Authentication Verifier for Images and its location using TensorFlow/Keras

T.E. mini-project report submitted in partial

fulfilment of the requirements of the degree of

Computer Engineering

by

Pratik Pathak EU1182007 (27) [A]

Shirley Pereira EU1182005 (32) [A]

Kaustubh Rai EU1172112 (78) [B]

Under the guidance of

Mrs. Aditi Raut
Assistant Professor

Department of Computer Engineering

St. John College of Engineering and Management, Palghar

University of Mumbai

2020–2021

CERTIFICATE

This is to certify that the T.E. mini-project entitled **“Authentication Verifier for Images and its Location using TensorFlow/Keras”** is a Mini Project report of

“Pratik Pathak” (EU1182007) (27) [B]

“Shirley Pereira” (EU1182005) (32) [B]

“Kaustubh Rai” (EU1172112) (78) [A]

submitted to University of Mumbai in partial fulfilment of the requirement for the award of the degree of **“Computer Engineering”** during the academic year 2020–2021.

Mrs. Aditi Raut

Guide

Dr. Rahul Khokale

Head of Department

Dr. G.V. Mulgund

Principal

T.E. Mini-Project Report Approval

This mini-project synopsis entitled ***Authentication Verifier for Images and Location using TensorFlow/Keras*** by ***Pratik, Shirley, Kaustubh*** is approved for the degree of ***Computer Engineering*** from ***University of Mumbai***.

Examiners

1.--

2.--

Date:

Place:

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature
Pratik Pathak (EU1182007)

Signature
Shirley Pereira (EU1182005)

Signature
Kaustubh Rai (EU1172112)

Date:

Abstract

Image processing has always been a core part of Computer Vision. This project aims to take the idea further by trying to provide an acute testing of not only the image's authenticity but also its GPS location. We present a machine learning model capable of evaluating whether the image being used has been tampered with or pre-processed, allowing the user to be sure of the authenticity of the image they use. For this, a new incremental learning approach is used for detecting image forgery. The model is trained on the Kaggle Dataset and performance measures such as confusion matrix are used. We also present a front-end for the end user to interact with the model on a desktop.

With this project we hope to increase people's awareness and allow them to verify and trust their own resources, so that as the world moves towards a Digital Age, it moves towards it in a secure and reassured manner.

Keywords—*Convolution Neural Network, Error Level Analysis, Image Forgery, ReLU, Features Extraction*

Table of Contents

List of Figures

.

List of Abbreviations

CNN	Convolution Neural Network
DL	Deep Learning
ELA	Error Level Analysis
ML	Machine Learning
SJCEM	St. John College of Engineering and Management

Chapter 1

Introduction

Computer Vision is the subset of Computer Science that deals with the ability of a computer to 'see' and understand visual data, such as images and videos. Tasks in Computer Vision include object detection, image labelling, motion analysis, etc. With the recent advancements in Artificial Intelligence, Deep Learning Systems, Compute Power and Data availability, numerous tasks which were previously thought impossible have now become possible.

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. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms a core research area within engineering and computer science disciplines too.

A software routine that changes the appearance of an image or part of an image by altering the shades and colors of the pixels in some manner. Filters are used to increase brightness and contrast as well as to add a wide variety of textures, tones and special effects to a picture.

1.1 Motivation

These days, where filters and photoshop is rampant, one can never be sure of whether what you see is truly what it is or not. Especially in situations where Money & Property is concerned. This leads to in-person inspections and wastage of various resource that could be used elsewhere. So we have the idea of making a image-verifier that not only verifies the authenticity of the image itself, but also the location at which it was taken. Allowing us to cross verify with the database provided by the facility for whether the image was taken at that specific location or not.

We believe the main use for this project could be in government jobs, in which we could check whether the construction of a facility was done properly(No Photo Editing) and on proper location. This could help the concerned people keep a track of progress of these facilities. The location that the facility has to be in will be given as a mini database and submitted into the Application which will be used to cross-verify.

1.2 Problem Statement

To create and train a machine learning algorithm that can check whether there has been any adulteration or filters applied to the image along with verifying its GPS location

1.3 Objectives

The objectives are as follows:

- To Identify Altered Images: This project help users to do on-spot checking of images for tampering.
- To show that with enough resources and planning, the incorporation of Digital Resource can be an everyday norm without the risk of getting scammed.

1.4 Scope

Due to the increasing incorporation of Digital Technologies in everyday life, people getting scammed on the internet is a danger to real-life property and finances. Using this project, people can be stress free and sure that what you see is what you get.

This product can be especially useful in the areas of :

- Government Officials trying to verify the progress of Government facilities being built.
- Users wanting to find whether the image has been tampered with or not. (Can be used in business/law for image verification of digital signatures)

Chapter 2

Review of Literature

This chapter contains information about the pre-existing research on the topic of Fake Image that we have referred to while working on our project. Image forging has been a topic of interest among the computer vision community for quite a while now, and has been researched on by multiple major computer or information technology companies such as Google and Microsoft. Referring to and taking inspiration from their works has been very helpful towards the completion of our project.

2.1 Detecting Fake Images on Social Media using Machine Learning

In this technological era, social media has a major role in people’s daily life. Most people share text, images, and videos on social media frequently (e.g. Twitter, Snapchat, Facebook, and Instagram). Images are one of the most common types of media share among users on social media. So, there is a need for monitoring of images contained in social media. It has become easy for individuals and small groups to fabricate these images and disseminate them widely in a very short time, which threatens the credibility of the news and public confidence in the means of social communication. This research attempted to propose an approach to extracting image content, classify it and verify the authenticity of digital images and uncover manipulation. Instagram is one of the most important websites and mobile image sharing applications on social media. This allows users to take photos, add digital photographic filters and upload pictures. There are many unwanted contents in Instagram's posts such as threats and forged images, which may cause problems to society and national security. This research aims to build a model that can be used to classify Instagram content (images) to detect any threats and forged images. The model was built using deep algorithms learning which is Convolutional Neural Network (CNN), Alexnet network and transfer learning using Alexnet. The results showed that the proposed Alexnet network offers more accurate detection of fake images compared to the other techniques with 97%. The results of this research will be helpful in monitoring and tracking in the shared images in social media for unusual content and forged images detection and to protect social media from electronic attacks and threats.

2.2 FaceForensics++: Learning to Detect Manipulated Facial Images

The rapid progress in synthetic image generation and manipulation has now come to a point where it raises significant concerns for the implications towards society. At best, this leads to a loss of trust in digital content, but could potentially cause further harm by spreading false information or fake news. This paper examines the realism of state-ofthe- art image manipulations, and how difficult it is to detect them, either automatically or by humans. To standardize the evaluation of detection methods, we propose an automated benchmark for facial manipulation detection1. In particular, the benchmark is based on Deep- Fakes [1], Face2Face [59], FaceSwap [2] and NeuralTextures [57] as prominent representatives for facial manipulations at random compression level and size. The benchmark is publicly available2 and contains a hidden test set as well as a database of over 1:8 million manipulated images. This dataset is over an order of magnitude larger than comparable, publicly available, forgery datasets. Based on this data, we performed a thorough analysis of data-driven forgery detectors. We show that the use of additional domain specific knowledge improves forgery detection to unprecedented accuracy, even in the presence of strong compression, and clearly outperforms human observers.

2.3 Learning to Detect Fake Face Images in the Wild

Although Generative Adversarial Network (GAN) can be used to generate the realistic image, improper use of these technologies brings hidden concerns. For example, GAN can be used to generate a tampered video for specific people and inappropriate events, creating images that are detrimental to a particular person, and may even affect that personal safety. In this paper, we will develop a deep forgery discriminator (DeepFD) to efficiently and effectively detect the computer-generated images. Directly learning a binary classifier is relatively tricky since it is hard to find the common discriminative features for judging the fake images generated from different GANs. To address this shortcoming, we adopt contrastive loss in seeking the typical features of the synthesized images generated by different GANs and follow by concatenating a classifier to detect such computer-generated images. Experimental results demonstrate that the proposed DeepFD successfully detected 94.7% fake images generated by several state-of-the-art GAN.

2.4 Forensics Face Detection From GANs Using Convolutional Neural Network

The rapid development of Generative Adversarial Networks (GANs) brings the new challenge in anti-forensics face techniques. Many applications use GANs to create fake images/videos leading identity theft and privacy breaches. In this paper, we proposed a deep convolutional neural network to detect forensics face. We use GANs to create fake faces with multiple resolutions and sizes to help data augments. Moreover, we apply a deep face recognition system to transfer weight to our system for robust face feature extraction. In additional, the network is fined tuning suitable for real/fake image classification. We experimented on the validation data from AI Challenge and achieved good results.The rapid development of Generative Adversarial Networks (GANs) brings the new challenge in anti-forensics face techniques. Many applications use GANs to create fake images/videos leading identity theft and privacy breaches. In this paper, we proposed a deep convolutional neural network to detect forensics face. We use GANs to create fake faces with multiple resolutions and sizes to help data augments. Moreover, we apply a deep face recognition system to transfer weight to our system for robust face feature extraction. In additional, the network is fined tuning suitable for real/fake image classification. We experimented on the validation data from AI Challenge and achieved good result.

Chapter 3

Requirement Analysis

Requirement analysis consists of all the functional and non-functional requirements of the project. Functional requirements are those requirements, which are responsible for proper output or functioning of the device, and non-functional requirements are those, which are required for the development of the project. In systems engineering and software engineering, requirements analysis focuses on the task that determines the needs or conditions to meet the new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analysing, documenting, validating and managing software or system requirements. Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design. Hardware and software are also a part of requirement analysis of a system. The hardware requirements are those requirements which are useful to the project in terms of hardware components for ex: connecting wires, micro-controllers etc. Software requirements are software which are required or important for developing the system to its best version for ex: programming languages, IDE etc. All the software's and modules mentioned above have some impact on the system, so all the requirements need to be fulfilled.

3.1 Minimum Software Requirements

- Platform: Windows 8/Windows 10 (Linux For REST-API, WSL if using Windows)
- Python 3.6 or 3.7
- Code Editor/IDE: Visual Studio Code/JetBrains PyCharm/Jupyter

3.2 Minimum Hardware Requirements

- Architecture: Intel x86-64 (64-bit, aka AMD64)
- RAM: 4 GB
- Free Storage: 512 MB

Chapter 4

Design

Diagrams enable better visualization and understanding of the working and scope of a project. Designing of any prototype initiates with a ground level diagram or flowchart and later the entities are defined along with their attributes. The relationships between the entities are established further. The workflow is determined and the functions are distributed to all the entities. The classes are defined and the structure of the system is elaborated further. The sequence of the workflow is specified. The executable part is accordingly planned and constructed and the overall system is described as a flowchart or tree diagram. This chapter contains all the design and diagrams related to the project. UML diagram like,

1. Use-Case Diagram

2. Sequence Diagram

4.1 Use Case Diagram

Use case diagrams are referred to as behaviour diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). It shows the user's interaction with the system and the relationship between the user and different use cases in which the user is involved.

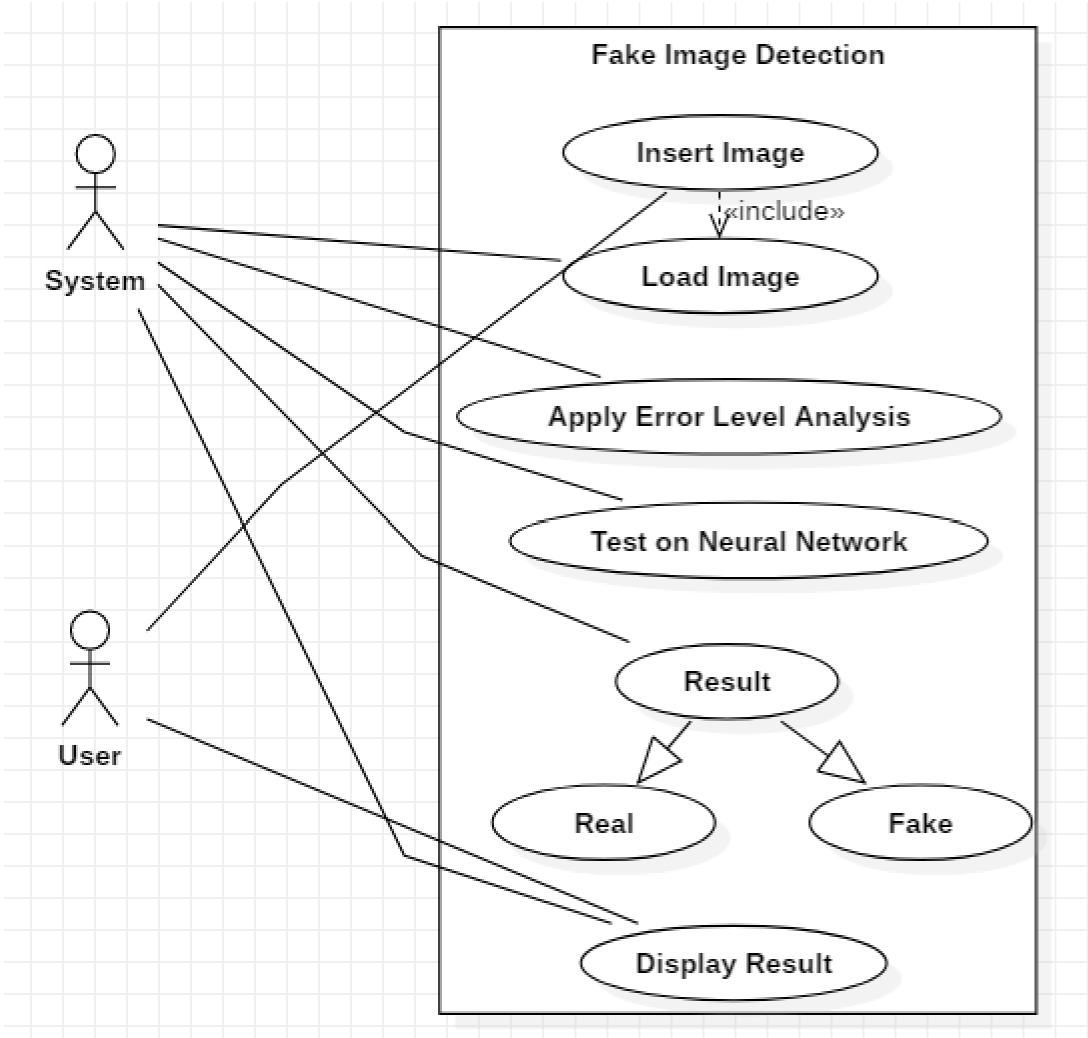

Fig. 4.1 Use Case Diagram

4.2 Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. Terms like event diagrams or event scenarios can be used to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. The system will trigger when the user opens the application. When the user submits submits a picture either by clicking it with their camera or selecting an existing image from their gallery, it will be sent to the backend server via a HTTP POST request, where the picture will undergo some pre-processing before being fed into the neural network for feature extraction and caption generation. The feature extraction is done via a pre-trained InceptionV3 model and the caption is generated using a RNN. The caption text will be sent to the client as a JSON response and the text will also be converted to voice audio on the client application.

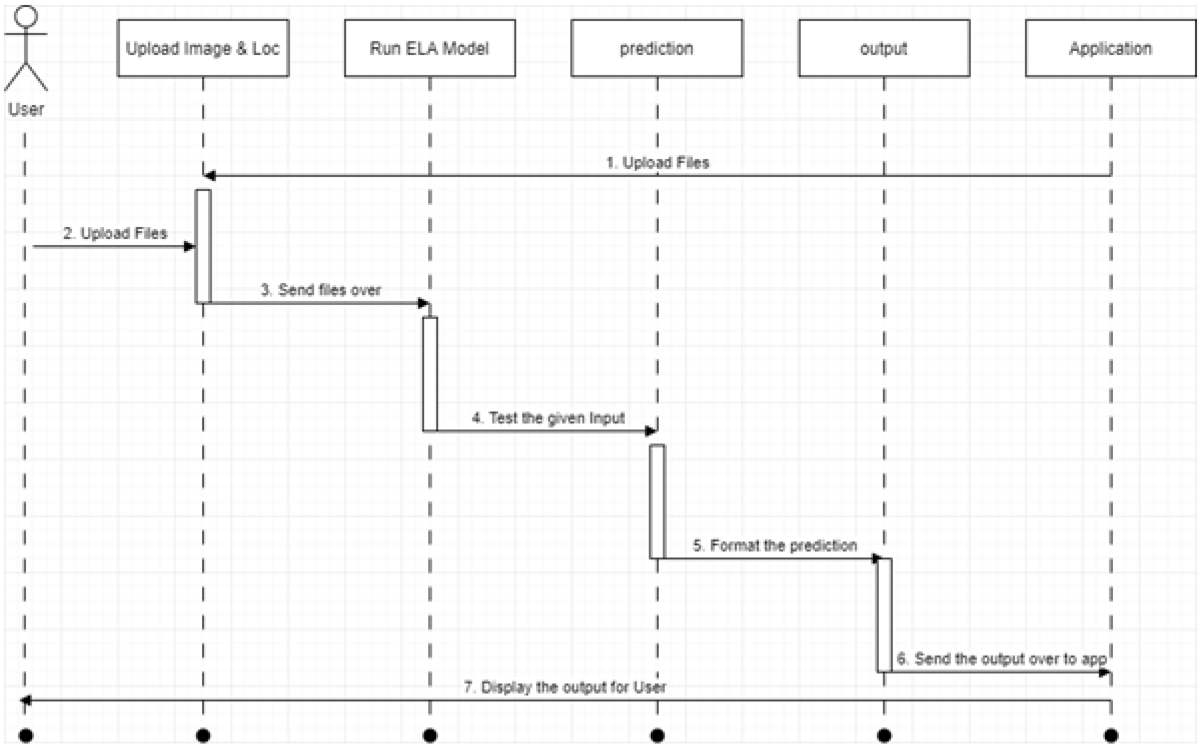

Fig 4.2.1: Sequence diagram for Image Verification Application

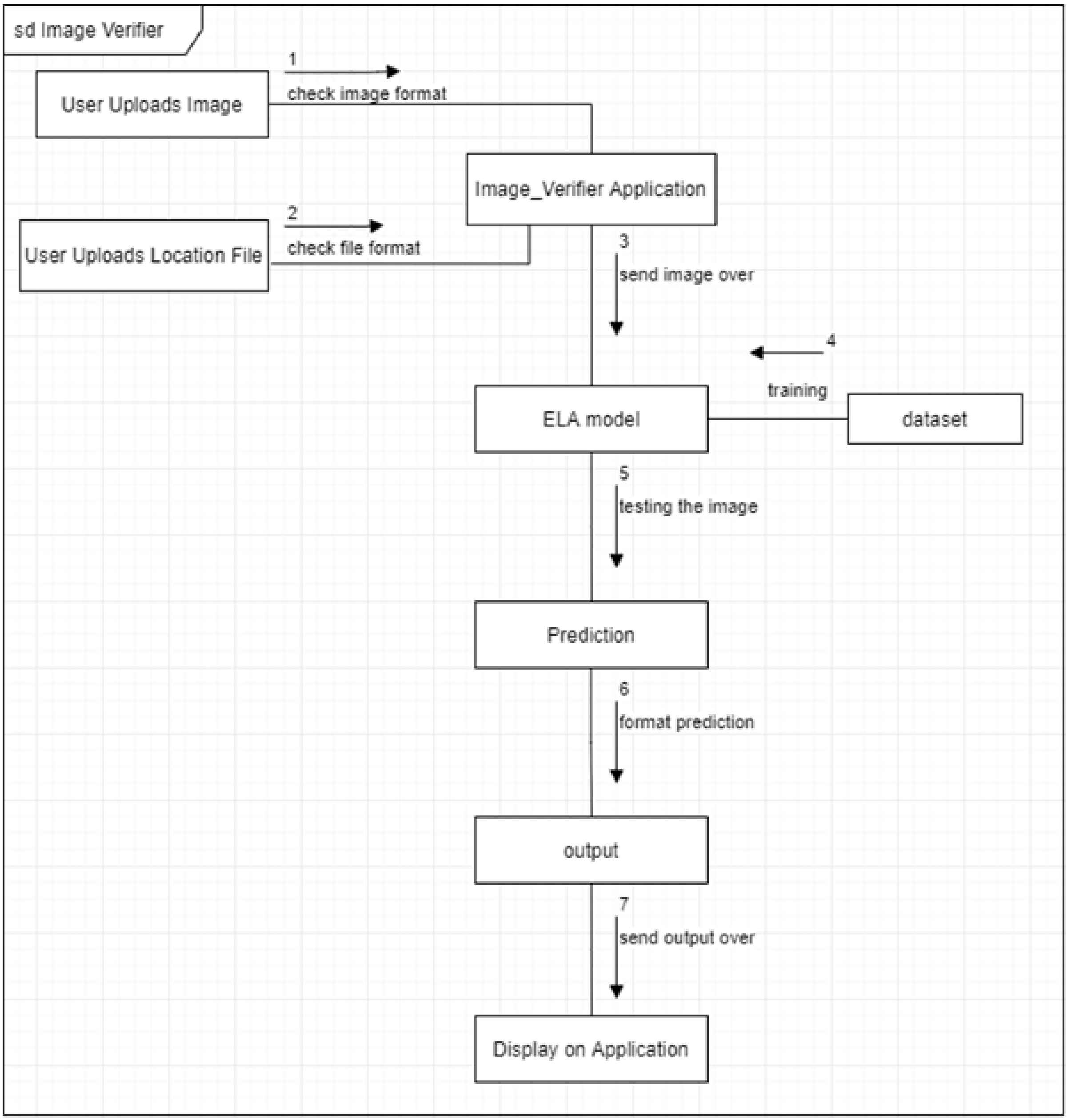

Fig 4.2.2: Collaboration diagram for Image Verification Application

Chapter 5

Report on Present Investigation

This chapter contains the breakdown of the system compartments and the way we built the ELA model and Application.

5.1 Proposed System

The project is divided into 4 Major Modules,

- Module 1: Obtaining Geo-location Metadata from the Image.
  - o In this stage we will be using OSINT Framework for extracting the metadata.
  - o This data will be used further in module 3.
- Module 2: Detection of Image Manipulation using TensorFlow/Keras
  - o We plan on including a way to check for filters and alterations using photoshop
- Module 3: To assess whether the geo-location of the image is accurate.
  - o Finding the distance between the pin-point location of the image extracted in module 1 and the geo-data provided in the database by the user.
  - o Using this distance we will verify whether the image was taken in the vicinity of the property at which the facility will be built.
- Module 4: Mobile/Web Application Module 4: Mobile/Web Application
  - o We plan on building a Web Application for start and continue to a Mobile Application for convenience.
  - o The Application will be the interface with the user where the user will be allowed to upload the image and the database(if required).

5.1.1 Model Architecture

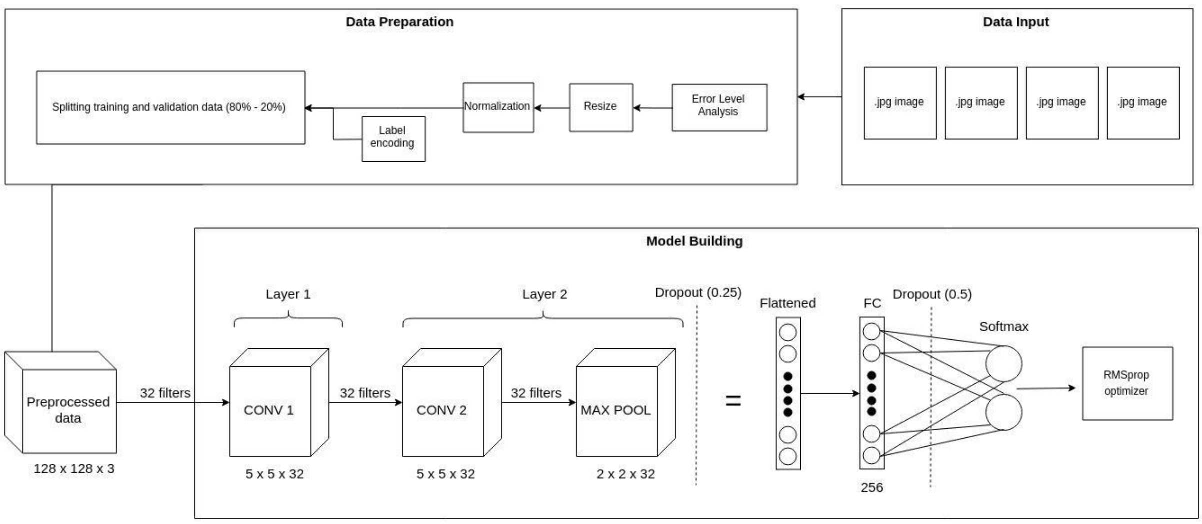

Fig 5.1.1: Model Architecture

5.2 Implementation

This subsection contains the steps involved in the implementation of the system.

5.2.1 Algorithm/Flowchart

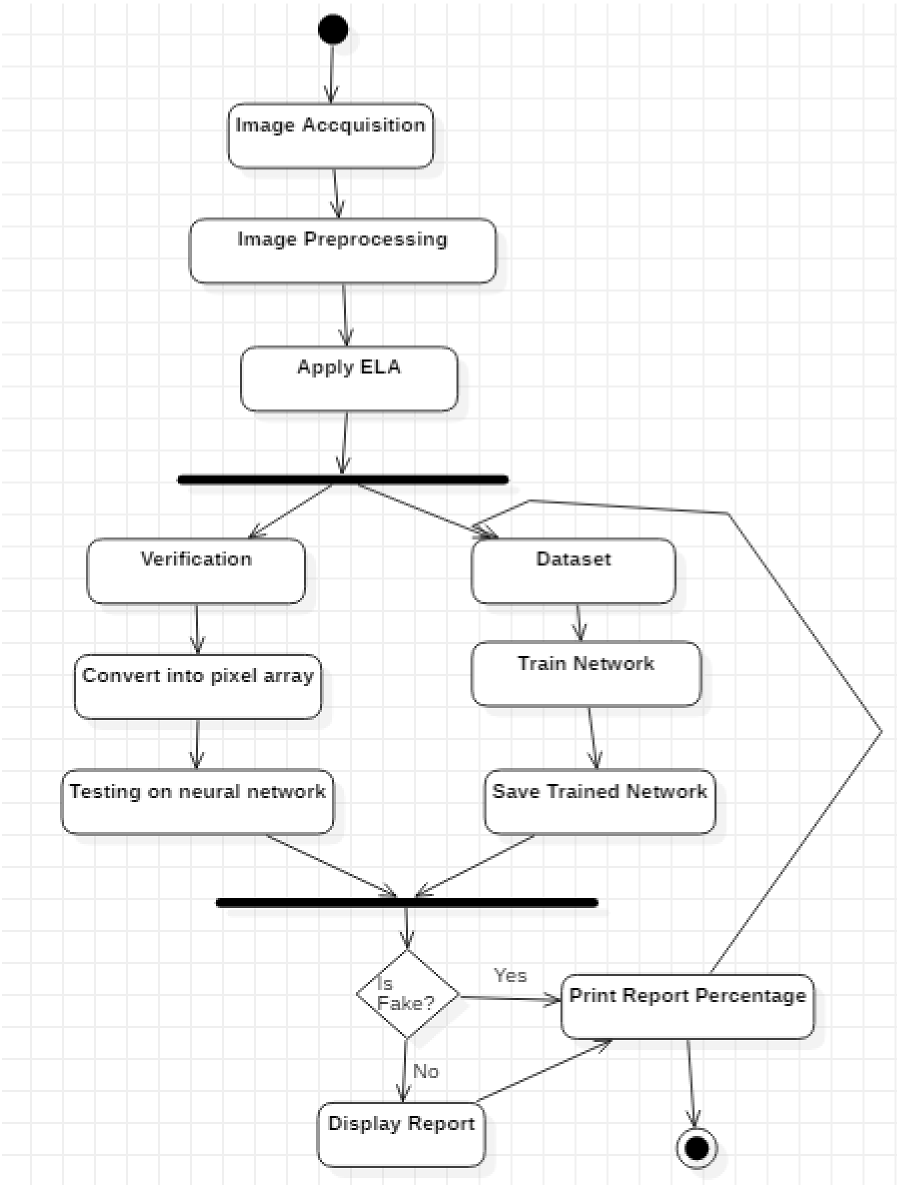

Fig 5.2.1: Flowchart

5.2.2 Dataset

The ‘Kaggle’ dataset is used to train the gender detection machine learning model. Kaggle, a subsidiary of Google LLC, allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers. It offers a public data platform, a cloud-based workbench for data science.

5.2.3 Pseudo Code

1. Filter the image by selecting only “.png” and “.jpg” in both Real and Fake.
2. Resize the image to 128x128 pixel
3. Split the images to train and test
4. Create the model
5. Choose activation function as Relu and Softmax for the layers
6. Set epoch =30, BatchSixe=32
7. Set Optimizer “ADAM” set loss function to “binary_crossentropy”
8. Fit the MODEL
9. Save the model
10. Plot the loss and accuracy curves for training and validation
11. Plot confusion_matrix
12. Check and optimize the model

5.2.4 Training

```
In [35]: import random
path = 'kaggle\input\casia-dataset\CASIA2\Au'
for dirname, _, filenames in os.walk(path):
    for filename in filenames:
        if filename.endswith('jpg') or filename.endswith('png'):
            full_path = os.path.join(dirname, filename)
            X.append(prepare_image(full_path))
            Y.append(1)
            if len(Y) % 500 == 0:
                print(f'Processing {len(Y)} images')

random.shuffle(X)
X = X[:2100]
Y = Y[:2100]
print(len(X), len(Y))
```

```
Processing 500 images
Processing 1000 images
Processing 1500 images
Processing 2000 images
Processing 2500 images
Processing 3000 images
Processing 3500 images
Processing 4000 images
Processing 4500 images
Processing 5000 images
Processing 5500 images
Processing 6000 images
Processing 6500 images
Processing 7000 images
2100 2100
```

Fig 5.2.4 (a): Preprocessing

```
In [36]: path = 'kaggle\input\casia-dataset\CASIA2\Tp'
for dirname, _, filenames in os.walk(path):
    for filename in filenames:
        if filename.endswith('jpg') or filename.endswith('png'):
            full_path = os.path.join(dirname, filename)
            X.append(prepare_image(full_path))
            Y.append(0)
            if len(Y) % 500 == 0:
                print(f'Processing {len(Y)} images')

print(len(X), len(Y))
```

```
Processing 2500 images
Processing 3000 images
Processing 3500 images
Processing 4000 images
4164 4164
```

```
In [37]: X = np.array(X)
Y = to_categorical(Y, 2)
X = X.reshape(-1, 128, 128, 3)
```

Train Test split with 80:20 ratio

```
In [38]: X_train, X_val, Y_train, Y_val = train_test_split(X, Y, test_size = 0.2, random_state=5)
X = X.reshape(-1,1,1,1)
print(len(X_train), len(Y_train))
print(len(X_val), len(Y_val))
```

```
3331 3331
833 833
```

Fig 5.2.4 (b): Pre-processing and splitting data into train and test

5.2.5 Testing

```
In [50]: class_names = ['fake', 'real']

In [51]: real_image_path = 'kaggle\input\casia-dataset\casia\CASIA2\Au\Au_ani_00001.jpg'
         image = prepare_image(real_image_path)
         image = image.reshape(-1, 128, 128, 3)
         y_pred = model.predict(image)
         y_pred_class = np.argmax(y_pred, axis = 1)[0]
         print(f'Class: {class_names[y_pred_class]} Confidence: {np.amax(y_pred) * 100:0.2f}')

         Class: real Confidence: 99.76

In [52]: fake_image_path = 'kaggle\input\casia-dataset\casia\CASIA2\Tp\Tp_D_NRN_S_N_ani10171_ani00001_12458.jpg'
         image = prepare_image(fake_image_path)
         image = image.reshape(-1, 128, 128, 3)
         y_pred = model.predict(image)
         y_pred_class = np.argmax(y_pred, axis = 1)[0]
         print(f'Class: {class_names[y_pred_class]} Confidence: {np.amax(y_pred) * 100:0.2f}')

         Class: fake Confidence: 99.55
```

Fig 5.2.5 (a): Testing on a single image

```
In [53]: fake_image = os.listdir('kaggle\input\casia-dataset\casia\CASIA2\Tp')
         correct = 0
         total = 0
         for file_name in fake_image:
             if file_name.endswith('jpg') or filename.endswith('png'):
                 fake_image_path = os.path.join('kaggle\input\casia-dataset\casia\CASIA2\Tp', file_name)
                 image = prepare_image(fake_image_path)
                 image = image.reshape(-1, 128, 128, 3)
                 y_pred = model.predict(image)
                 y_pred_class = np.argmax(y_pred, axis = 1)[0]
                 total += 1
                 if y_pred_class == 0:
                     correct += 1
         #            print(f'Class: {class_names[y_pred_class]} Confidence: {np.amax(y_pred) * 100:0.2f}')

In [54]: print(f'Total: {total}, Correct: {correct}, Acc: {correct / total * 100.0}')

         Total: 2064, Correct: 2031, Acc: 98.40116279069767
```

Fig 5.2.5 (b): Testing on all images

5.2.6 Screenshots of the output with description

Homepage:

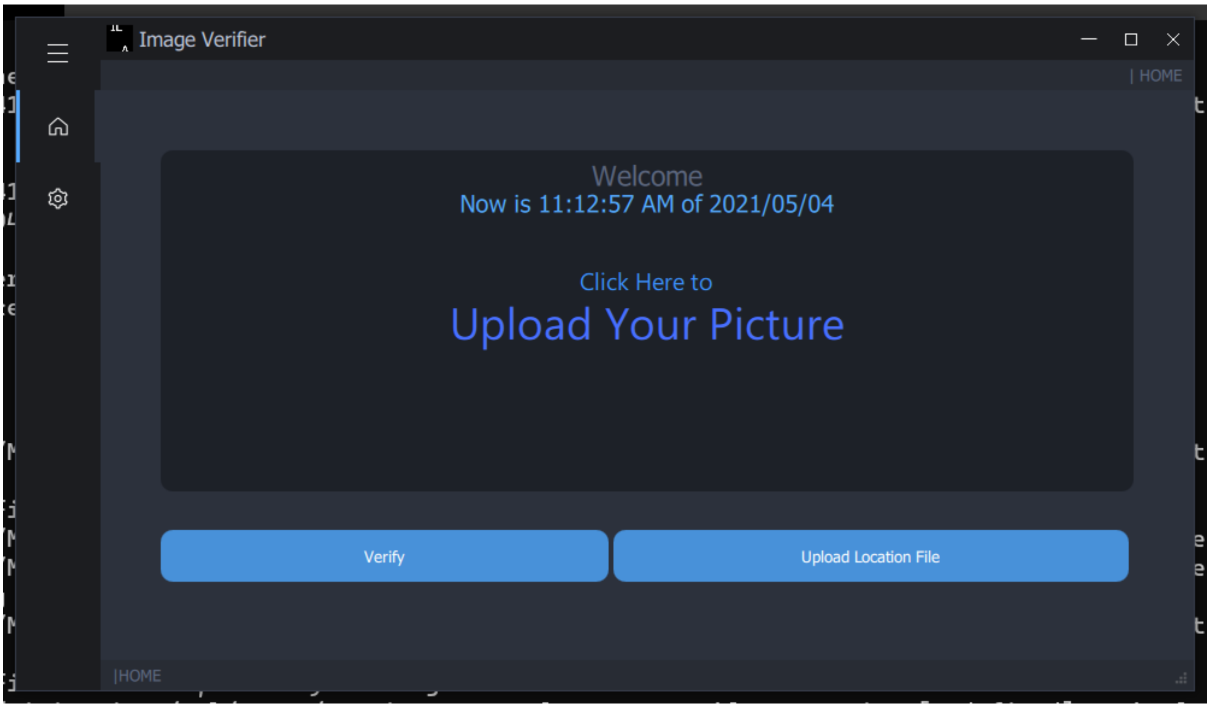

Fig 5.2.6 (a): HomeScreen

Selecting an image:

Click on the “Upload Your Picture” it will image selection dialogue box

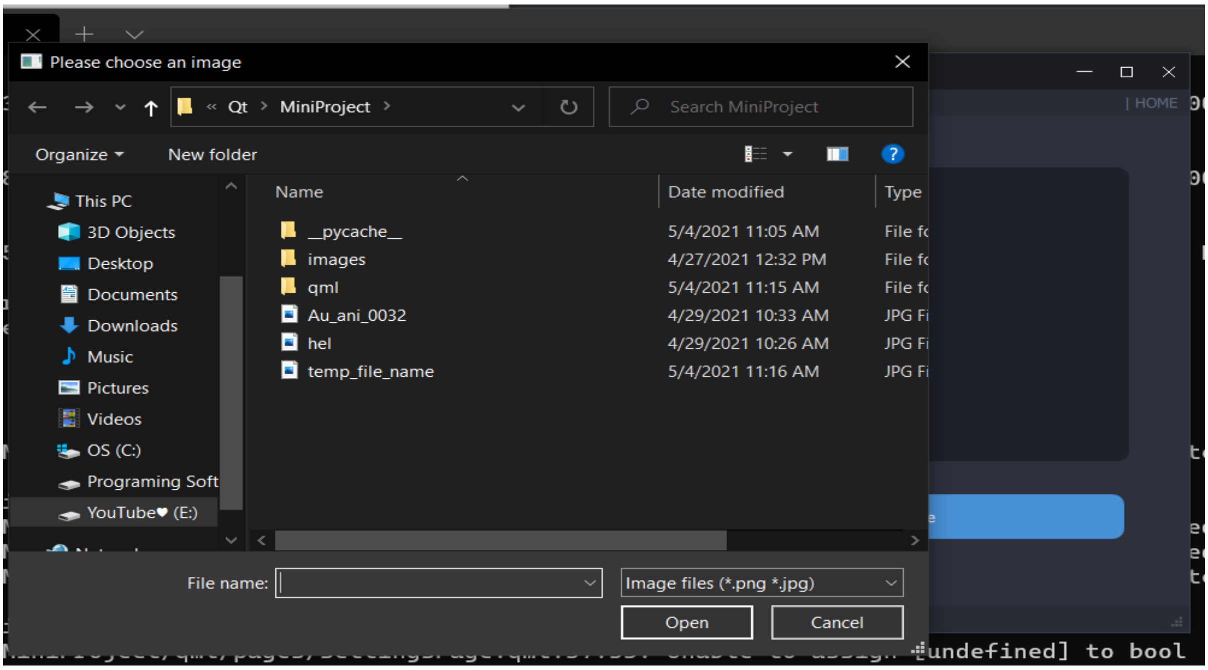

Fig 5.2.6 (b): Upload Image File

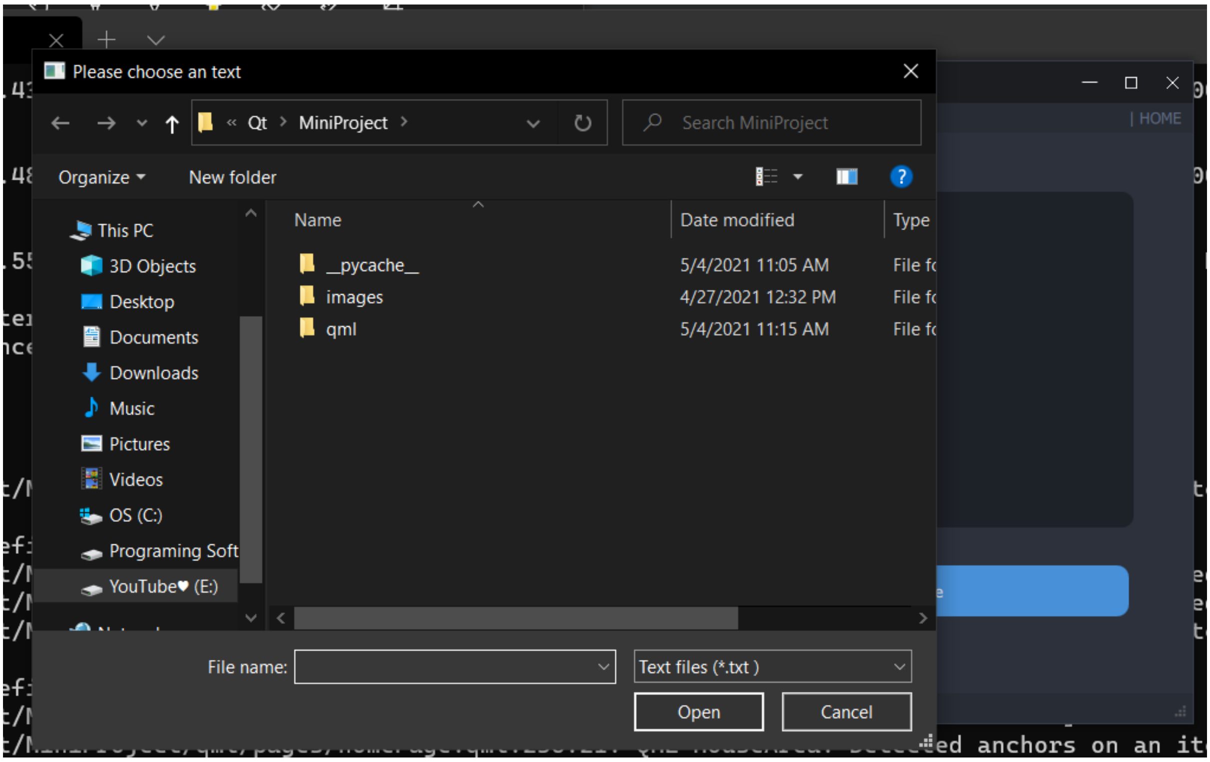

Fig 5.2.6 (c): Upload Location Text File

Loading Screen: Click on ‘Verify’. It will show the loading Screen

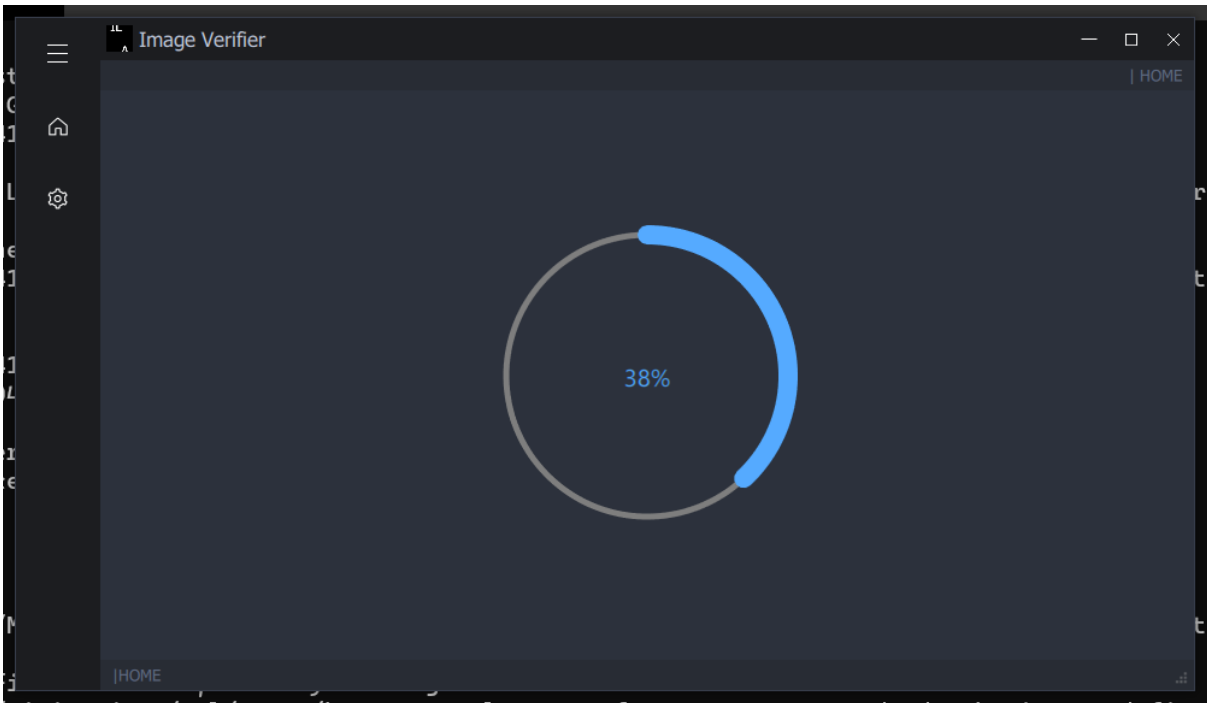

Fig 5.2.6 (d): Loading Screen

Output Screen:

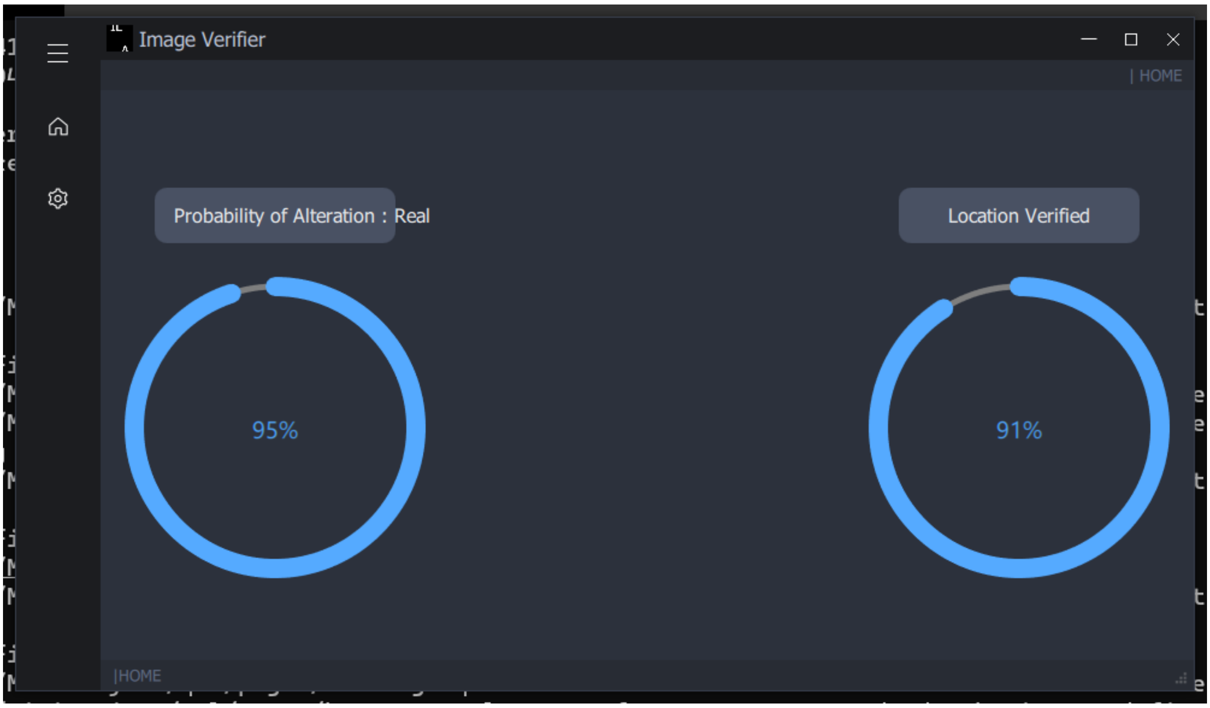

Fig 5.2.6 (e): Result

Chapter 6

Result and Analysis

This Chapter contains the results of the testing methods applied to model after the training phase. It also contains the screenshot of the output and the technologies used for the implementation of system with their definitions.

6.1 Results

In this section, the screenshots and their explanations are discussed.

Out[28]:

After converting to ELA image

```
In [29]: convert_to_ela_image(real_image_path, 90)
```

Out[29]:

Fig 6.1.1: Real Image and it’s ELA

In Fig 6.1.1 a Real Image and it’s ELA image has been shown. If the image is real then it’s ELA image will have Black and White distinct dots.

Out[30]:

Fig 6.1.2: Fake Image

Out[31]:

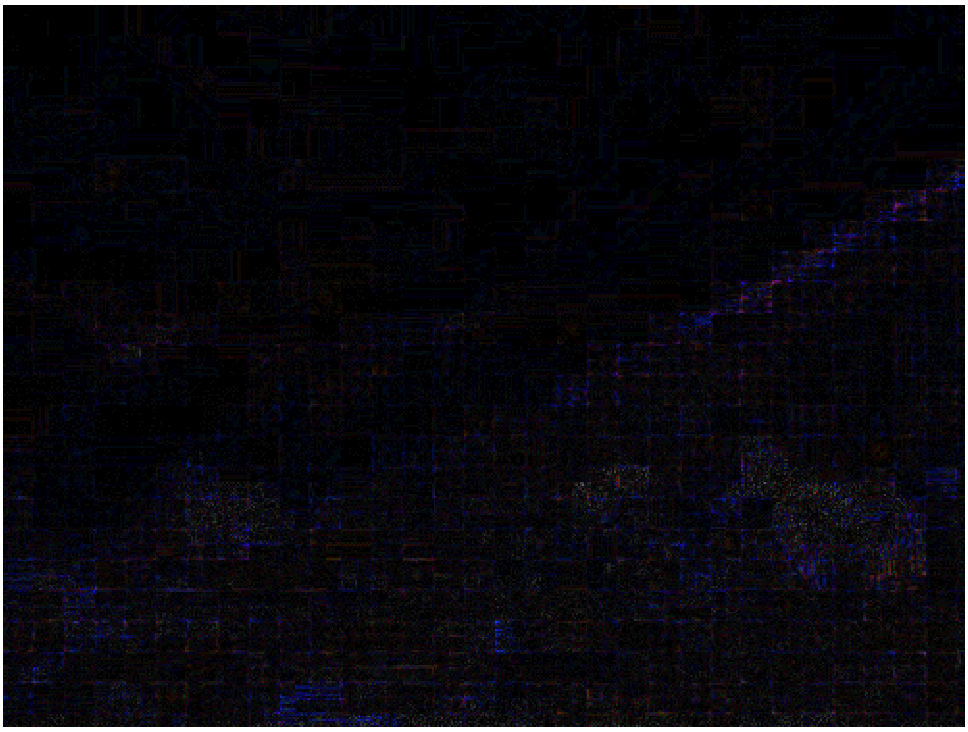

Fig 6.1.3: Fake Image ELA

Fig 6.1.2 is a Fake image and Fig 6.1.3 is the ELA image of Fig 6.1.2 . As we can see it does not have blank and white distinct dots hus we can say that this image is digitally altered.

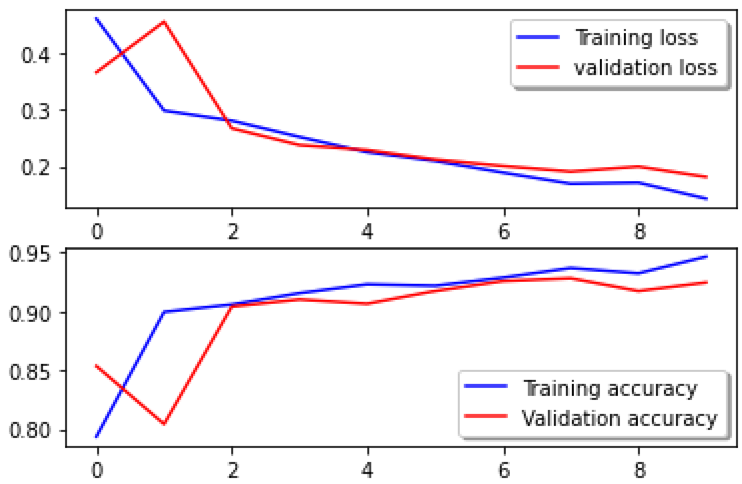

Fig 6.1.4: Training Loss vs Validation Loss

6.2 Confusion Matrix

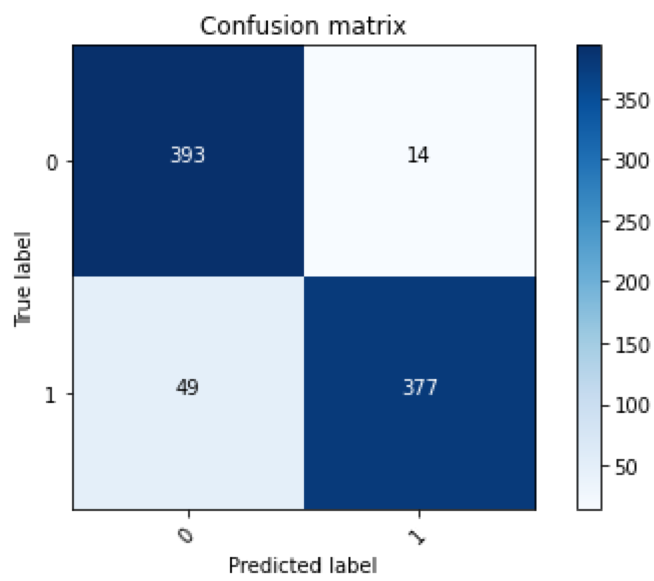

Fig 6.2: Confusion Matrix

Chapter 7

Conclusion

The proposed system is an attempt to build an optimized CNN architecture for Image Alteration Verification. The model has obtained an accuracy of 94.52%. The research work can be extended for other alteration method. The method is suitable for custom hardware implementation targeted for real-time processing in resource-constrained environments. Image Alteration recognition is essential and critical for many applications in the commercial domains such as government applications of human-computer interaction and computer-aided physiological or psychological analysis, since it contains a wide range of information regarding the characteristics. The existing approaches also have some limitations such as low accuracy, low efficiency, and restricted application domain. This review brings a new insight analysis on various gender classification methods in different contexts.

References

[1] G.Mohamed Sikandar, "100 Social Media Statistics You must know," [online] Available at: https://blog.statusbrew.com/social-media-statistics-2018-for-business/ [Accessed 02 Mar 2019].

[2] Damian Radcliffe, Amanda Lam, "Social Media in the Middle East,"[online]Available:https://www.researchgate.net/publication/323185146_Social_Media_in_the_Middle_East_The_Story_of_2017 [Accessed 06 Feb 2019].

[3] GMI_BLOGGER,"Saudi Arabia Social Media Statistics," GMI_ blogger. [online] Available at:https://www.globalmediainsight.com/ blog/saudi-arabia-social-media-statistics/ [Accessed 04 May 2019].

[4] Kit Smith,"49 Incredible Instagram Statistics,". Brandwatch. [online] Available at: https://www.brandwatch.com/blog/instagram-stats/ [Accessed 10 May 2019].

[5] Selling Stock. (2014). Selling Stock. [online] Available at: https://www. selling-stock.com/Article/18-billion-images-uploaded-to-the-web-every-d [Accessed 12 Feb 2019].

[6] Li, W., Prasad, S., Fowler, J. E., & Bruce, L. M. (2012). Locality-preserving dimensionality reduction and classification for hyperspectral image analysis. IEEE Transactions on Geoscience and Remote Sensing, 50(4), 1185–1198.

[7] A. Krizhevsky, I. Sutskever, & G. E. Hinton, (2012). Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems, 1097–1105.

[8] K. Ravi, (2018). Detecting fake images with Machine Learning. Harkuch Journal

[9] L. Zheng, Y. Yang, J. Zhang, Q. Cui, X. Zhang, Z. Li, et al. (2018). TI-CNN: Convolutional Neural Networks for Fake News Detection. United States