

**Detection of forged face from images
using ELA and MTCNN**

A project submitted to

MALLA REDDY UNIVERSITY

In partial fulfillment of the requirements for the award degree of

**BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE & ENGINEERING (AI & ML)**

Submitted by

S SAMPATH KUMAR: (2011CS020471)

Under the guidance of

Prof. H. PACKIARAJ

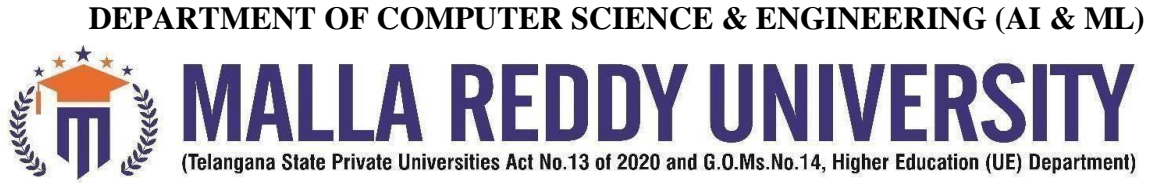
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (AI & ML)



MALLA REDDY UNIVERSITY

(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

2023- 2024



2022-2023

COLLEGE CERTIFICATE

This is to certify that this is the bonafide record of the application development entitled,
“ Detection of forged face from images using ELA and MTCNN” Submitted by S. Sampath Kumar
(2011CS0204671) of B. Tech IV'th Year I Semester, Department of CSE-AIML during the year 2023- 2024.
The results embodied in this report have not been submitted to any other University or Institute for the award of
any degree or diploma.

Internal Guide
(Prof. H. Packiaraj)

Head Of The Department
(Dr. Thayyaba Khatoon)

External Examiner

ACKNOWLEDGEMENT

We feel ourselves honoured and privileged to place our warm salutation to our college Malla Reddy University and which gave us the opportunity to have experience in engineering and profound technical knowledge. We express our heartiest thanks to my Head of the Department **Dr. Thayyaba Khatoon** for encouraging us in every aspect of our project and helping us realize our full potential. We would like to thank our internal guide **Prof. H. Packiaraj** for their regular guidance and constant encouragement. We are extremely grateful to their valuable suggestions and co-operation throughout project work. We would also like to thank all the supporting staff of the Department of **CSE-(AIML)** in making our project success.

ABSTRACT

Many forged face images are spreading through digital media nowadays. Our aim is to make a real and forged facial image recognition using Deep learning. Identifying such forged face images are promising research areas in this digital era. To make a forged face detector we must have decent amount of data (Data Set). We are using 10,000 images of both the classes i.e. real and fake class. These images are then preprocessed using Error Level Analysis (ELA). For the detection of forged faces We are going to identify the faces in an image. There are multiple techniques available for identifying the face in an image but the preferred one among all are, deep learning based Multi-task Cascade CNN(MTCNN). Finally, we will create a detector that classify the faces as real or fake.

CONTENTS

CHAPTER NUMBER	CHAPTER NAME	PAGE NUMBERS
1	Introduction	6-7
2	Literature Survey	8
3	Proposed Methodology	9-10
4	Results	11-13
5	Codes	14-16
6	Conclusion	17
7	References	18

CHAPTER 1

INTRODUCTION

1.1 Problem Definition

- This project aims to create a forged face detector in which we are going to identify the face in an image and then detects the face as real or fake.
- We believe what we see right? But with the advancement of science and technology we have come to see a lot of new things which we as a regular person didn't actually see.
- let me ask you a question:



- Can you look at the picture and tell if it is real or not?
- You may be thinking it is real but what if I tell you this is a fake image generated using GANs (Generative Adversarial Networks) one of the growing field in Machine Learning/Artificial Intelligence.
- We are going to identify the faces in an image. There are multiple techniques available for identifying the face in an image but the preferred one among all are, deep learning based Multi-task Cascade CNN(MTCNN).
- We will create a detector that classify the faces as real or fake.

1.2 Objective of Project

- There are many websites that are used as tools for creating fake face images.
- Detection of such fake images with our naked eyes are impossible.
- Improvements in artificial intelligence (A.I.), mean that it is easier to make very realistic fake pictures, video, and sound recordings.
- All of these things make it harder to believe what we see online.
- Looking at all the possibilities that one can use these fake face images, we are motivated to create forged face detector which helps us to differentiate between real and fake faces.

1.2 Limitations of Project

Every project has its constraints and potential drawbacks. Acknowledging these limitations is crucial for understanding the scope and potential areas for improvement.

Potential Limitations:

- **Dataset Bias:** The performance of the model heavily relies on the diversity and representativeness of the dataset. If the dataset is biased or not comprehensive, the model might not generalize well.
- **Computational Resources:** The complexity of deep learning models might pose challenges in terms of computational requirements, especially for real-time applications or resource-constrained environments.
- **Adversarial Attacks:** Deep learning models, including face detectors, are susceptible to adversarial attacks where slight modifications to input data could lead to incorrect predictions.

CHAPTER 2

LITERATURE SURVEY

LITERATURE SURVEY

In this chapter, we will take a brief look into some papers based on CNN. We will analyse the projects, take note of their purpose, implementation, advantages and limitations. Finally, we will discuss how we have understood and overcame those limitations in our project.

2.1 Paper 1: Vehicle Detection and License Plate Recognition using Deep Learning.

In this paper the author has trained the YOLO model on 200 images and validated the same using 8000 images, which is a blunt mistake. The author has later corrected himself by training on 400 images and validate on 100 images. But the author has used the first weights itself for later in the project due to performance issues and time constraints. The ideal mixture of images is about training on 2000 images and validate on 1000 images according to official darknet documentation.

2.2 Paper 2: An Efficient Scheme for Face Detection.

This work is based on skin color and feature extraction to provide an efficient and simple way to detect human faces in images. The features under consideration are mouth, eyes, and nose. The results are with good accuracy, great speed and simple computations.

2.3 Paper 3: When Face Recognition Meets with Deep Learning.

The paper aims to provide a common ground to all students and researchers alike by conducting an evaluation of easily reproducible face recognition systems based on CNNs. It uses public database LFW (Labelled Faces in the Wild) to train CNNs instead of a personal database. It proposes three CNN architectures which are the first reported architectures trained using LFW data.

CHAPTER 3

PROPOSED METHODOLOGY

3.1 EXISTING SYSTEM

The existing system refers to the state-of-the-art or commonly used methods for face detection and forgery detection.

Common Existing Techniques:

- **Haarcascades:** This is a traditional face detection method based on machine learning where a cascade function is trained from a lot of positive and negative images.
- **Feature Extraction:** Some methods rely on extracting facial features like eyes, nose, and mouth based on color information.
- **Simple CNN Architectures:** Earlier face detection models may have used simpler Convolutional Neural Networks (CNNs) without the complexity of modern architectures.

Limitations of Existing Methods:

- **Accuracy:** Traditional methods may not be as accurate, especially in the presence of variations in lighting, pose, and facial expressions.
- **Forgery Detection:** Many existing methods might not be adept at distinguishing between real and forged faces, especially with the rise of deepfake technology.

3.2 PROPOSED SYSTEM

The proposed system in our novel approach to addressing the limitations of existing methods. In our case, it involves using the Multi-task Cascade CNN (MTCNN) for face detection and subsequently training a more sophisticated model for forged face classification.

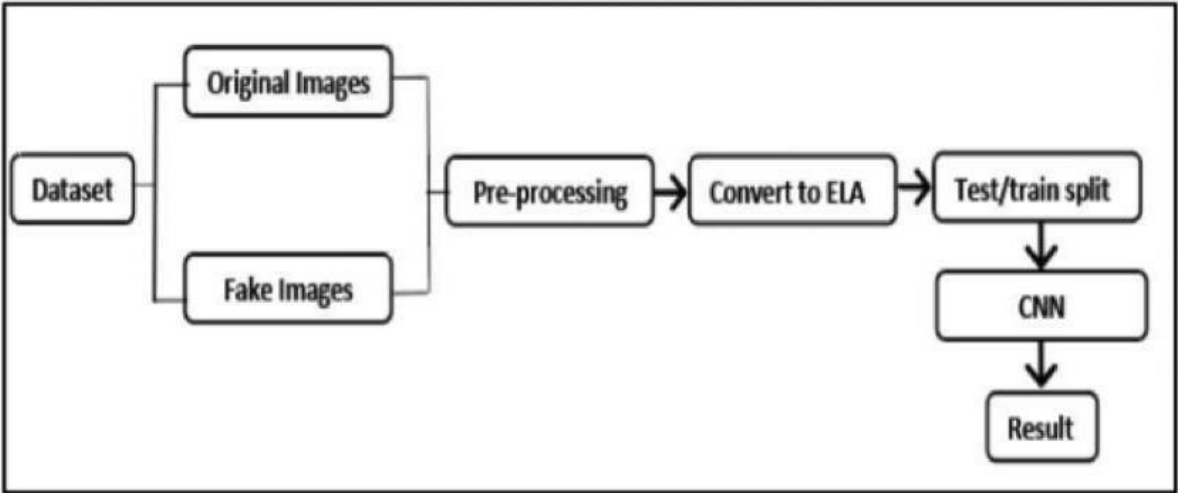
Key Components of the Proposed System:

- **MTCNN for Face Detection:** MTCNN is chosen for its ability to efficiently detect faces in images. It consists of multiple stages for progressively refining the face region.
- **CNN for Classification:** A Convolutional Neural Network is employed for classifying the detected faces as real or fake. The architecture includes convolution layers, max-pooling layers, flattening layer, and an output layer with sigmoid activation.

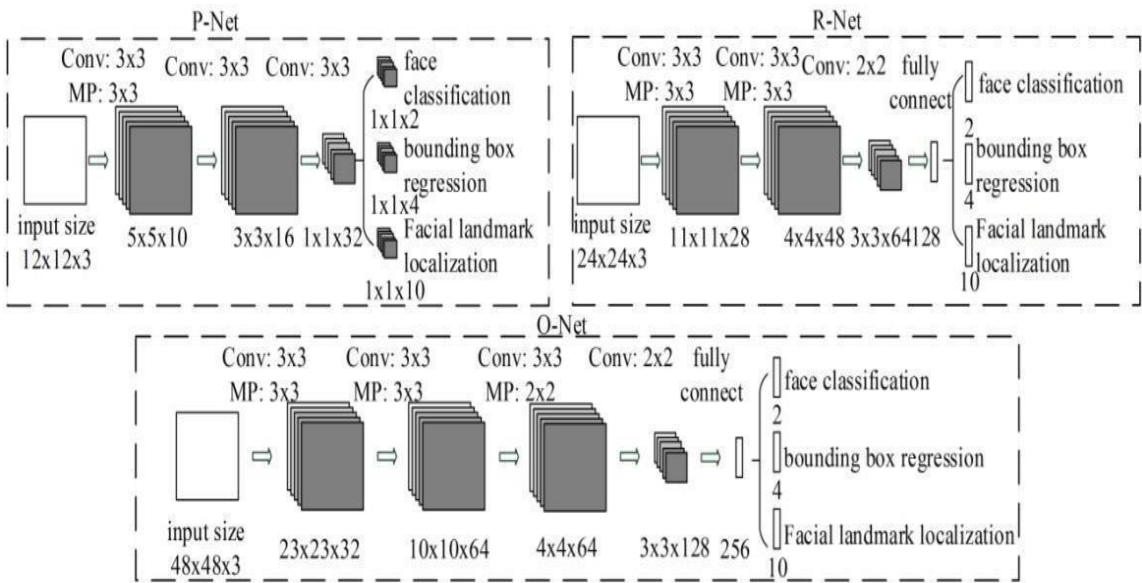
Advantages of the Proposed System:

- **Accuracy:** Leveraging deep learning techniques typically results in higher accuracy in face detection and forgery classification tasks.
- **Robustness:** MTCNN, being a multi-task approach, enhances the robustness of face detection.

3.3 Flow Chart



3.4 UML DIAGRAM



CHAPTER 4

RESULTS

Results after training the model:

- We got 98.3% accuracy in just 8 epochs while training the model.

```
Epoch 1/50
234/234 [=====] - 1670s 7s/step - loss: 0.1179 - acc: 0.9565 - val_loss: 0.0865 - val_acc: 0.9667
Epoch 2/50
234/234 [=====] - 1641s 7s/step - loss: 0.1962 - acc: 0.9269 - val_loss: 0.1060 - val_acc: 0.9649
Epoch 3/50
234/234 [=====] - 1737s 7s/step - loss: 0.1331 - acc: 0.9486 - val_loss: 0.1009 - val_acc: 0.9633
Epoch 4/50
234/234 [=====] - 1673s 7s/step - loss: 0.0919 - acc: 0.9635 - val_loss: 0.1043 - val_acc: 0.9637
Epoch 5/50
234/234 [=====] - 76306s 326s/step - loss: 0.0757 - acc: 0.9701 - val_loss: 0.0771 - val_acc: 0.9659
Epoch 6/50
234/234 [=====] - 1669s 7s/step - loss: 0.0620 - acc: 0.9760 - val_loss: 0.0589 - val_acc: 0.9766
Epoch 7/50
234/234 [=====] - 1658s 7s/step - loss: 0.0612 - acc: 0.9751 - val_loss: 0.0517 - val_acc: 0.9804
Epoch 8/50
234/234 [=====] - 1659s 7s/step - loss: 0.0526 - acc: 0.9787 - val_loss: 0.0448 - val_acc: 0.9837
```

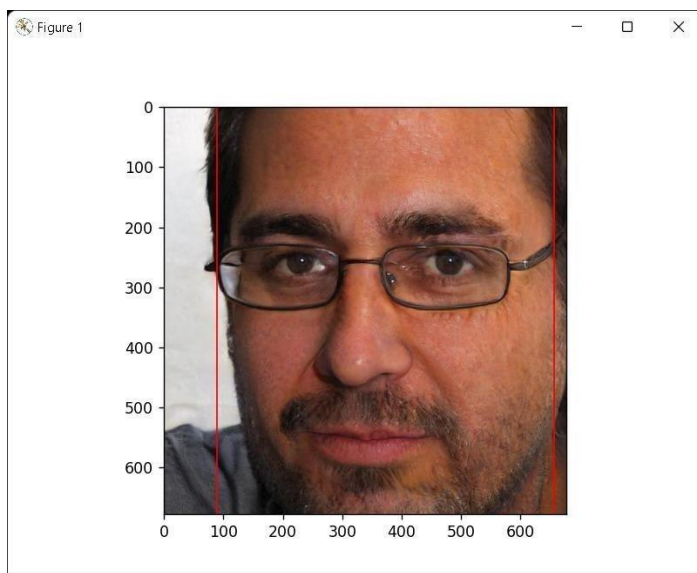
- Now we will test our model on different fake and real images.

Input and Output:

- Sample input of fake face.

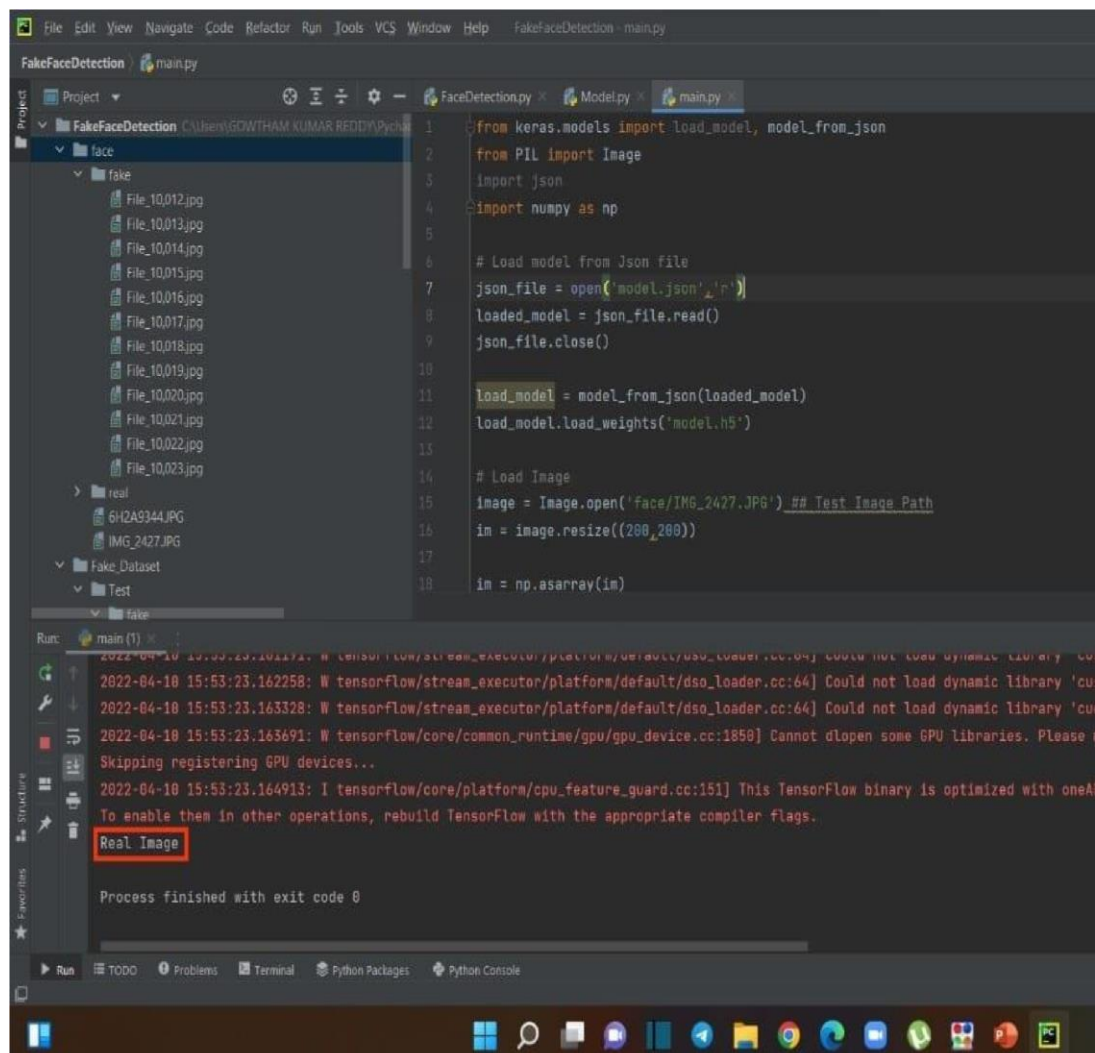


- Sample output of face detection using MTCNN.



- Output after classifying the face as real or fake.

Classification Report:



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help FakeFaceDetection - main.py
FakeFaceDetection main.py
Project
  FakeFaceDetection C:\Users\GOWTHAM KUMAR REDDY\PyCharm
    face
      File_10,012.jpg
      File_10,013.jpg
      File_10,014.jpg
      File_10,015.jpg
      File_10,016.jpg
      File_10,017.jpg
      File_10,018.jpg
      File_10,019.jpg
      File_10,020.jpg
      File_10,021.jpg
      File_10,022.jpg
      File_10,023.jpg
    real
      6H2A9344.JPG
      IMG_2427.JPG
    Fake_Dataset
    Test
    fake
Run
  main (1)
  2022-04-10 15:53:23.101172: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cu
  2022-04-10 15:53:23.162258: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cu
  2022-04-10 15:53:23.163328: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cu
  2022-04-10 15:53:23.163691: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1858] Cannot dlopen some GPU libraries. Please
  Skipping registering GPU devices...
  2022-04-10 15:53:23.164913: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneA
  To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
  Real Image
  Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
```

CHAPTER 5

CODES

5.1 Code for face detection using MTCNN

```
FaceDetection.py x Model.py x main.py x
1 from matplotlib import pyplot
2 from matplotlib.patches import Rectangle
3 from mtcnn.mtcnn import MTCNN
4 from PIL import Image
5
6 # load image from folder
7 filename = 'face/fake/File_10,012.jpg'
8
9
10 def draw_image_with_boxes(filename, result_list):
11     # load the image
12     data = pyplot.imread(filename)
13     # plot the image
14     pyplot.imshow(data)
15     # get the context for drawing boxes
16     ax = pyplot.gca()
17     # plot each box
18     for result in result_list:
19         # get coordinates
20         x, y, width, height = result['box']
21         # create the shape
22         rect = Rectangle((x, y), width, height, fill=False, color='red')
23         # draw the box
24         ax.add_patch(rect)
25     # show the plot
26     pyplot.show()
27     return (x, y, x + width, y + height)
28
29
30 detector = MTCNN()
31 # detect faces in the image
32 pixels = pyplot.imread(filename)
33 faces = detector.detect_faces(pixels)
34 # display faces on the original image
35 draw_image_with_boxes(filename, faces)
```

5.2 Code to train the model.

```
FaceDetection.py x Model.py x main.py x
1 from keras.models import Sequential
2 from keras.layers import Conv2D,MaxPool2D,Flatten,Dense,Dropout
3 from keras.preprocessing.image import ImageDataGenerator
4 from tensorflow.keras.optimizers import Adam
5 from tensorflow.keras.utils import to_categorical
6 model = Sequential()
7
8 model.add(Conv2D(32,(3,3),input_shape=(200,200,3),activation='relu',padding='same'))
9 model.add(MaxPool2D(pool_size=(2,2)))
10 model.add(Dropout(0.2))
11
12 model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
13 model.add(MaxPool2D((2, 2)))
14 model.add(Dropout(0.2))
15
16 model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
17 model.add(MaxPool2D((2, 2)))
18 model.add(Dropout(0.2))
19
20 model.add(Flatten())
21
22 model.add(Dense(128, activation='relu'))
23 model.add(Dropout(0.5))
24 model.add(Dense(1, activation='sigmoid'))
25
26 model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])
27
28 datagen = ImageDataGenerator(rescale=1.0/255.0)
29
30 train = datagen.flow_from_directory('Fake_Dataset/Train/,'
```

```
31         class_mode='binary',
32         batch_size=64,
33         target_size=(200,200))
34
35 test = datagen.flow_from_directory('Fake_Dataset/Test/',
36         class_mode='binary',
37         batch_size=64,
38         target_size=(200,200))
39
40 history = model.fit_generator(train,
41         validation_data=(test),
42         epochs=20,
43         steps_per_epoch=len(train),
44         validation_steps=len(test))
45
46 model_json = model.to_json()
47
48 with open('model.json','w') as json_file:
49     json_file.write(model_json)
50
51 model.save_weights('model.h5')
```

5.3 Code for classifying the face as real or fake.

```
FaceDetection.py x Model.py x main.py x
1 from keras.models import load_model, model_from_json
2 from PIL import Image
3 import json
4 import numpy as np
5
6 # Load model from Json file
7 json_file = open('model.json','r')
8 loaded_model = json_file.read()
9 json_file.close()
10
11 load_model = model_from_json(loaded_model)
12 load_model.load_weights('model.h5')
13
14 # Load Image
15 image = Image.open('face/IMG_2427.JPG') ## Test Image Path
16 im = image.resize((200,200))
17
18 im = np.asarray(im)
19 im = np.reshape(im,(1,im.shape[0],im.shape[1],im.shape[2]))
20
21 # Make Prediction
22 prediction = load_model.predict(im)
23 if prediction == 1:
24     print('Real Image')
25 else:
26     print('Fake Image')
```


CHAPTER 6

CONCLUSION

In summary, we used MTCNN for face detection in an image. We used 10,000 images of both the classes i.e. real and fake class as dataset to train our model. These images are preprocessed using Error Level Analysis (ELA). The dataset we used is from Kaggle, you can search it with keyword "fake image/video classification dataset". For training the MTCNN classifier we used keras module of python. This classifier will have 3 convolution layers, 3 maxpool layers, 1 flattening layer and finally an output layer with sigmoid activation. We've used Image Data Generator for training the model. Then we saved the model weights and its structure so that every time when we use this model we will not have to train it again and again. Then we've run the model with different real and fake face images. The trained model was able to recognize the image as fake or real.

CHAPTER 7

REFERENCES

- https://www.researchgate.net/publication/338382561_Deep_Fake_Image_Detection_Based_on_Pairwise_Learning
- <https://medium.com/analytics-vidhya/fake-faceimage-classification-5a4151db9b8d>
- https://www.ripublication.com/ijaer20/ijaerv15n3_8.pdf
- IRACST - International Journal of Computer Science and Information Technology & Security (IJSITS), ISSN: 2249-9555 Vol.7, No.2, Mar-April 2017
- https://drive.google.com/open?id=1smby8vBB0g8bNtUsQ10OdjF-4FK9k_GS
- <https://towardsdatascience.com/face-detectionusing-mtcnn-a-guide-for-face-extraction-with-a-focus-on-speed-c6d59f82d4>