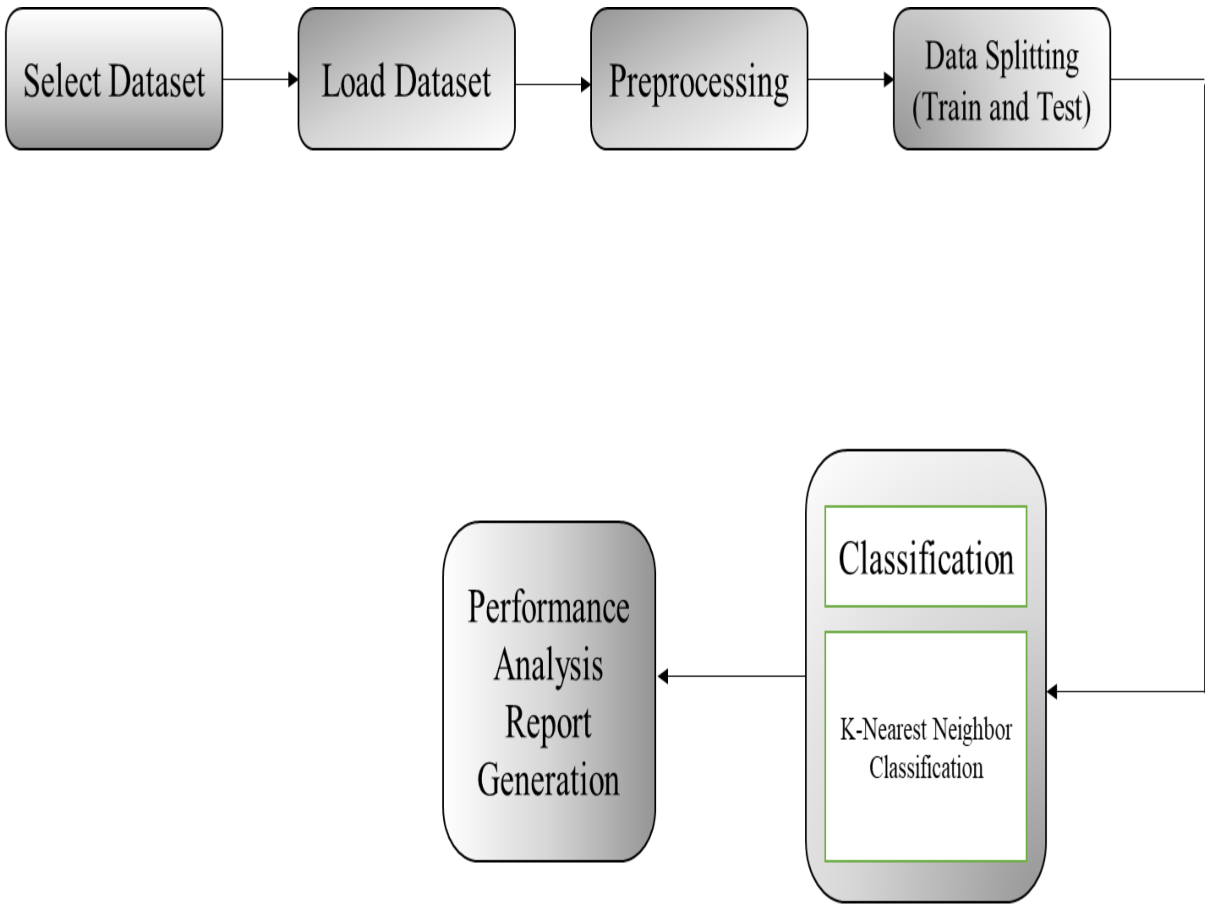

Select Dataset
Load Dataset
Preprocessing
Data Splitting (Train and Test)
Classification
K-Nearest Neighbor Classification
Performance Analysis Report Generation

FIGURE 3.2: FLOW DIAGRAM

3.3 UML DIAGRAMS:

3.3.1* USE CASE DIAGRAM*:

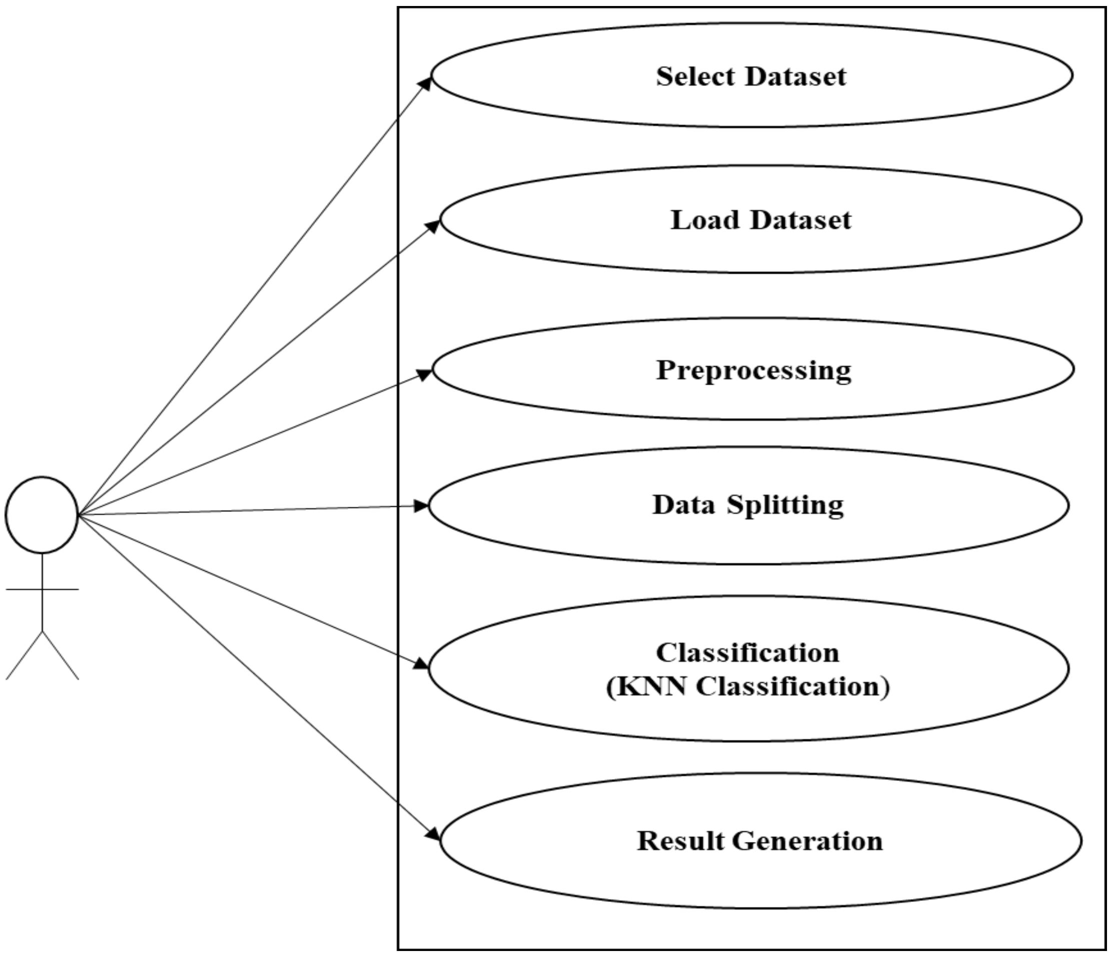


FIGURE 3.3.1: USE CASE DIAGRAM

3.3.2 CLASS DIAGRAM:

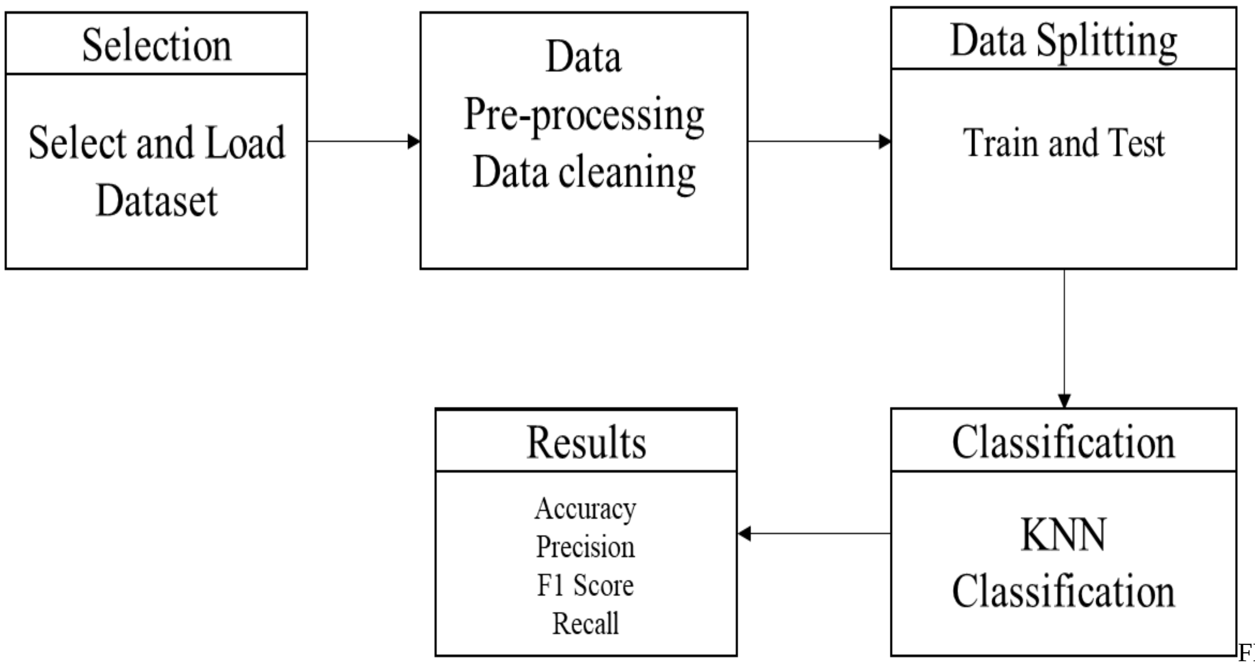


FIGURE 3.3.2: CLASS DIAGRAM

3.3.3 SEQUENCE DIAGRAM:

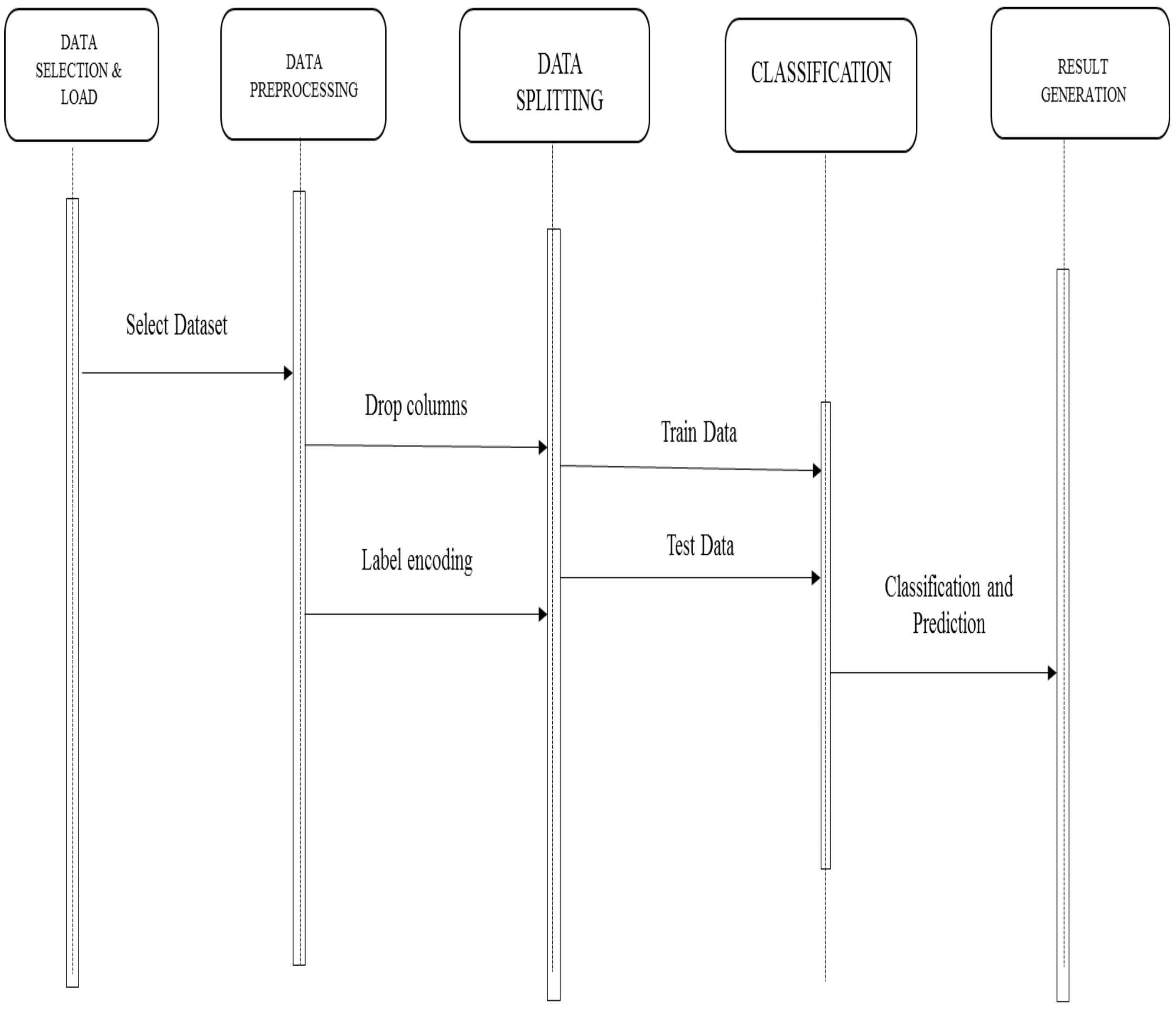


FIGURE 3.3.3: SEQUENCE DIAGRAM

3.3.4 ACTIVITY DIAGRAM:

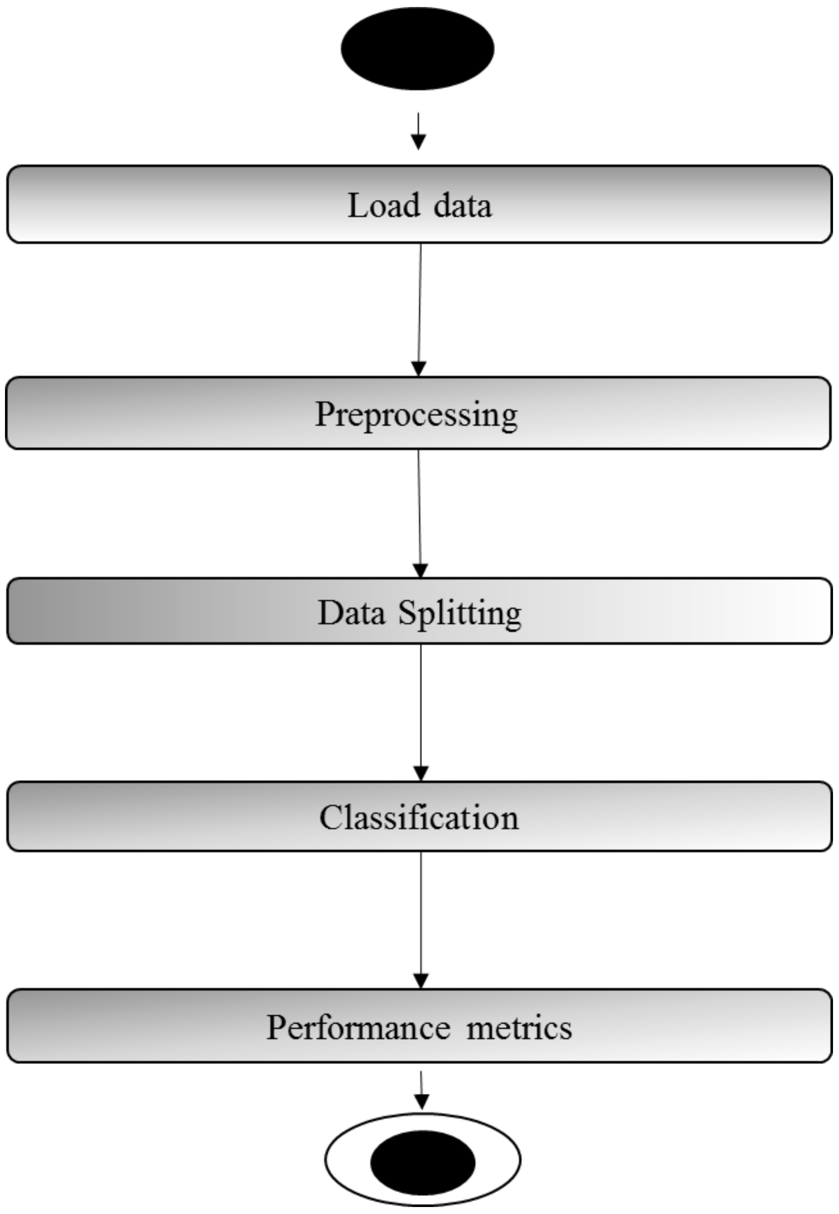


FIGURE 3.3.4: ACTIVITY DIAGRAM

3.3.4 ER DIAGRAM:

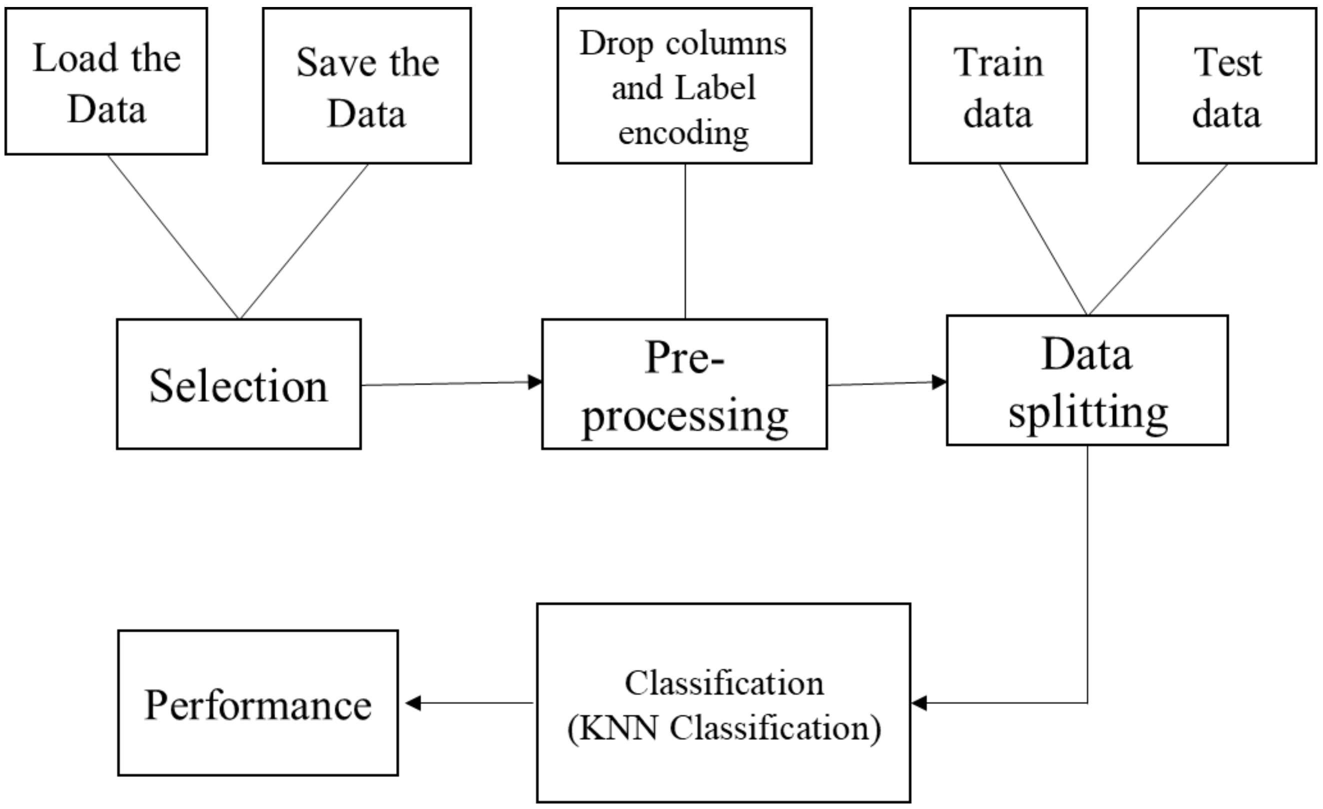


FIGURE 3.3.4: ER DIAGRAM

CHAPTER 4

IMPLEMENTATION

4.1 MODULES:

- Data selection
- Data preprocessing
- Data Splitting
- Classification
- Performance metrics

4.2 MODULES DESCRIPTION:

4.2.1 DATA SELECTION

- The input data was collected UCI repository one of the online website.
- In this work all having test dataset and train dataset in the test data set.
- In our collected dataset was read in this process.

4.2.2 DATA PREPROCESSING

Data pre-processing is the process of removing the unwanted data from the dataset. Pre-processing data transformation operations are used to transform the dataset into a structure suitable for machine learning. Missing data removal: In this process, the null values such as missing values and Nan values are replaced by 0.

4.2.3 DATA SPLITTING

During the machine learning process, data are needed so that learning can take place. In addition to the data required for training, test data are needed to evaluate the performance of the algorithm but here we have training and testing dataset separately. In our process, we have to divide as training and testing into x_train, y_train, x_test, y_test.

4.2.4 **CLASSIFICATION**

K-Nearest Neighbour

- K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

4.2.7 PERFORMANCE METRICS

Accuracy
Accuracy simply measures how often the classifier correctly predicts. We can define accuracy as the ratio of the number of correct predictions and the total number of predictions.
Precision
Precision explains how many of the correctly predicted cases actually turned out to be positive. Precision is useful in the cases where False Positive is a higher concern than False Negatives.
F1 Score
It gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.
Recall
Recall explains how many of the actual positive cases we were able to predict correctly with our model.

CHAPTER 5

SYSTEM REQUIREMENTS

5.1 HARDWARE REQUIREMENTS:

System	: Pentium IV 2.4 GHz
Hard Disk	: 200 GB
Mouse	: Logitech.
Keyboard	: 110 keys enhanced
Ram	: 4GB

5.2* SOFTWARE REQUIREMENTS*:

O/S	: Windows 7.
Language	: Python
Front End	: Anaconda Navigator – Spyder

5.3 SOFTWARE DESCRIPTION:

5.3.1 Python
Python is one of those rare languages which can claim to be both *simple* and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. I will discuss most of these features in more detail in the next section.

***5.3.2* Features of Python**

- **Simple**

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

- **Easy to Learn**

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

- **Free and Open Source**

Python is an example of a *FLOSS* (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

- **High-level Language**

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

- **Portable**

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like Kivy to create games for your computer *and* for iPhone, iPad, and Android.

- **Interpreted**

This requires a bit of explanation.

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just *run* the program directly from the source code. Internally, Python converts the source code into an intermediate form called byte codes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

- **Object Oriented**

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In *object*-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

- **Extensible**

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

- **Embeddable**

You can embed Python within your C/C++ programs to give *scripting* capabilities for your program's users.

- **Extensive Libraries**

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the Batteries Included philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

5.4 TESTING PRODUCTS:

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. . A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that "al gears mesh", that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

*5.4.*1 UNIT TESTING:

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as 'module testing'. The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

*5.4.2*INTEGRATION TESTING:

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

i) Top-down integration testing. ii) Bottom-up integration testing.

5.4. TESTING TECHNIQUES/STRATEGIES:

- ***BLACK BOX TESTING:***

1. Black box testing is done to find incorrect or missing function
2. Interface error
3. Errors in external database access
4. Performance errors.

5. Initialization and termination errors

In 'functional testing', is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called 'black box testing'. It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

- ***WHITEBOX TESTING:***

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we Derived test cases that guarantee that all independent paths within a module have been exercised at least once.

5.4.4 SOFTWARE TESTING STRATEGIES

VALIDATION TESTING:

- After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many,
- But a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer

USER ACCEPTANCE TESTING:

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

OUTPUT TESTING*:*

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

CHAPTER 6

CONCLUSION

In this process, we propose a facial expression recognition method using a KNN model which extracts facial features effectively. Compared to traditional methods, the proposed method can automatically learn pattern features and reduce the incompleteness caused by artificial design features. This project achieves the desired output that is it detects face from the given input within a boundary and it detects the emotion of face and displayed the detected output as happy, sad, neutral, angry and surprise with good accuracy.

CHAPTER 7

FUTURE ENHANCEMENT

Having examined techniques to cope with expression variation, in future it may be investigated in more depth about the face classification problem and optimal fusion of colour and depth information. Further study can be laid down in the direction of allele of gene matching to the geometric factors of the facial expressions. The genetic property evolution framework for facial expressional system can be studied to suit the requirement of different security models such as criminal detection, governmental confidential security breaches etc.

CHAPTER 8

SAMPLE CODING

```
import warnings
warnings.filterwarnings("ignore")
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
import random
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import Convolution2D, MaxPooling2D
from keras.utils import np_utils
from numpy import argmax
from sklearn.metrics import confusion_matrix, classification_report
from mlxtend.plotting import plot_confusion_matrix

print()

print("<-------------- INPUT --------------->")

print()

path=r'dataset/'


# let's display some of the pictures
for category in categories:
    fig, _ = plt.subplots(3,4)
    fig.suptitle(category)

    for k, v in enumerate(os.listdir(path+category)[:12]):
        img = cv2.imread(path+category+'/'+v)
        plt.subplot(3, 4, k+1)
        plt.axis('off')
        plt.imshow(img)
    plt.show()

shape0 = []
shape1 = []

for category in categories:
    for files in os.listdir(path+category):
        shape0.append(cv2.imread(path+category+'/'+ files).shape[0])
        shape1.append(cv2.imread(path+category+'/'+ files).shape[1])
    print(category, ' => height min : ', min(shape0), 'width min : ', min(shape1))
    print(category, ' => height max : ', max(shape0), 'width max : ', max(shape1))
    shape0 = []
    shape1 = []

print()

print("<------------- Data Preprocessing -------------->")

print()

# initialize the data and labels
data = []
labels = []
imagePaths = []
HEIGHT = 32
WIDTH = 32
```

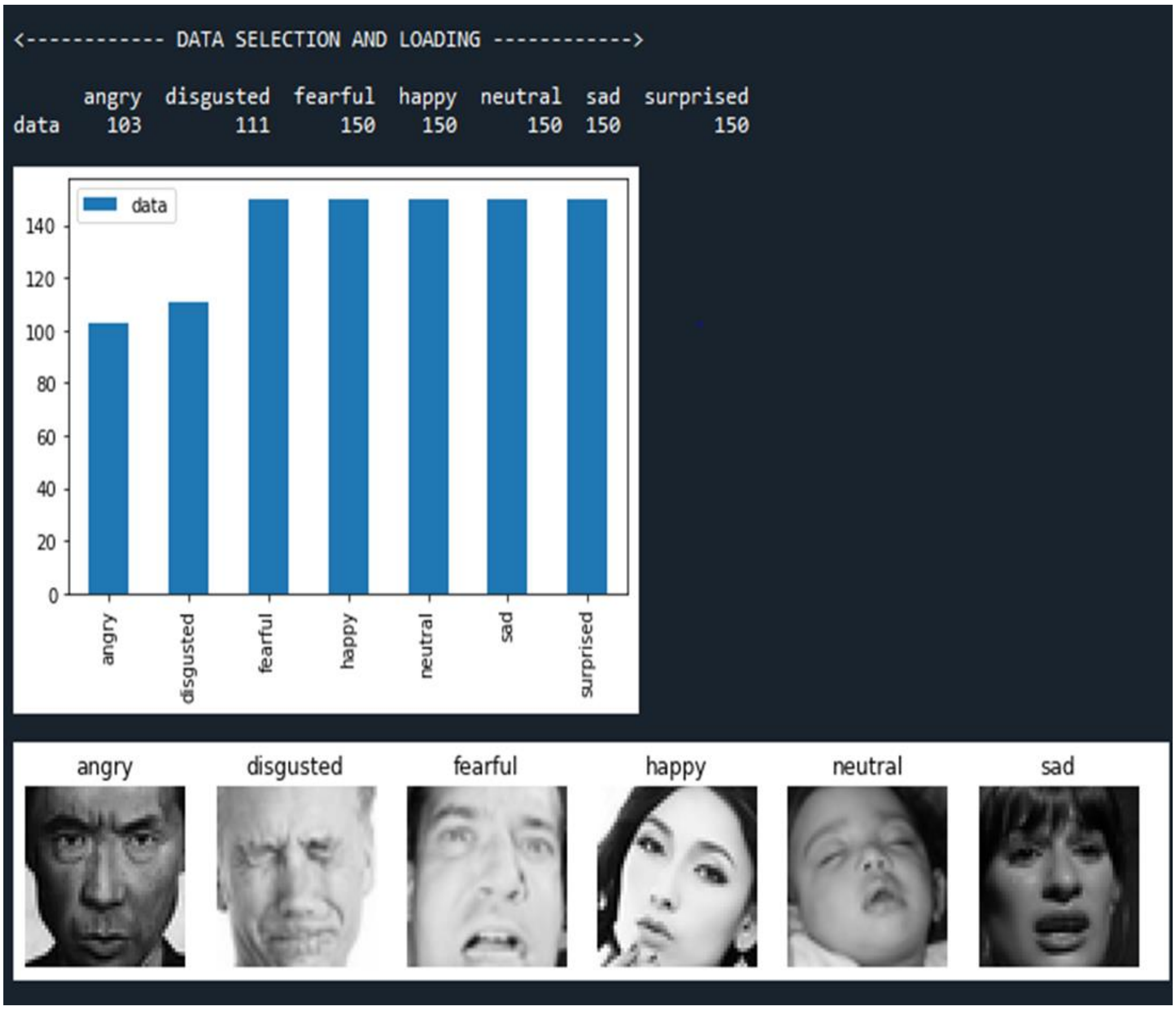
```
N_CHANNELS = 3

# grab the image paths and randomly shuffle them
for k, category in enumerate(categories):
    for f in os.listdir(path+category):
        imagePaths.append([path+category+'/'+f, k])

random.shuffle(imagePaths)
print(imagePaths[:10])

# loop over the input images
for imagePath in imagePaths:
# load the image, resize the image to be HEIGHT * WIDTH pixels (ignoring aspect ratio) and store the image in the data list
    image = cv2.imread(imagePath[0])
    image = cv2.resize(image, (WIDTH, HEIGHT))
    data.append(image)

    # extract the class label from the image path and update the
    # labels list
    label = imagePath[1]
    labels.append(label)

# scale the raw pixel intensities to the range [0, 1]
data = np.array(data, dtype="float") / 255.0
labels = np.array(labels)

# Let's check everything is ok
fig, _ = plt.subplots(4,5)
fig.suptitle("Sample Input")
fig.patch.set_facecolor('xkcd:white')
for i in range(20):
    plt.subplot(4,5, i+1)
    plt.imshow(data[i])
    plt.axis('off')
#   plt.title(categories[labels[i]])
plt.show()

print()

print("<----------------- Data Splitting ------------------->")

print()

(trainX, testX, trainY, testY) = train_test_split(data, labels, test_size=0.4, random_state=42)
# Preprocess class labels
trainY = np_utils.to_categorical(trainY, 2)

print(trainX.shape)
print(testX.shape)
print(trainY.shape)
print(testY.shape)
print()


import matplotlib.pyplot as plt
import numpy as np
import cv2
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
# import sklearn.metrics as metrics
from sklearn.neighbors import KNeighborsClassifier

knn=KNeighborsClassifier(n_neighbors=7)
knn.fit(x_train2,trainY)
y_pred_knn=knn.predict(x_test2)
print(y_pred_knn)

print(classification_report(y_pred_knn,testY))

knn=confusion_matrix(y_pred_knn,testY)

print(knn)

TP=knn[6][4]
TN=knn[2][6]
FP=knn[0][1]
FN=knn[5][3]
Total=TP+TN+FP+FN
acc_knn=(TP+TN)/Total
pre_knn=TP/(TP+FP)
rec_knn=TP/(TP+FN)
f1_knn=2*(pre_knn*rec_knn)/(pre_knn+rec_knn)
Accuracy=acc_knn*100
print()
print("Accuracy of KNN =",Accuracy,"%" )
print()
print("Precision=",pre_knn*100)
print()
print("Recall=",rec_knn*100)
print()
print("F1 Score=",f1_knn*100)

print
print("<------------------- PREDICTION -------------------->")

print()

from tkinter import filedialog
test_data=[]
Image = filedialog.askopenfilename()
```

```
head_tail = os.path.split(Image)
fileNo=head_tail[1].split('.')
test_image_o = cv2.imread(head_tail[0]+'/'+fileNo[0]+'.jpg')
test_image = cv2.resize(test_image_o, (WIDTH, HEIGHT))
test_data.append(test_image)

# scale the raw pixel intensities to the range [0, 1]
test_data = np.array(test_image, dtype="float") / 255.0
test_data=test_data.reshape([-1,32, 32, 3])
pred = model.predict(test_data)
predictions = argmax(pred, axis=1) # return to label
print ('Prediction : '+categories[predictions[0]])

fig = plt.figure()
fig.patch.set_facecolor('xkcd:white')
plt.title(categories[predictions[0]])
plt.imshow(test_image_o)
```

CHAPTER 9

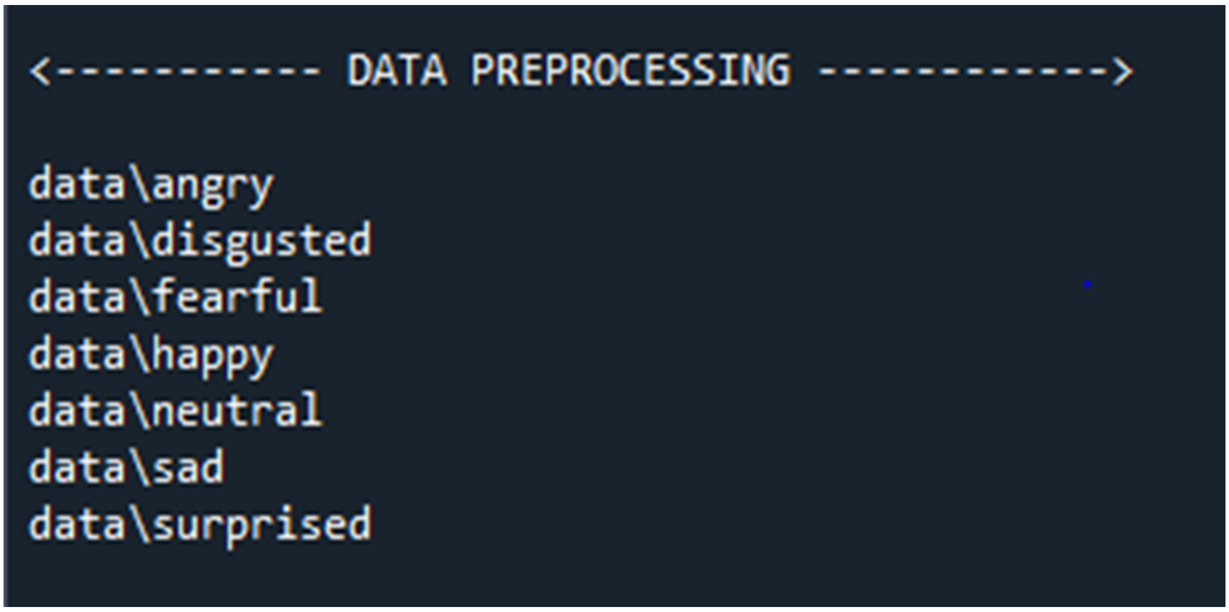
SAMPLE SCREENSHOTS

Data selection:

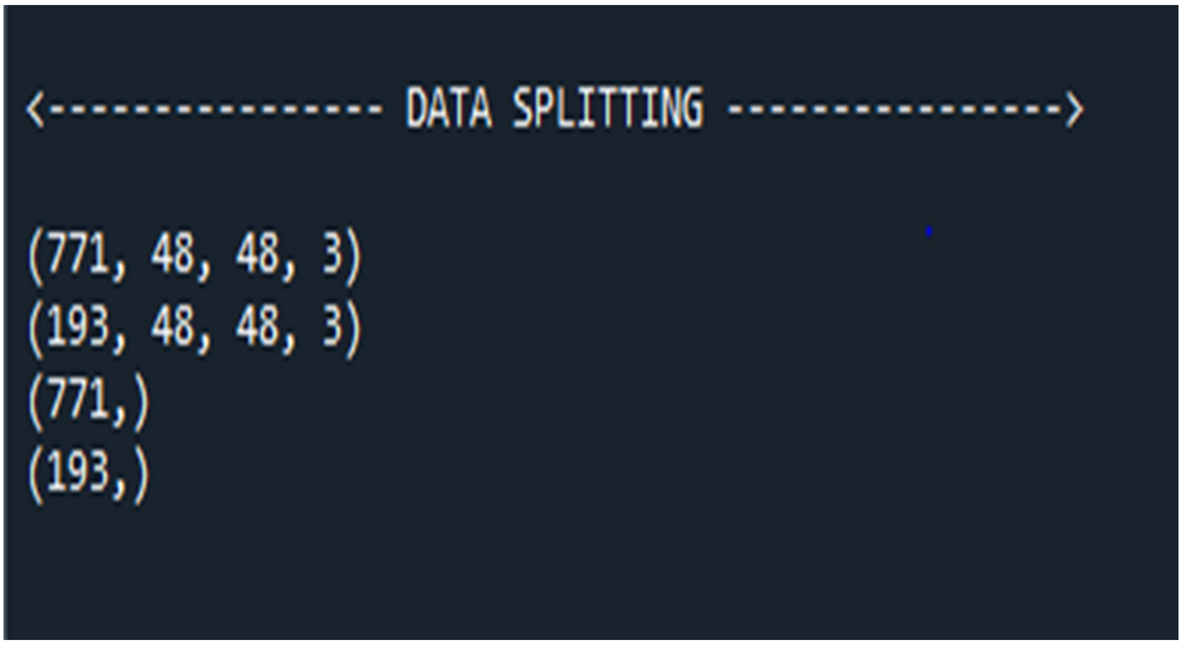


```
<------------ DATA SELECTION AND LOADING ------------>

      angry  disgusted  fearful  happy  neutral  sad  surprised
data    103        111      150    150      150  150        150
```

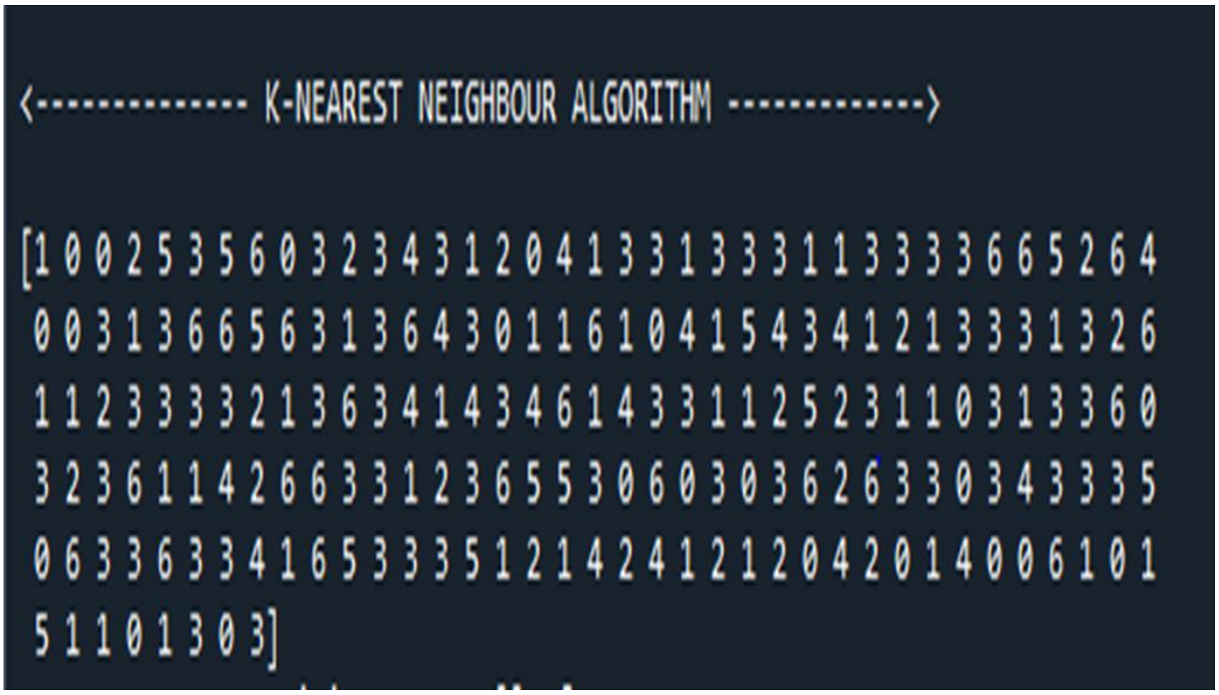
Data pre-processing:


```
<----------- DATA PREPROCESSING ------------>

data\angry
data\disgusted
data\fearful
data\happy
data\neutral
data\sad
data\surprised
```

Data splitting:

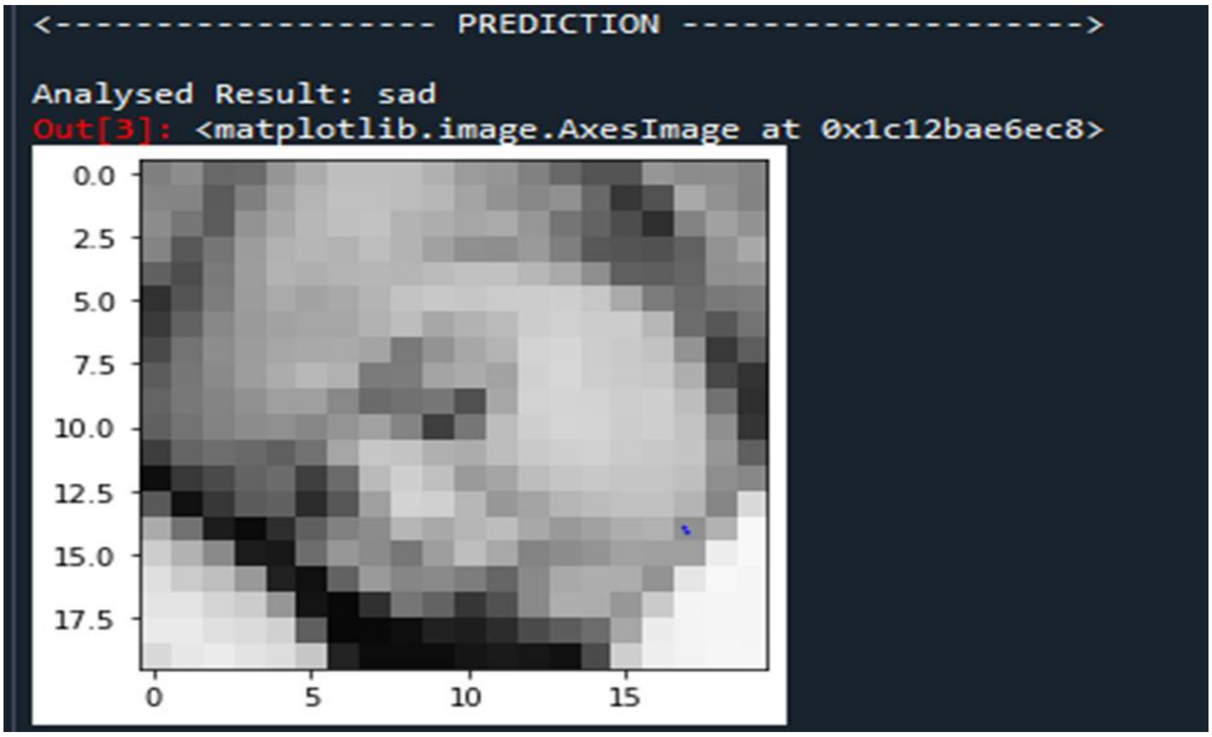

```
<--------------- DATA SPLITTING --------------->

(771, 48, 48, 3)
(193, 48, 48, 3)
(771,)
(193,)
```

Classification:



```
<-------------- K-NEAREST NEIGHBOUR ALGORITHM ------------->

[1 0 0 2 5 3 5 6 0 3 2 3 4 3 1 2 0 4 1 3 3 1 3 3 3 1 1 3 3 3 6 6 5 2 6 4
 0 0 3 1 3 6 6 5 6 3 1 3 6 4 3 0 1 1 6 1 0 4 1 5 4 3 4 1 2 1 3 3 3 1 3 2 6
 1 1 2 3 3 3 3 2 1 3 6 3 4 1 4 3 4 6 1 4 3 3 1 1 2 5 2 3 1 1 0 3 1 3 3 6 0
 3 2 3 6 1 1 4 2 6 6 3 3 1 2 3 6 5 5 3 0 6 0 3 0 3 6 2 6 3 3 0 3 4 3 3 3 5
 0 6 3 3 6 3 3 4 1 6 5 3 3 3 5 1 2 1 4 2 4 1 2 1 2 0 4 2 0 1 4 0 0 6 1 0 1
 5 1 1 0 1 3 0 3]
```

Prediction:



```
<-------------------- PREDICTION -------------------->

Analysed Result: sad
Out[3]: <matplotlib.image.AxesImage at 0x1c12bae6ec8>
```

REFERENCES:

1. Gangeh MJ, Sørensen L, Shaker SB, et al. A texton-based approach for the classification of lung parenchyma in CT images. In: International conference on medical image computing and computer-assisted intervention. Berlin, Heidelberg: Springer; 2010.
2. Park SC, Tan J, Wang X, et al. Computer aided detection of early interstitial lung diseases using low-dose CT images. Phys Med Biol
3. Koratis PD, Karahaliou AN, Kazantzi AD, et al. Texture-based identication and characterization of interstitial pneumonia patterns in lung multidetector ct. IEEE Trans Inform Technol Biomed 2010
4. El-Baz A, Soliman A, McClure P, et al. Early assessment of malignant lung nodules based on the spatial analysis of detected lung nodules. In: 2012 9th IEEE international symposium on biomedical imaging (ISBI). Piscataway: IEEE; 2012.

5. Jacobs C, Sanchez CI, Saur SC, et al. Computer aided detection of ground glass nodules in thoracic ct images using shape, intensity and context features. In: International conference on medical image computing and computer-assisted intervention Berlin, Heidelberg: Springer; 2011.
6. Depeursinge A, Foncubierta-Rodriguez A, Van de Ville D, et al Lung texture classification using locally-oriented riesz components. Med Image Comput Comput Assist Interv2011.
7. Depeursinge A1, Van de Ville D, Platon A, et al. Near-affine-invariant texture learning for lung tissue analysis using isotropic wavelet frames IEEE Trans Inf Technol Biomed 2012 Jul.
8. Song Y, Cai W, Kim J, et al. A multistage discriminative model for tumor and lymph node detection in thoracic images. IEEE Trans Med Imaging 2012 May.
9. Xu R, Hirano Y, Tachibana R, et al. Classification of diffuse lung disease patterns on high-resolution computed tomography by a bag of words approach. Med Image Comput Comput Assist Interv2011.