VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANASANGAMA, BELAGAVI - 590018

Project Report

on

CNN BASED DEEPFAKES DETECTION MODEL FOR FORGED IMAGE IDENTIFICATION IN SOCIAL MEDIA

Submitted in partial fulfillment for the award of degree of

Bachelor of Engineering
in
COMPUTER SCIENCE AND ENGINEERING

Submitted by

Preethi V	**1BG17CS072**
R Narendranath Reddy	**1BG17CS075**
Suhas H	**1BG17CS101**
Swarnamalya M	**1BG17CS104**

Internal Guide
Prof. Priyanka Padki
Assistant Professor, Dept. of CSE
BNMIT, Bengaluru

Vidyayāmruthamashnuthe

B.N.M. Institute of Technology

Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.
All UG branches – CSE, ECE, EEE, ISE & Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021
Post box no. 7087, 27th cross, 12th Main, Banashankari 2nd Stage, Bengaluru- 560070, INDIA
Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

Department of Computer Science and Engineering

2020 - 2021

B.N.M. Institute of Technology

Approved by AICTE, Affiliated to VTU, Accredited as grade A Institution by NAAC.
All UG branches – CSE, ECE, EEE, ISE & Mech.E accredited by NBA for academic years 2018-19 to 2020-21 & valid upto 30.06.2021
Post box no. 7087, 27th cross, 12th Main, Banashankari 2nd Stage, Bengaluru- 560070, INDIA
Ph: 91-80- 26711780/81/82 Email: principal@bnmit.in, www. bnmit.org

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vidyayāmruthamashnuthe

CERTIFICATE

Certified that the project work entitled **CNN BASED DEEPFAKES DETECTION MODEL FOR FORGED IMAGE IDENTIFICATION IN SOCIAL MEDIA** carried out by **Ms. Preethi V** USN **1BG17CS072**, **Mr. R Narendranath Reddy** USN **1BG17CS075**, **Mr. Suhas H** USN **1BG17CS101, Ms. Swarnamalya M** USN **1BG17CS104** are bonafide students of VIII Semester, BNM Institute of Technology in partial fulfillment for the award of Bachelor of Engineering in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the year 2020-21. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The Project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said Degree.

Prof. Priyanka Padki **Assistant Professor,** **Department of CSE,** **BNMIT, Bengaluru**	**Dr. Sahana D. Gowda** **Professor and HOD,** **Department of CSE,** **BNMIT, Bengaluru**	**Dr. Krishnamurthy G N** **Principal,** **BNMIT,** **Bengaluru**

Examiner 1:

Examiner 2:

ACKNOWLEDGEMENT

The success and final outcome of the project Forged Image Identification in social media required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project.

We would like to thank **Sri. Narayan Rao R Maanay**, Secretary, BNMEI, Bengaluru for providing excellent academic environment in the college.

We take this opportunity to express our profound gratitude to **Prof. T J Rama Murthy**, Director, BNMIT, Bengaluru for his constant support and encouragement.

We would like to thank **Dr. S Y Kulkarni**, Additional Director, BNMIT, Bengaluru for his constant encouragement.

We would like to express our gratitude to **Prof. Eishwar N Maanay**, Dean, BNMIT, Bengaluru for his relentless support, guidance and assistance.

We would like to thank **Dr. Krishnamurthy G N**, Principal, BNMIT, Bengaluru for his constant encouragement.

We would like to thank, **Dr. Sahana D Gowda**, Professor and Head of the Department of Computer Science and Engineering whose guidance and support was truly valuable.

We would like to thank our project coordinator, **Mr. Raghavendra C K,** Assistant Professor, Department of Computer Science for his guidance and assistance in the project work.

We are thankful to our project guide, **Mrs. Priyanka Padki,** Assistant Professor, Department of Computer Science for her guidance and valuable advice at every stage of our report in the project work.

Finally, we take this opportunity to extend our earnest gratitude and respect to our parents and friends, for giving us continuous moral support at all times in all possible ways.

PREETHI V
R NARENDRANATH REDDY
SUHAS H
SWARNAMALYA M

ABSTRACT

With the rapid progress of recent years, techniques that generate and manipulate multimedia content can now provide a very advanced level of realism. The boundary between real and synthetic media has become very thin. On the one hand, this opens the door to a series of exciting applications in different fields such as creative arts, advertising, film production, video games. On the other hand, it poses enormous security threats. Software packages freely available on the web allow any individual, without special skills to create very realistic fake images. Fake images has become a central problem in the last few years, especially after the advent of the so called deep fakes. Fake images are so easily manipulated with the help of powerful and easy-to-use deep learning tools like auto encoders (AE) or generative adversarial networks (GAN). These can be used for malicious purposes, like building fake-news campaigns to manipulate public opinion during elections, commit fraud, discredit or blackmail people with fake images. In the long run, it may also reduce trust in journalism, including serious and reliable sources. To contend with this growing threat, we implement a model working in 2 different stages. Firstly through Metadata Analysis where the metadata information is extracted and searched for keywords like Photoshop, Adobe, Gimp etc. which elucidates the high possibility of image being tampered. Secondly, a CNN based automated model that utilizes Error Level Analysis (ELA) through deep learning and modern data-driven forensic techniques and therefore renders it capable of detecting forged multimedia images. Although not visually apparent, these correlations are often violated by the nature of how deep fake images are created, and therefore, the presented tool be used for visual media integrity verification.

TABLE OF CONTENTS

SL. No.	Title	Page No.
1.	**INTRODUCTION**	**1-4**
	1.1 Overview	1
	1.2 Motivation	2
	1.3 Problem Statement	2
	1.4 Objectives	2
	1.4.1 Dataset	3
	1.5 Summary	3
2.	**LITERATURE SURVEY**	**5-11**
	2.1 Introduction	5
	2.2 Literature Survey	6
	2.3 Methodology	9
	2.4 Summary	11
3.	**SYSTEM REQUIREMENTS**	**12-18**
	3.1 Introduction	12
	3.1.1 Feasibility Study	12
	3.1.2 Operational Feasibility	13
	3.1.3 Technical Feasibility	13
	3.1.4 Economic Feasibility	13
	3.2 System Requirements	14
	3.3 Software Requirements	15
	3.4 Hardware Requirements	16
	3.5 Functional - Non-Functional Requirements	17
	3.6 Summary	18

4.	**SYSTEM DESIGN**	**19-25**
	4.1 Introduction	19
	4.2 Proposed System	20
	4.3 Data Flow Diagram	22
	4.4 Summary	26
5.	**IMPLEMENTATION**	**27-30**
	5.1 Introduction	27
	5.2 System Design	27
	5.2.1 VGG16 CNN Architecture	27
	5.2.2 Algorithm	28
	5.3 Summary	30
6.	**RESULT ANALYSIS**	**31-42**
	6.1 Introduction	31
	6.1.1 Types of Tests	32
	6.2 Test Cases	34
	6.3 Results	36
	6.3.1 Snapshots	37
	6.4 Performance Evaluation	40
	6.5 Summary	41
	CONCLUSION	**43**
	FUTURE ENHANCEMENTS	**44**
	REFERENCES	

List of Figures

Figure No.	Description	Page No.
Figure 2.3.1	Block Diagram of Implementation	10
Figure 4.3.3.1	Level 0 DFD	24
Figure 4.3.3.2	Level 1 DFD	25
Figure 4.3.3.3	Level 2 DFD	25
Figure 5.2.1.1	VGG16 CNN Architecture	28
Figure 5.2.2.1	CNN Layers	29
Figure 5.2.2.2	Pooling Layer	29
Figure 6.1	Black and White Box Testing	33
Figure 6.3.1.1	Metadata Analysis for Real Image	37
Figure 6.3.1.2	Metadata Information for Real Image	38
Figure 6.3.1.3	Neural Network Analysis for Real Image	38
Figure 6.3.1.4	Metadata Analysis for Fake Image	39
Figure 6.3.1.5	Metadata Information for Fake Image	39
Figure 6.3.1.6	Neural Network Analysis for Fake Image	40

List of Tables

Table No.	Description	Page No.
Table 6.1	Unit Testing for Fake Image Detection 1	35
Table 6.2	Unit Testing for Fake Image Detection 2	35
Table 6.3	Unit Testing for Fake Image Detection 3	36
Table 6.4	Unit Testing for Fake Image Detection 4	36
Table 6.5	Model Performance	41

CHAPTER 1
INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 Overview

Fake images has become a central problem in the last few years, especially after the advent of the so called deep fakes, i.e., fake images manipulated with the help of powerful and easy-to-use deep learning tools, such as Auto Encoders (AE) or Generative Adversarial Networks (GAN). With this technology, creating realistic manipulated media assets may be very easy, provided one can access large amounts of data. The rapid proliferation of image editing technologies has increased both the ease with which images can be manipulated and the difficulty in distinguishing between altered and natural images. Applications include photography, video-games, virtual reality, and may soon expand to movie productions. With the advent of social networking services such as Facebook and Instagram there has been a huge increase in volume of fake images generated in the last decade and it has become a major concern for the internet companies. These images are prime sources of fake news and are often used malevolent ways such as for mob incitement. The very same technology, however, can also be used for malicious purposes, like creating fake porn images to blackmail people, or building fake-news campaigns to manipulate the public opinion. In the long run, it may also reduce trust in journalism, including serious and reliable sources. These fakes are easy to spot since they are generated for fun and involve well-known actors and politicians in unlikely situations. 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. Image control has disintegrated our trust of computerized pictures, with progressively unobtrusive fraud techniques representing a regularly expanding test to the integrity of images and their legitimacy. With the progress of advanced image controlling software and modifying tools, an electronic picture can be successively controlled. Checking the decency of pictures and recognizing indications of modifying without requiring extra pre-inserted data of the image is the basic field of inspection.

1.2 Motivation

Digital image manipulation is the act of distorting the contents of an image in order to fulfil some fraudulent purposes. In digital forensics, such manipulations are known as forgeries. Due to the evolution of technology, various photo manipulation tools have been developed and made available over the internet. The images generated are nearly impossible to detect if real or fake. It therefore becomes extremely important to find whether the image under consideration has been manipulated or not, as images are used in court of laws, in news, in sciences, for medicinal purposes and in many more fields as a proof of result. The major focus of a digital image forgery or manipulation is to contrive the illegitimate changes in an authentic image so that the image closely mimics the legitimacy of an authentic one. Thus, it becomes harder for the human visual system to differentiate between legitimate and forged/manipulated images. Comprehensive ongoing research aims at delivering solutions to the problem of differentiating a forged image with that of an authentic image. Devising effective and real-time detection and localization methods is currently important as these forgery attacks are increasing with time. Hence, there arises the requirement for efficient and reliable image forgery detection methods that can distinguish between authentic and forged images.

1.3 Problem Statement

The boundary between real and synthetic media has become very thin and thus poses enormous security threats as they can be misused to abuse information and generate fake identification. Therefore detecting forged images in social media is critical for protecting individuals from various misuses and authenticity in media. To content with this growing threat, the project aims to implement an automated image forensic platform that is capable of detecting forged and manipulated multimedia images.

1.4 Objectives

Many fake images are spreading through digital media nowadays. Detection of such fake images is inevitable for the unveiling of the image based cybercrimes. Forging images and identifying such images are promising research areas in this digital era. The tampered images are a detected using neural network which also recognizes the regions of the image that have been manipulated and reveals the segments of the original image. It can be implemented on Android platform and hence made available to common users. The compression ratio of the foreign content in a fake image is different from that of

the original image and is detected using Error Level Analysis. Another feature used along with compression ratio is image metadata. Although it is possible to alter metadata content making it unreliable on its own, here it is used as a supporting parameter for error level analysis decision.

The proposed project aims to present an analysis of the methods for visual media integrity verification, that is, the detection of manipulated images. Special emphasis will be placed on the emerging phenomenon of deep fakes and is implemented by using a Metadata Extractor and Analyzer and by building a CNN based automated model that utilizes Error Level Analysis(ELA). VGG16 architecture of CNN is used for pretraining the model which is recognized for more precise classification results.

1.4.1 Dataset

A dataset is a collection of related, discrete item of related data that may be accessed individually or in the combination or managed as a whole entity.

Before going into the dataset overview, the terminology used will be made clear.

Fake image: An image that has been manipulated/doctored using the two most common manipulation operations namely: copy/pasting and image splicing.

Pristine image: An image that has not been manipulated except for the resizing needed to bring all images to a standard size as per competition rules.

The structure of the dataset contains two directories comprising of 5123 fake and 7491 real training images under varying sizes obtained from CASIA dataset in Kaggle. All images are in JPEG format.

1.5 Summary

The rapid growth of digital image processing technologies and editing software has given rise to large amounts of tampered images circulating in our daily lives. Digital image acquisition is now a simple task and information in the form of digital images is drastically increasing on social media, which has both positive and negative impacts on a society in many different ways. Advanced user-friendly tools have made it easy to manipulate image content in order to gain illegal advantage or to make false propaganda. Deep fakes that leverage ML to manipulate images has garnered widespread attention for its fraudulent use in Face swapping, Facial re-enactment,

Pornographic images and Fake-news campaigns. This undermines the credibility and trustworthiness of digital images and also creates false beliefs in many real world situations. Hence it is generating a great demand for automatic forgery detection algorithms in order to determine the authenticity of a candidate image.

To contend with these threats, this project aims to implement an automatic image forensic platform that works in 2 stages, simplest one using the Metadata Analyzer based on ELA and the other using deep learning techniques of CNN to detect if an image is authentic or forged.

CHAPTER 2
LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

Analyzing existing system is an important step in the development process. Before implementing the proposed technique, it is necessary to determine the features that are in existence and to analyze their working by considering various factors that influence them like load balancing, server switching and their efficiency. Once it is found that they are satisfactory, the next step would be to determine the programming language and the platform that it is implemented in, which requires a lot of support from experienced programmers, books or websites. These are some of the factors that are considered at length before developing the proposed system.

Literature survey is a text of scholarly paper which includes the current and latest knowledge considering substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature surveys are secondary sources and do not report new or original experimental work. A literature review is a body of text that aims to review the critical aspects of current knowledge including substantial findings as well as theoretical and methodological contributions to a particular topic.

Fake images on social networks is a fast growing problem. Commercial media editing tools allow anyone to remove, add, or clone people and objects to generate fake images. Image manipulation has eroded trust of digital images, with more subtle forgery methods posing an ever-increasing challenge to the integrity of images and their authenticity. Many techniques have been proposed to detect such conventional fakes, but new attacks emerge by the day. The aim of this systematic survey is to gain insights into the current project on the detection of these forgeries by comprehensively analyzing the selected studies.

2.2 Literature Survey

[1] Chih-Chung Hsu , Chia-Yen Lee , Yi-Xiu Zhuang , " Learning to Detect Fake Face Images in the Wild" , IEEE 2018 , International Symposium on Computer, Consumer and Control (IS3C)

The main aim of this paper is to develop a deep forgery discriminator (DeepFD) to effectively detect the fake images generated from different GAN (Generative Adversarial Networks). It implements a deep neural network based discriminator that adopts the technique of contrastive loss in seeking the typical features of the synthesized images generated from different GANs. The proposed method has two learning phases: First, the fake and real image dataset is jointly used to learn the discriminative features D1 based on the contrastive loss. Later, a discriminator (classifier) D2 will be concatenated with D1 to further distinguish the fake image. Experimental results demonstrate that the proposed DeepFD successfully detected 94.7% fake images generated by several state-of-the-art GANs.

[2] Yuanfang Guo, Xiaochun Cao, Wei Zhang, Rui Wang, Member, " Fake Colorized Image Detection " , Submitted To IEEE Transactions On Information Forensics And Security, 2019

This paper aims to detect image colorization, the emerging image editing technique in which grayscale images are colorized with realistic colors. Colorized images which are generated by state-of-the-art methods posses statistical differences for the hue and saturation channels. The two simple and effective detection methods proposed for fake colorized images are: Histogram based Fake Colorized Image Detection (FCID-HIST) and Feature Encoding based Fake Colorized Image Detection (FCID-FE). In FCID-HIST, four detection features, the hue feature Fh, the saturation feature Fs, the dark channel feature Fdc and the bright channel feature Fbc, are proposed to detect forgeries. Intuitively, to differentiate the fake colorized images from the natural images, the distinctive features should reveal the largest divergences between the two types of images.

[3] Savita Walia & Krishan Kumar , " Digital image forgery detection: a systematic scrutiny " , Australian Journal of Forensic Sciences, 2019

The aim of this paper on systematic survey is to gain insights into the current research

on the detection of image forgeries by comprehensively analysing the methods to implement the detection process.

Active methods: The methods used in the active methods exploit certain information inserted inside the digital image. The embedded data in the image is used to detect the source of such an image or to perceive an alteration in that image. Digital watermarking and digital signatures are examples of active techniques. Passive Methods: Passive or blind methods of forgery detection take advantage of the traces left by the image processing operations such as noise variation, lighting and shadows to detect the manipulations on the image. Copy-move forgery Detection Methods: In copy-move forgery detection methods, the major goal is to match the regions in the image, which are similar and can distinguish between the copied and the pasted region. Splicing Based Methods: Splicing-based methods use a variety of features such as Bi-coherence features, camera response function, DCT and DWT coefficients, invariant image moments, Weber local descriptors, etc.

[4] Khurshid Asghar, Xianfang Sun4 , Paul L. Rosin4 , Mubbashar Saddique2 , Muhammad Hussain3, " Edge–texture feature-based image forgery detection with cross-dataset evaluation" , Springer-Verlag GmbH Germany, part of Springer Nature 2019

This paper aims to perform extensive experiments using the developed dataset as well as the public domain benchmark datasets in which the results demonstrate the robustness and effectiveness of the proposed method for tamper detection and validate its cross-dataset generalization. The proposed system is composed of four major components, i.e., (i) preprocessing, (ii) feature extraction, (iii) classification model building and (iv) testing, using the trained model with cross-dataset validation. The model is trained using an SVM classifier on a set of images and then the trained model is used to test/recognize unseen authentic and forged images. A cross-validation (CV) protocol is used to divide each dataset or combination of datasets into k-fold (tenfold). DRLBP is a robust texture descriptor, which models the structural changes occurred in images due to forgery using edge–texture features that incorporate information such as texture, boundary discontinuities and inconsistencies.

[5] Francesco Marra, Diego Gragnaniello, Davide Cozzolino, Luisa Verdoliva, "Detection of GAN-generated Fake Images over Social Networks", IEEE Conference on Multimedia Information Processing and Retrieval, 2018

This paper aims to detect images manipulated by GAN based image–to-image translation. Under the adversarial training paradigm, two actors play the role: the image-to-image network (generator), and a support network (discriminator). The discriminator tries to tell apart real images from those created by the generator, while the Generator is trained to deceive the discriminator. To assess the performance in a real scenario where the manipulation is unknown a priori, a leave-one-manipulation-out (LOMO) procedure is adopted. At each iteration, all images referring to a certain category are set aside for validation while the remaining ones are used for training. Thereby the classifier does not adapt to features of a specific class of translations, but learns patterns that are shared by all images generated by this procedure, thus generalizing across different manipulations. The performance of several image forgery detectors against image-to-image translation is studied, both in ideal conditions, and in the presence of compression, routinely performed upon uploading on social networks. The study, carried out on a dataset of 36302 images, shows that detection accuracies up to 95% can be achieved by both conventional and deep learning detectors, but only the latter keep providing a high accuracy, up to 89%, on compressed data.

[6] Khurshid Asghar, Zulfiqar Habib & Muhammad Hussain "Copy-move and splicing image forgery detection and localization techniques: a review ", Australian Journal of Forensic Sciences, 2019

This paper aims to explore various forged image detection techniques employed for copy-move and image splicing. Texture and Intensity based Algorithms: The proposed algorithm is used to extract image features using the statistical analysis of pixels of small overlapped blocks of an image, then they compare the similarity of these blocks. Finally, possible duplicated regions are identified using intensity-based characteristic features. SVD based Algorithms: The proposed copy-move forgery detection algorithm is based on a discrete wavelet transform (DWT) and SVD along with robust features for matching duplicate regions in images. PCA-based algorithms: Principle Component Analysis (PCA) is also a candidate algorithm to extract image features and has been used to detect copy-move forgery and spliced images. DCT-based algorithms: Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT) are popular techniques to transform an image into the frequency domain before extracting its features. However the drawback was these methods were robust only to JPEG compressions and did not provide achievable performance in terms of accuracy.

2.3 Methodology

The system proposed detects almost all kinds of tampered images (spliced, colored, blurred) and yields result with an accuracy of 90%. Once an image is selected for processing, it is tunneled into 2 separate stages. First stage is metadata analysis. After extracting metadata, the metadata text is fed into metadata analysis module. In the second stage the data is preprocessed using Error level analysis in which the image is compressed. Incase the image is forged, the respective areas will have high levels of intensity, that is the forged areas will have unique black and white spots compared to other regions. This feature will then be used by the CNN to classify the image as Real or Fake.

METADATA ANALYSIS

Most image files do not just contain a picture. They also contain information (metadata) about the picture. Metadata provides information about a picture's pedigree, including the type of camera used, color space information, and application notes. Different picture formats include different types of metadata. Some formats, like BMP, PPM, and PBM contain very little information beyond the image dimensions and color space. In contrast, a JPEG from a camera usually contains a wide variety of information, including the camera's make and model, focal and aperture information, and timestamps. Metadata provides information related to how the file was generated and handled. This information can be used to identify if the metadata appears to be from a digital camera, processed by a graphical program, or altered to convey misleading information.

Metadata analyzer is basically a tag searching algorithm. If keywords like Photoshop, Gimp, Adobe etc. is found in the text and then the possibility of being tampered is increased. Two separate variables are maintained which are called fakeness and realness. Each variable represents the weight of being real or fake image. Once a tag is taken, it is analyzed and corresponding variable is incremented by a certain predefined weight.

DATA PROCESSING USING ERROR LEVEL ANALYSIS

Error Level Analysis is a forensic method to identify portions of an image with different levels of compression. The technique could be used to determine if a picture has been digitally modified. Error Level Analysis (ELA) permits identifying areas within an image that are at different compression levels. With JPEG images, the entire

picture should be at roughly the same level. If a section of the image is at a significantly different error level, then it likely indicates a digital modification.

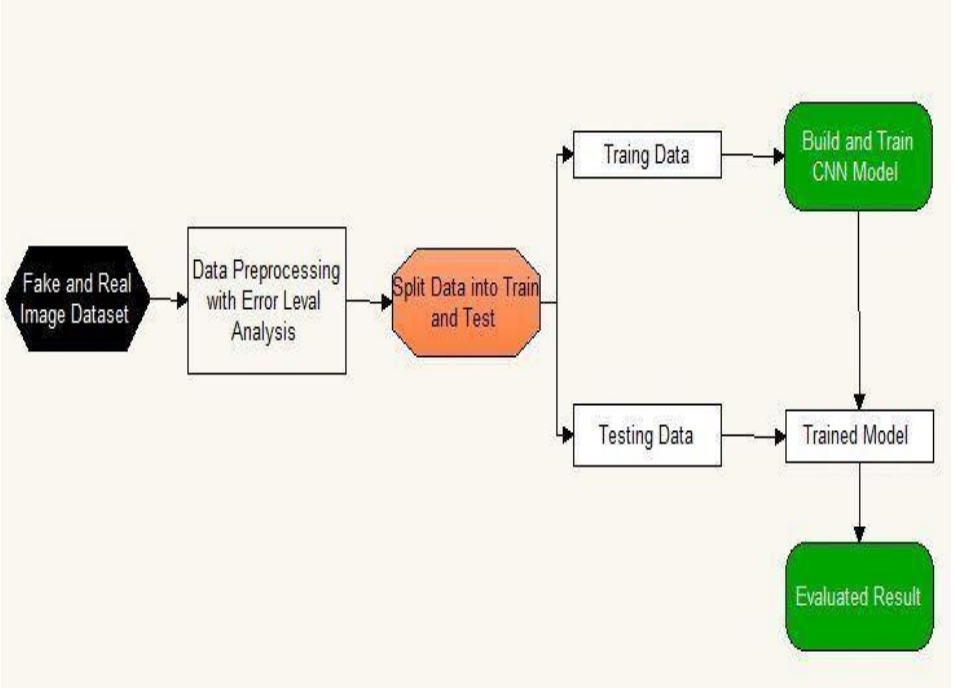

Figure 2.3.1 Block Diagram of Implementation(CNN)

CNN CLASSIFIER MODEL USING VGG16 ARCHITECTURE

To classify whether an image is authentic or forged, a general approach can be developed using CNN making the implementation more robust. In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analyse visual imagery. It is a specialized type of deep learning algorithm designed for working with 2D image data. A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of a series of convolutional layers that convolve with a multiplication or other dot product. CNN image classification takes the output image yielded after ELA pre-processing, as the input image and performs processing, training and classifies it under certain categories. The VGG16 architecture of CNN is used for pretraining the model as it is a significant way of improving the performance of deep neural networks by increasing their size. This architecture has gained recognition in producing more precise classification results.

2.4 Summary

A thoughtful systematic survey of the available literature on detection of various image forgery attacks has been studied. To classify whether an image is authentic or forged, a general and automatic approach is developed using CNN and Metadata Analyzer. Metadata is analogous to the chain of custody for evidence handing. It can identify how a picture was generated, processed, and last saved. Metadata analyzer is basically a tag searching algorithm. If keywords like Photoshop, Gimp, Adobe etc. is found in the text and then the possibility of being tampered is increased. Two separate variables are maintained which are called fakeness and realness. CNN is a kind of neural network that shares weights among neurons in the same layer. It discovers spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. CNN combines image segmentation, feature extraction and the classification process in one trainable module. CNN accepts a two dimensional (2D) raw image with minimal pre-processing and retains the 2D topology throughout its training. Classification is performed during training and, at the end of training, the learned weights help to classify the input image accurately. The system aims to verify the authenticity of digital images without any prior knowledge of the original image. There are many ways for tampering an image such as splicing or copy-move, resampling an image (resize, rotate, stretch), addition and removal of any object from the image. To counter this threat the proposed project aims to build a CNN based automated model that utilizes Error Level Analysis(ELA). Expected outcome of this project is when a new digital image is given to the system it applies the ELA Algorithm and CNN to the image and classify if the given image is Fake or Real.

CHAPTER 3

SYSTEM REQUIREMENTS

CHAPTER 3
SYSTEM REQUIREMENTS

3.1 Introduction

A software requirement specification is a description of a software system to be developed. It lays out functional and non-functional requirements and they include a set of use cases that describe user interactions that the software must provide. The software requirements specification document enlists enough and necessary requirements that are required for the project development. Software requirement specification is an important part of software development process. These contents are very much useful in fulfilling goals while implementing the project. This section is divided into two major parts, the software requirements and the hardware requirements necessary for Forged image identification in social media.

Requirement analysis also called requirements engineering. It is the process of determining user expectation for a new or modified product. These features called requirements must be quantifiable, relevant and detailed. Requirement analysis is critical to the success of a system or software project. Conceptually requirements analysis includes three types of activities.

- Eliciting requirements for business process, documentation and stakeholder interviews. This is sometimes also called requirements gathering.
- Analyzing requirements for determining whether the stated requirements are clear, complete, consistent and unambiguous and resolving any apparent conflict.
- Recording requirements for requirements to be documented in various forms, usually including a summary list and may include a natural language documents, use case, use stories or process specifications.

3.1.1 Feasibility Study

A feasibility study is carried out to select the best system that meets performance requirements. The main aim of feasibility study activity is to determine whether it would be financially and technically feasible to develop the product. The feasibility study activity involves the analysis of problem and collection of all relevant information

relating to the product such as the different data items which would be input to the system, the processing required to be carried out on these data. The output data required to be produced by the system as well as various constraints on the behavior of the system. The key objective of the feasibility study is to weigh up three types of feasibility.

They are,

- Operational feasibility
- Technical feasibility
- Economical feasibility

3.1.2 Operational Feasibility

Operational feasibility is necessary as it ensures that the project developed is a successful one. As the execution process of the proposed work is very much user friendly, the operational feasibility of the project is high.

3.1.3 Technical Feasibility

Technical feasibility analysis makes a comparison between the level of technology available and that is needed for the development of the project. The level of technology consists of the factors like software tools, machine environment and platform developed and so on.

3.1.4 Economical Feasibility

This is the most important part of the project because the terms and conditions for implementing the project have to be economically feasible. The risk of finance does not exist as the existing hardware is free of cost. Hence, the system is economically feasible**.**

3.2 System Requirements

In order to operate the system efficiently, all computer software need certain hardware components and other software resources to be present on a computer. These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements minimum and recommended.

3.2.1 Software Interface

Describes the connections between the products and other specific software components (name, version) including databases, operating systems, tools, libraries and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components.

3.2.2 Software Components

The following are the Software Requirements required in the application Forged image identification in social media.

- Operating System : Windows 7/10
- Coding Language : Java
- Web Technology : JavaFX
- Web Server : TomCAT 6.0
- IDE : Netbeans

3.2.3 Hardware Interface

Describes the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware communication protocols to be used.

3.2.4 Hardware Requirements

The following are the Hardware requirements required in Framework for spam detection in online social media.

- System : Intel Core I7
- Hard Disk : 500 GB
- RAM : 8 GB
- System type : 64 bit Operating System

3.3 Software Requirements

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning on an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed on installed separately before the software is installed. Some of the important software requirements are listed below:

Platform:

A computing platform describes some sort of framework either in hardware or software, which allows software to run. Typical platforms include a computer's architecture, operating system or programming languages and their runtime libraries. Operating system is one of the requirements mentioned when defining systems requirements. Software may not be compatible with different versions of same line of operating systems, although some measure of backward compatibility is often maintained. For example, most software designed for Microsoft Windows XP does not run on Microsoft Windows 98, although the converse is not true.

APIs and Drives:

Software making extensive use of special hardware devices like high end display adopters needs special API or newer device drivers. A good example is DirectX, which is a collection of APIs for handling tasks related to multimedia, especially game programming on Microsoft platforms.

Web Browser:

Most web applications and software depending heavily on Internet technologies make use of the default browser installed on system. Microsoft Internet Explorer is a frequent choice of the software running on Microsoft Windows, which makes use of ActiveX controls, despite their vulnerabilities.

3.4 Hardware Requirements

The most common set of requirements defined by any operating system or software application in the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists instead, compatible and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

Architecture:

All computer operating systems are designed for particular computer architecture. Most software applications are limited to particular operating systems running on particular architectures. Although architecture independent operating systems and application exist, most need to be recompiled to run on a new architecture.

Processing Power:

The power of the central processing unit (CPU) is a fundamental system requirement for any software. Most software running on x86 architecture defines processing power as the model and the clock speed of the CPU. Many other features of a CPU that influence its speed and power, like bus speed, cache, and MIPS are often ignored.

This definition of power is often erroneous and inaccurate, as AMD Athlon and Intel Pentium CPUs at similar clock speed often have different throughput speeds. Intel Pentium CPUs have enjoyed a considerable degree of popularity, and are often measured in this category.

Memory:

All software when run, resides in the Random access memory (RAM) of a computer. Memory requirements are defined after considering the demands of the application, operating system, supporting software and files, and other running process. Optimal

performance of other unrelated software running on multi-tasking computer system is also considered when defining this requirement.

3.5 Functional and Non-Functional Requirements

In Software engineering as well as in systems engineering functional requirements defines a function of a system or its components. A function is described as a set of inputs, the behavior, and outputs. A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behavior. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supported to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases.

The following functional requirements are necessary for the application Forged image identification in social media.

- Upload image for forge detection
- Metadata Analysis
- Error Level Analysis
- Training Neural network
- Graph Creation

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design.

Non-Functional requirements are often called “quality attributes” of a system. Other terms for non-functional requirements are “qualities”, “quality goals”, “quality of service requirement”, “constraints” and “non-behavioral requirements”. Informally these are sometimes called “ilities”, from attributes like stability and portability. Qualities that can non-functional requirements can be divided into two main categories.

They are,

- Execution qualities, such as security and usability, which are observable at run time.

- Evolution qualities, such as testability, maintainability, extensibility and scalability, which are embodied in the static structure of the software system.

The following non-functional requirements are necessary for Framework for spam detection inonline social media:

- **Performance**: The performance for the developed project "Forged image identification in social media" depends on how accurately the fake images are detected. The accuracy of classification is done by using cosine similarity algorithm.
- **Availability**: The system developed is available for use by the user at any time as longas internet is available.

3.6 Summary

In this chapter, the importance of feasibility study and requirement analysis is explained. A feasibility study is carried out to select the best system that meets performance requirements. The main aim of the feasibility study activity is to determine whether it would be financially and technically feasible to develop the product. The necessary software and hardware requirements are listed. The complexity involved in each of the algorithms under consideration need to be studied and an appropriate algorithms needs to be chosen in order suit the purposes of the classification task at the hand. By performing the feasibility study, one can determine the applicability of the algorithm and its efficiency in solving the current problem.

CHAPTER 4
SYSTEM DESIGN

CHAPTER 4

SYSTEM DESIGN

4.1 Introduction

This chapter covers the proposed model that is considered for the project. It covers the design decision and the data flow diagram that was used in the process of developing the frame work based spam detection in social media reviews.

4.1.1 Architectural design

The architectural design of the system emphasizes the design of the system architecture that describes the structure, behavior and more views of that system and analysis.

4.1.2 Logical design

The logical design of the system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modeling, using an over-abstract (and sometimes graphical) model of the actual system. In context of the systems, designs are included.

4.1.3 Physical design

The physical design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified/authenticated, how it is processed and how it is displayed. In physical design, the following requirements about the system are decided:

Input requirements

Output requirements

Storage requirements

Processing requirements

System control and backup or recovery

Put another way, the physical portion of the system design can generally be broken down into three sub-tasks:

- User Interface Design
- Data Design
- Physical Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them.

Data Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the system design phase, documentation describing the three sub-tasks is produced and made available for use in next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input keyboard, processing within the CPU, and output via a monitor, printer, etc. It would be monitor, CPU, motherboard, hard drive, modems video/graphics cards, USB slots, etc. It involves a detailed design of a user and product database structure processor and a control processor. The H/S personal specification is developed for proposed system.

4.2 Proposed System

The proposed system works in five crucial stages:

i. Metadata Analysis

ii. Data Preprocessing using Error Level Analysis

iii. Build and Train a CNN Classifier Model

iv. Test and detect output using learned neural network

v. Feedback Analysis using Backpropogation rule

- Metadata Analysis

Metadata provides information about a picture's pedigree, including the type of camera used, color space information, application notes and information related to how the file was generated and handled. This information can be used to identify if the metadata appears to be from a digital camera, processed by a graphical program, or altered to convey misleading information.

- Data Preprocessing using ELA (Error Level Analysis)

Error Level Analysis is a forensic method to identify portions of an image with a different level of compression. The technique could be used to determine if a picture has been digitally modified. They result in poor quality compressed images. Error Level Analysis (ELA) permits identifying areas within an image that are at different compression levels. With JPEG images, the entire picture should be at roughly the same level. If a section of the image is at a significantly different error level, then it likely indicates a digital modification.

- Build and Train a CNN Classifier Model

To classify whether an image is real or tampered, a general and automatic approach can be developed using CNN making the system more robust. Once ELA is calculated, the image is preprocessed to convert into 100x100px width and height. After preprocessing, the image is serialized in to an array. The array contains 30,000 integer values representing 10,000 pixels. Since each pixel has red, green and blue components, 10,000 pixels will have 30,000 values. During training, the array is given as input to the multilayer perceptron network and output neurons also set. There are 2 output neurons. First neuron is for representing fake and the second one for real image. If the given image is fake one, then the fake neuron is set to one and real is set to zero. Else fake is set to zero and real set to one.

- Test and detect output using learned neural network

During testing, the image array is fed into the input neurons and values of output neurons are taken. Sigmoid activation function is deployed in the process.

- Feedback Analysis using Backpropogation rule

The momentum backpropagation learning rule adjust the neuron connection weights. It is a supervised learning rule that tries to minimize the error function. The algorithm is used to effectively train a network through a method called chain rule. The error between system's output and a known expected output is presented to the system which is used to modify it's state accordingly.

4.3 Data Flow Diagram(DFD)

A data Flow Diagram is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as preliminary

step to create an overview of the system without going into great detail, which can later be elaborated.

DFDs can also be used for visualization of data processing (structured design). There are various levels of DFDs that vary in the amount of detail they provide. DFDs shows what kind of information about the timing of process or information about whether processes will operate in sequence or in parallel unlike a flowchart which also shows this information.

When developing any type of computer program, from a simple command-line game to an ornate operating system, one of the most important things a programming the actual code and future increase the productivity of the programmer (or programming group).

4.3.1 Applications of DFD

DFDs ate common way of modeling data flow for software development. For example, a DFD for a word-processing program might show the way the software processes the data that the user enters by pressing keys on the keyboard to produce the letters on the screen.

4.3.2 Significance of DFD

DFDs are popular for software design because the diagram makes the flow of data easy to understand and analyze. DFDs represent the functions and moving through various layers or levels of sub functions. As a modeling technique, DFDs are useful for performing a structured analysis of software problems allowing developers to spot and pinpoint issues in software development. Every system is developed either to satisfy a need or to overcome the drawbacks of the existing system.

- **Organization**

The most important thing a data flow diagram does is to keep the program organized. Programmers use data flow diagram to plan exactly how their new program is going to accomplish its intended purpose. While simpler programs could probably be made without using a data flow diagram for organization, creating more complex ones,

especially with groups of programmers, definitely requires the use of a data flow diagram to help keep the program on track.

- **Decision Building**

In almost every program available to consumers , there are many parts where a program is going to have to make a decision based on data that was given either by the user or from another part of the program what to do when given specific data to work with. Data flow diagram help the programmer figure out what options the programs will need in order to handle the data it is given.

- **Presentation**

The worst possible thing a programmer can do is discussing the program with laypeople is to use the code to explain what the program does and how it will do it. Computer code is like a foreign language to most people, and using it as your backup will only result in confusion about your project. Instead, use the dataflow diagram to explain the program to lay people. It will definitely save the amount of time you would have spent explaining the code to them.

- **Adaptability**

During the course of a project, a programmer will sometimes find a better tool or realize that there is a better way to optimize the code but not be sure where to put it or what else the programmer will need to modify in order to accommodate the code. If the programmer uses a data flow diagram, the diagram will help the programmer be able to see what will happen if certain code is injected into the program.

- **Error detection**

Program can have a lot of errors, or bugs, when they are being made. Because the amount of code can be extensive in bigger programs, sometimes it's difficult to pinpoint where exactly a problem is in the code, but with the help of a debugger and a data flow diagram, a programmer can eventually find the error in question and begin to figure out how to correct it.

4.3.3 Data Flow Diagram Level 0 for Forged image identification is social media.

The Level 0 DFD is the context level data flow diagram, which shows the interaction between the system and the external agents that acts as the data source and data sink The context diagram shows the entire system as a single process, and gives no clues as

to its internal organization. The following figure is the level 0 data flow diagram for the framework Forged image identification in Social media. It is a simple representation of the data flow between the user and the system.

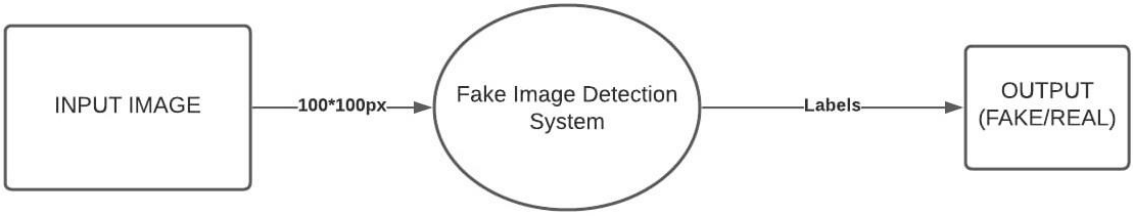

Figure 4.3.3.1 Level 0 DFD

4.3.4 Data Flow Diagram Level 1 and Level 2 for Forged image identification is social media.

The context level data flow diagram is next exploded to produce a data flow diagram that shows the details of the system being modelled. This data flow diagram shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provides all of the functionality of the system as a whole.

At the level 1 of DFD, fake and real image dataset is the input being passed. The images undergo pre-processing like scaling and normalization. The preprocessed data then undergoes 2 level analysis. Firstly, the image data is analyzed under Metadata extractor which deploys Tag-search algorithm to search for metadata that helps in identifying tampering. The second level of analysis is where the image is sent to ELA Analyzer for error level analysis. Feature extraction is then followed by splitting of image set into 80% train and 20% test data. The train dataset which is 10000pixels along with RGB values for each (30000 values) is given to the input layer of CNN (DFD level 5). The VGG16 architecture of CNN is used for pretraining the model as it is the most precise classifier model. The result is analyzed by considering the two output neurons received, first neuron for representingreal and other one for fake image. The final level of DFD depicts the Feedback Analysis which tries to minimize error function using the inputs received from user. By combining the results of both levels of analysis, a reliable fake image detection model is developed and tested.

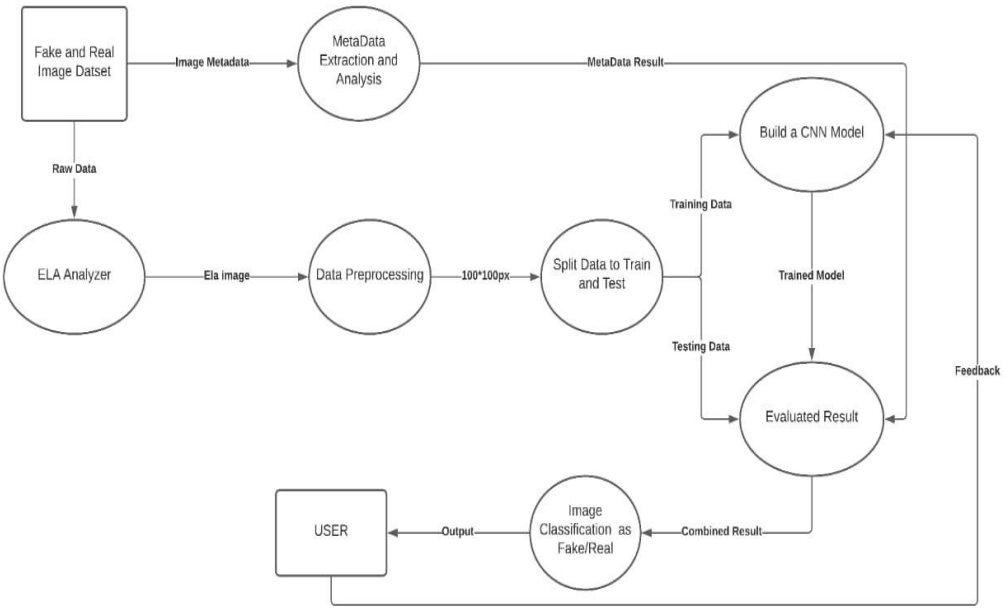

Figure 4.3.3.2 Level 1 DFD

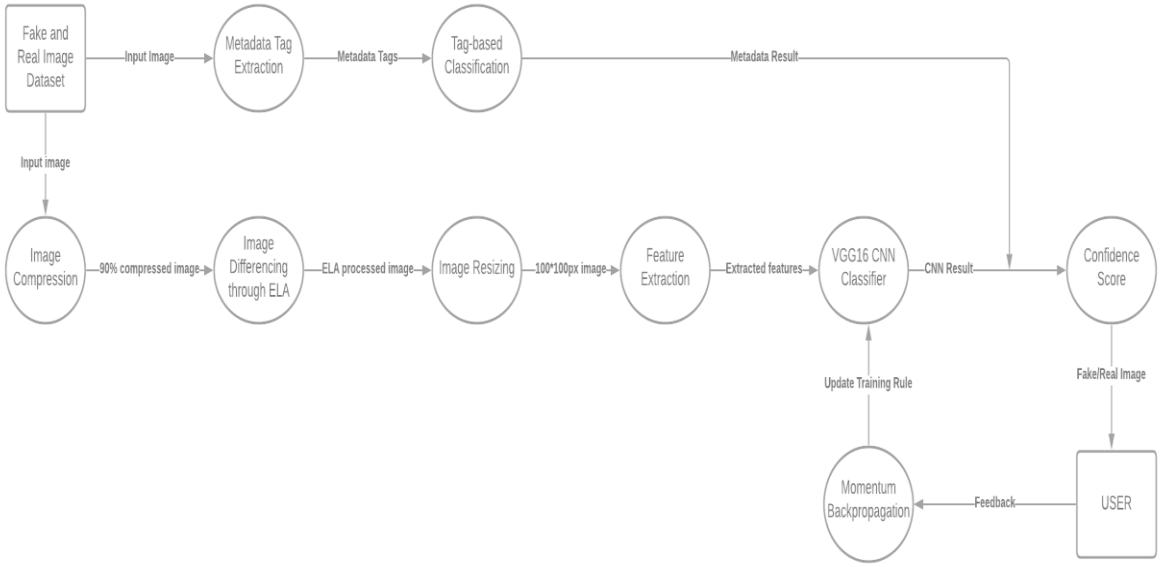

Figure 4.3.3.3 Level 2 DFD

4.4 Summary

In this chapter, the dataflow diagram clearly depicts how the proposed system allows the network to provide higher classification accuracies using unlabeled samples. In this chapter, the importance of DFD's and sequence diagrams is explained. The procedure to develop a DFDstarts with one DFD giving an overview of the system to be designed. This is called context diagram. The context diagram is expanded into a series of DFDs, each describing a specific function. This method of top down analysis and breaking down DFDs to give more and more detail is known as levelling. The various levels of DFDs of different modules are depicted in the figure. The DFDs help to better understand the system being implemented and eases future advancements related to the project.

CHAPTER 5

IMPLEMENTATION

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

Implementation is the phase of project where the theoretical outline is transformed into a working framework. Project implementation is the phase where visions and plans become reality. If implementation is not consciously arranged and controlled, it can bring about confusion and mystification.

5.2 System design

System design is a phase where the internal logic of each of the modules specified in high-level design is specified. In this phase further details and algorithmic design of each of the modules are specified. Other low-level components and sub-components are described as well. This chapter also discusses the control flow in the software in great detail by specifying the functionality, purpose, input and output of each function.

5.2.1 VGG16 CNN Architecture

The VGG16 architecture of CNN is used for pre-training the model as it is the most straightforward way of improving the performance of deep neural networks by increasing their size. The 16 in VGG16 refers to it has 16 layers that have weights. VGG16 has thus gained recognition in producing more precise classification results.
The input to cov1 layer is of fixed size 224 x 224 RGB image. The image is passed through a stack of convolutional (conv.) layers, where the filters were used with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, center). In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 pixel window, with stride 2.

Three Fully-Connected (FC) layers follow a stack of convolutional layers (which has a different depth in different architectures): the first two have 4096 channels each, the

third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks.

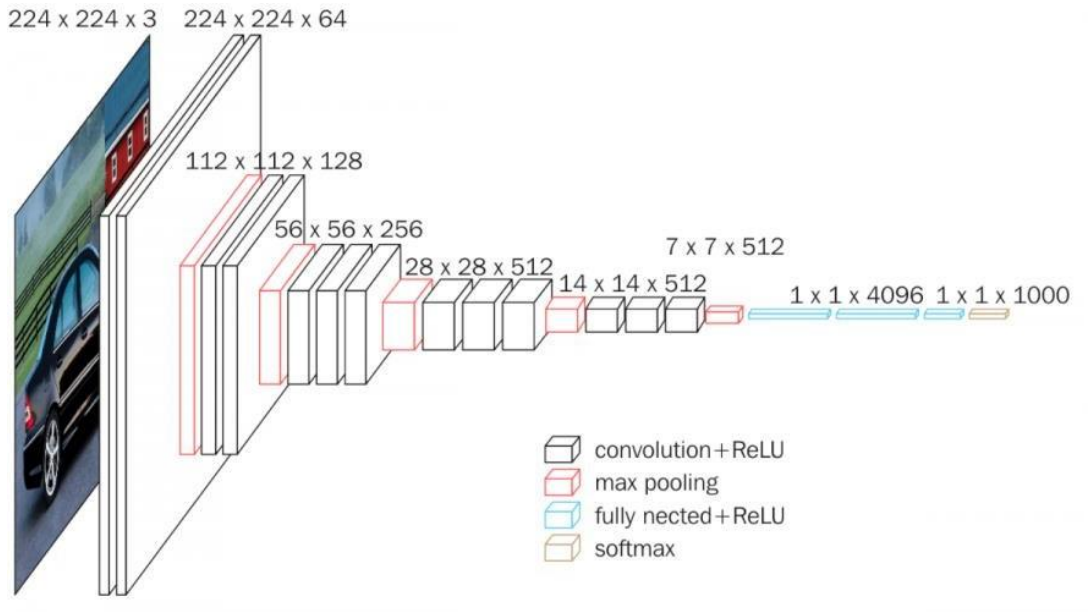

Figure 5.2.1.1 VGG16 CNN Architecture

All hidden layers are equipped with the rectification (ReLU) non-linearity. It is also noted that none of the networks (except for one) contain Local Response Normalisation (LRN), such normalization does not improve the performance on the ILSVRC dataset, but leads to increased memory consumption and computation time.

VGG 16 architecture of CNN is chosen because VGG 16 is perfect for training withminimal datasets. It results in a more precise classifier model.

5.2.2 Algorithm

The invention of the CNN in 1994 by Yann LeCun is what propelled the field of Artificial Intelligence and Deep learning to its former glory. The first neural network named LeNet5 had a very less validation accuracy of 42% since then we have come a long way in this field. Nowadays almost every giant technology firms rely on CNN for more efficient performance.

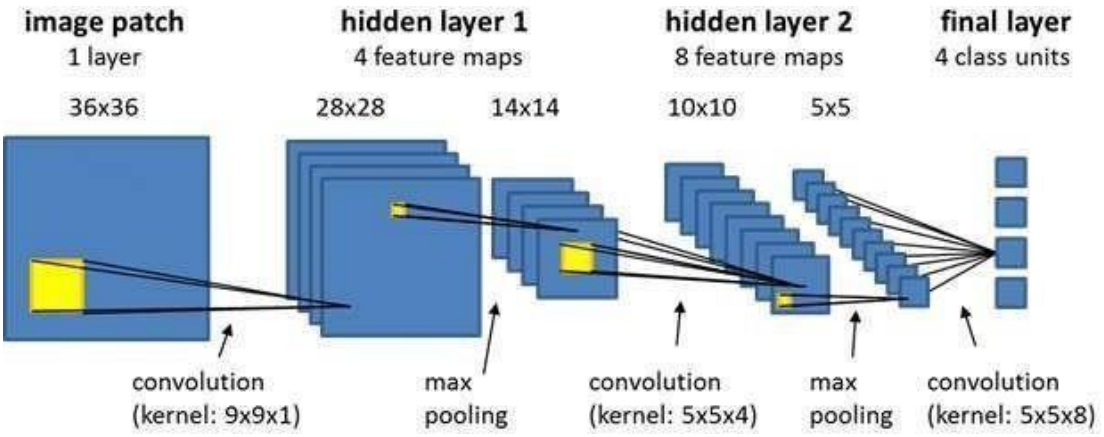

Figure 5.2.2.1 CNN Layers

A. Convolution layer :

This layer involves scanning the whole image for patterns andformulating it in the form of a 3x3 matrix. This convolved feature matrix of the imageis known as Kernel. Each value in the kernel is known as weight vector.

B. Pooling layer:

After the convolution comes to the pooling here the image matrix is broken down into the sets 4 rectangular segments which are non-overlapping. There are two types of pooling, Max pooling is an average pooling. Max pooling gives the maximum value in the relative matrix region which I taken. Average pooling gives the average value in the relative matrix region. The main advantage of the pooling layer is that it increases computer performance and decreases over-fitting chances.

Figure 5.2.2.2 Pooling Layer

C. Activation layer:

It the part of the Convolutional Neural Networks where the values are Normalized that is, they are fitted in a certain range. The used convolutional function is ReLU which allows only the positive values and then rejects the negative values. It is the function of

low computational cost.

5.3 Summary

This chapter provides the indepth view about the methods and algorithms used in different modules present in the project Forged Image Identification in Social media. Each of the modules consists of various other subtasks to perform the task efficiently. The steps involved in each module are represented with an independent flow chart. A brief review of the various algorithms and their working was presented in the chapter. This procedure can be studied in order to understand the algorithms themselves and provides the improved accuracy compared to the state of the art in terms of time complexity, which highly depends to the number of features used to identify a fake image.

CHAPTER 6

RESULT ANALYSIS

CHAPTER 6

RESULT ANALYSIS

6.1 Introduction

The following chapter contains the various experiments conducted using proposed methodology of the previous section. These experiments are conducted in a controlled environment where the parameter of each of the successive iteration is kept to the previous similar apart from the training set used for learning within the algorithms. This section is very important and informational as it provides us with empirical proof that the method adopted is better than the existing system in place. By analyzing the result of the experiments presented below, we can come to an informed decision on the effectiveness of the algorithm when compared to others used in same field.

Testing the process of trying to discover very conceivable fault or weakness in work product. It provides a way to check the functionality of components, subassemblies, assemblies and a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.1.1 Types of Tests

Unit Testing

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

Integration Testing

Integration Tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basis outcome of screens or fields. Integration tests demonstrate the components were individually satisfied by successful unit testing and also the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional Testing

Functional Tests provide systematic demonstration that the functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Organization and preparation of functional tests is focused on requirements, key functions or special test cases. In addition, systematic coverage pertaining to identify business process flows data fields, predefined process, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current test is determined.

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 rejecting.

Functions : Identified functions must be exercised.

Output : Identified classes of application outputs must be exercised.

Systems/ Procedures: Interfacing system or procedure must be invoked.

System Testing

System Testing ensures that the entire integrated software system meets the requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integrated test. System testing is based on process description and flows, emphasizing pre-driven process links and integration points. System testing takes, as its input, all of the “integrated” software components that have passes integration testing and also the software system itself integrated with any application hardware system.

White Box Testing

White Box Testing is a testing in which the software tester has knowledge of inner workings, structure and language of the software, or its purpose. It is used to test areas that cannot be reached from a black box level. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases as shown in fig 6.1. The tester chooses input to exercise paths through the code determine the appropriate outputs. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test. Through this method of test design can uncover many errors or problems; it has the potential to miss unimplemented parts of the specification or missing requirements.

Black Box Testing

Black Box Testing is testing the software without any knowledge of inner workings, structure or language of the module being tested. It is a testing, in which the software under test is treated, as a black box you cannot "see" into. Black box test, as most other kinds of test must be written from definitive source document, such as specification or requirements document. The test provides inputs and responds to outputs shown in Fig.6.1.

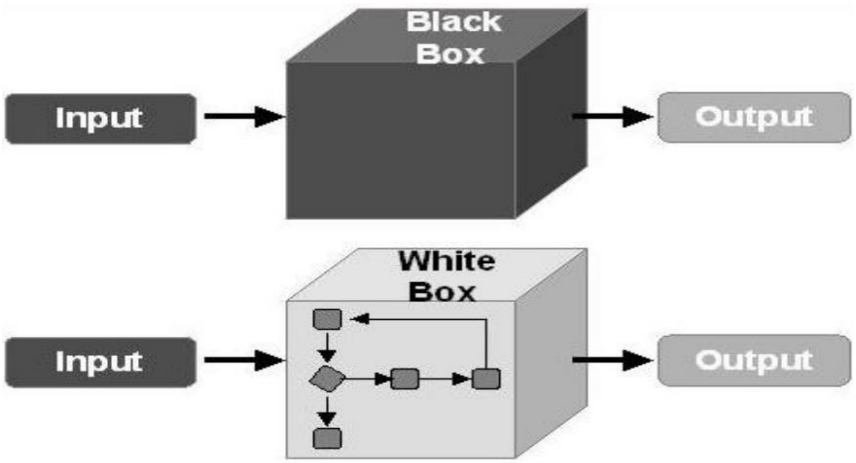

Figure 6.1 Black and White box Testing

6.2 Test Cases

A test case is a set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. A test case could simply be a question that you ask of the program. The point of running the test is to gain information, for example whether the program will pass or fail the test. Test case is the cornerstone of Quality Assurance whereas they are developed to verify quality and behaviour of a product.

Unit Testing is usually conducted as part of a combined code and unit test phase of software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases. Both the phases are very crucial and it is not recommended to skip any of phases. It is done after the completion of an individual unit before integration.

The strategy that is used to perform unit testing is described below:

Feature to be tested- The features to be tested, most importantly include the operation of individual component for the proper execution of entire program.

Items to be tested- The items to be tested include all the individual units or functions which collectively form the whole system. In case of unit testing the items to be tested are the Registration page, Login page, Uploading File, Fake Detection and Result analysis.

Purpose of Testing- The purpose of the testing is to check the unit functionality of the main project source.

In order to fully test that all the requirements of an application are met, there must be at least two test cases for each requirement: one positive test and one negative test. If a requirement has sub-requirements, each sub-requirement must have at least two test cases. Keeping track of the link between the requirement and test is frequently done using a traceability matrix. Written test case includes a description of the functionality to be tested, and the preparation required to ensure that test can be conducted. For an applications or system without formal requirements, test case can be written based on accepted normal operation of programs of a similar class. A formal written test case is

characterized by a known input and by an expected output, which worked out before the test is executed. The known input should test a precondition and the expected output should test a post condition.

6.2.1 Unit Testing for Fake Image Detection 1

Test Case	Unit Test Case 1
Name of Test	Unit testing of “Fake image detection 1”
Item being tested	Images
Description	Checks for authenticity of image based on metadata
Sample Input	A digitally modified image
Expected Output	Fake image
Actual Output	Same as expected output
Remarks	Successful

Table 6.2.1 Unit Testing for Fake Image Detection 1

Test Case	Unit Test Case 2
Name of Test	Unit testing of “Fake image detection 2”
Item being tested	Images
Description	Checks for authenticity of image based on metadata
Sample Input	An original image
Expected Output	Real image
Actual Output	Same as expected output
Remarks	Successful

Table 6.2.2 Unit Testing for Fake Image Detection 2

Test Case	Unit Test Case 3
Name of Test	Unit testing of “Fake image detection 3”
Item being tested	Images
Description	Checks for authenticity of image using neural network
Sample Input	A digitally modified image
Expected Output	Fake image
Actual Output	Same as expected output
Remarks	Successful

Table 6.2.3 Unit Testing for Fake Image Detection 3

Test Case	Unit Test Case 4
Name of Test	Unit testing of “Fake image detection 4”
Item being tested	Images
Description	Checks for authenticity of image using neural network
Sample Input	An original image
Expected Output	Real image
Actual Output	Same as expected output
Remarks	Successful

Table 6.2.4 Unit Testing for Fake Image Detection 4

6.3 Results

The results of the implementation and the inferences to be made from the testing results shall be listed in the following section. A real dataset was taken into consideration and

the program was run some multiple times for reviews. The result is forgery is detected and confidence level of the model is being displayed.

It contains the various experiments conducted using the proposed methodology of the previous section. These experiments are conducted in a controlled environment where the parameters of each of the successive iteration is kept to the previous similar apart from training set used for learning within the algorithms. This section is very important and informational as it provides us with empirical proof that the method adopted is better than the existing system in place. By analyzing the results of the experiments presented below, and informed decision on the effectiveness of the algorithm when compared to others used in same fields.

6.3.1 Snapshots

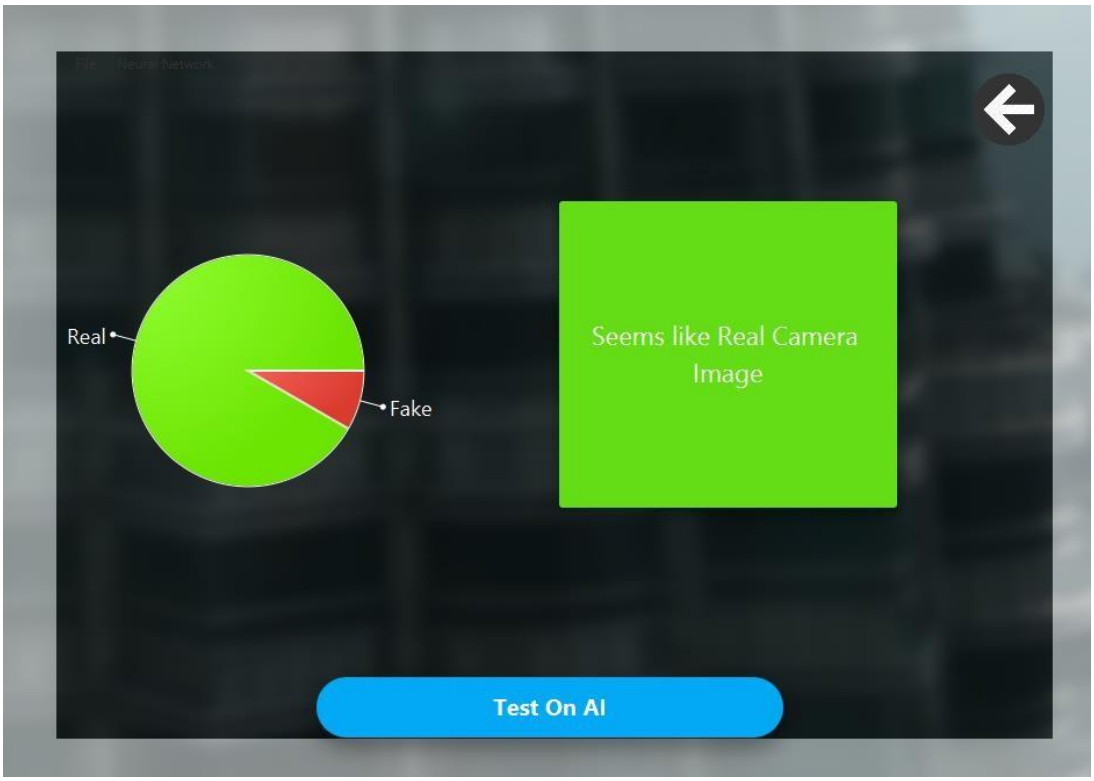

Fig 6.3.1.1 Metadata Analysis for Real image

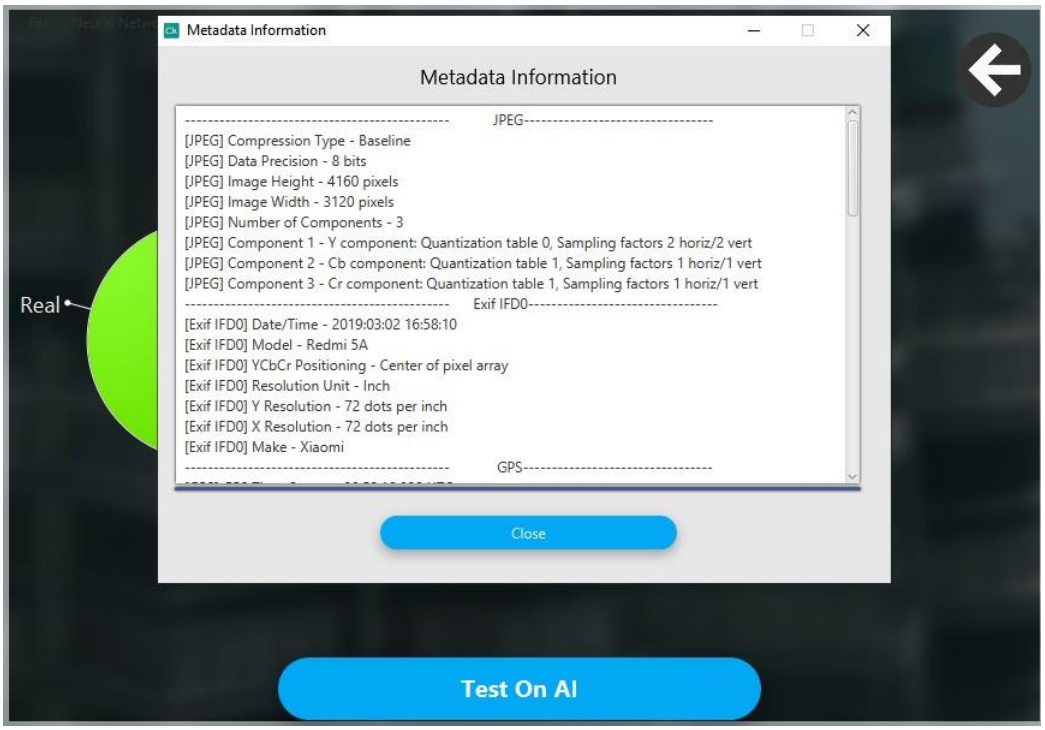

Fig 6.3.1.2 Metadata Information for Real image

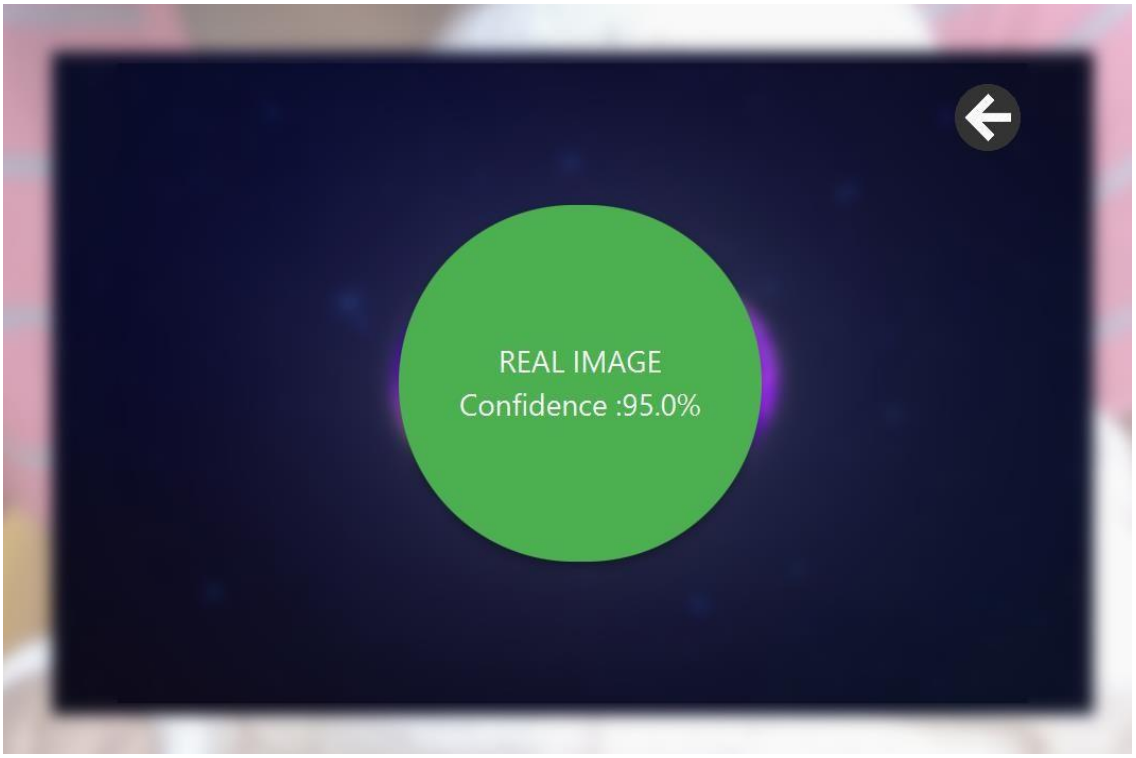

Fig 6.3.1.3 Neural network analysis for Real image

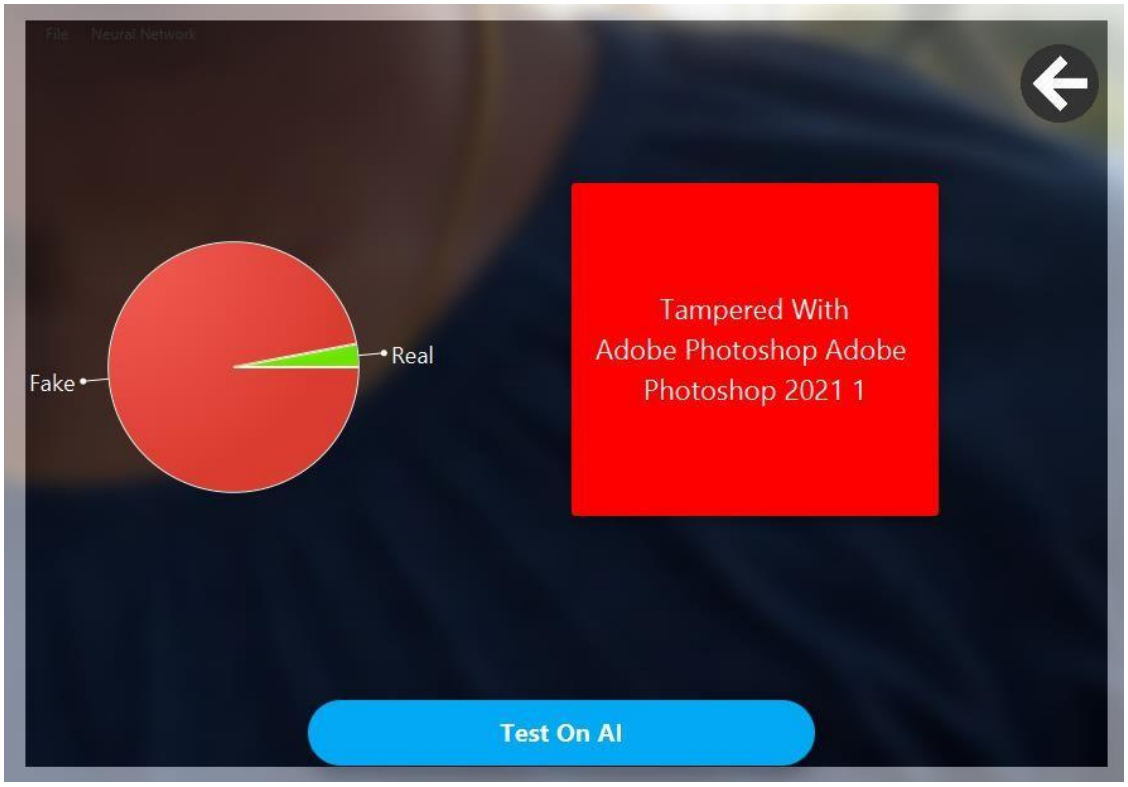

Fig 6.3.1.4 Metadata Analysis for Fake image

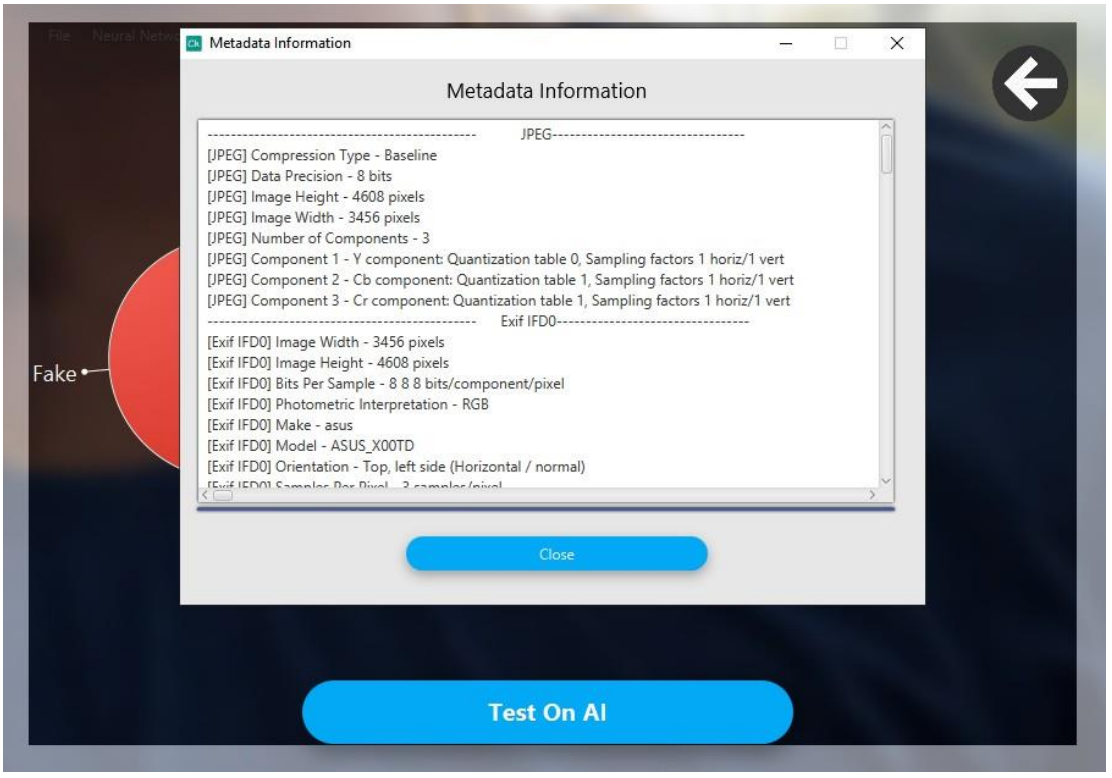

Fig 6.3.1.5 Metadata Information for Fake image

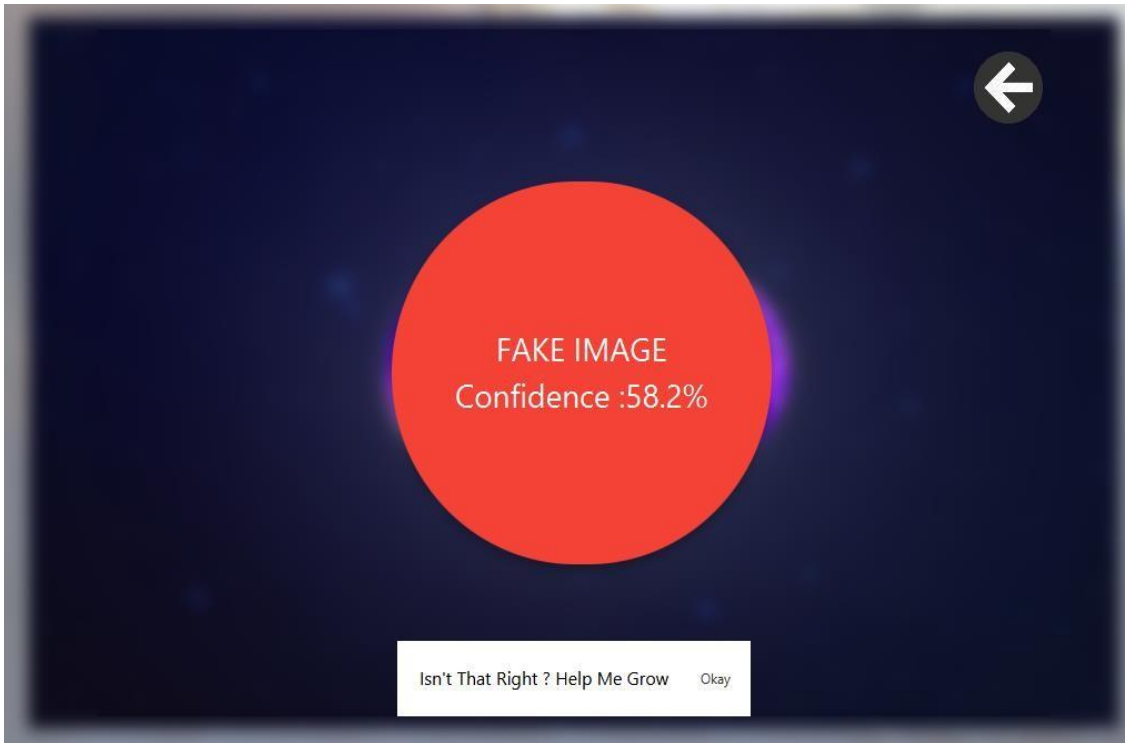

Fig 6.3.1.6 Neural network analysis for Fake image

6.4 Performance Evaluation

The models under comparison are:

- VGG16
- ResNet 50
- Inception

The above models were compared on various parameters such as the training time, training accuracy and the validation accuracy.

Model Training Time: The time costing of 100 epochs of each model are compared and VGG16 has got the least training time.

VGG16 - 900s

ResNet50 - 1100s

Inception - 3000s

Model Training Performance: The training accuracy of various models are assessed and is highest for VGG 16.

VGG16 - 0.9994

Resnet50 - 0.991

Inception - 0.97

Model Validation Performance: The validation of the various models are assessed.

VGG16 - 0.975

Resnet50 - 0.978

Inception - 0.95

TRAINING MODELS	TRAINING TIME (in seconds)	TRAINING ACCURACY	VALIDATION ACCURACY
VGG16	900	0.9994	0.975
ALEXNET	1100	0.991	0.973
INCEPTION	3000	0.97	0.95

Table 6.4.1 Model Performance

6.5 Summary

To conclude, this report has illustrated that an effective analysis can be performed on the application Forged image identification in Social media. There are two cases involved. Firstly, an original image is considered for which 2 level analysis is performed. When the image underwent metadata analysis the result displayed REAL along with the metadata information. The same image when tested on AI yielded resultas real with a confidence percentage of 95. Secondly, a digitally modified image is taken and the same procedure is followed. The image on undergoing metadata analysisyielded FAKE along with the metadata information. On performing neural network analysis for same image, results displayed was fake with a confidence percentage of 58. Metadata analysis has shown promising result in non-shared images. It is able to detect anomaly in all ‘photoshopped’ or ‘gimped’ images under a very small processing. It failed on images shared through WhatsApp, Google+ etc. Moreover, it became completely erroneous when images with manipulated metadata given. Neural network is trained with CASIA dataset [3]. The dataset contains 7491 real images and 5123 tampered images under varying sizes. All the images are

preprocessed to 100x100 pixels so that total pixel values to be fed into the neural network will be 30,000. From the dataset we have used 4000 real and fake images fortraining. Remaining images were used for testing of the neural network. Thus the useof this application in mobile platforms will greatly reduce the spreading of fake images through social media.

CONCLUSION

The proposed project successfully detects if an image is authentic or forged based on ELA and the deep learning techniques of CNN. The dataset comprising of tampered images and original images undergoes processing using ELA method. Neural network has been successfully trained using the error level analysis with 4000 fake and 4000 real images. The trained neural network is able to recognize the image as fake or real at a maximum success rate of 83%. VGG16 architecture of CNN is chosen because it is perfect for training with minimal datasets and results in a more precise classifier model. The use of this application in mobile platforms will greatly reduce the spreading of fake images through social media. This project can also be used as a false proof technique in digital authentication, court evidence evaluation etc. By combining the results of metadata analysis (40%) and neural network output (60%) a reliable fake image detection program is developed and tested.

FUTURE ENHANCEMENTS

There has been intense research on multimedia forensics, and great progresses. Nonetheless, many issues remain unsolved, new challenges appear by the day and, eventually, most of the road seems to be still ahead of us. For sure, the advent of deep learning has given extraordinary impulse to both media manipulation methods and forensic tools, opening new research areas. The presence of skilled attackers is a guarantee that no tool will protect us forever, and new solutions will be always necessary to cope with unforeseen menaces. With this premise, it is important trying to identify the most promising areas for future research.

As a future work, a better User interface can be deployed in order to improvise the look and feel of the application. Confidence score that proves the authenticity of a particular image can be increased. Alternative architecture such as GoogleNet can be deployed for training the model, as it is faster than VGG16 model that is currently being implemented. Higher robustness should be pursued by various means. To deal with the rapid advances in manipulation technology, deep networks should be able to adapt readily to new manipulations, without a full re-training, which may be simply impossible for lack of training data or entail a catastrophic forgetting phenomena.

REFERENCES

[1] Chih-Chung Hsu, Chia-Yen Lee, Yi-Xiu Zhuang, "Learning to Detect Fake Face Images in the Wild" , IEEE 2018 , International Symposium on Computer, Consumer and Control (IS3C).

[2] Yuanfang Guo, Xiaochun Cao, Wei Zhang, Rui Wang, Member, "Fake Colorized Image Detection ", Submitted To IEEE Transactions On Information Forensics And Security, 2019.

[3] Savita Walia & Krishan Kumar, "Digital image forgery detection: a systematic scrutiny ", Australian Journal of Forensic Sciences, 2019.

[4] Muhammed Afsal Villan, Johns Paul, Kuncheria Kuruvilla and Prof. Eldo P Elias "Fake image detection using Machine Learning", International Journal of Computer Science and Information Technology & Security,2018.

[5] Khurshid Asghar, Xianfang Sun4, Paul L. Rosin4, Mubbashar Saddique2, Muhammad Hussain3, "Edge–texture feature-based image forgery detection with cross-dataset evaluation", Springer-Verlag GmbH Germany, part of Springer Nature 2019.

[6] B. Bayar and M. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," in ACM Workshop on Information Hiding and Multimedia Security, 2018.

[7] Khurshid Asghar, Zulfiqar Habib & Muhammad Hussain "Copy-move and splicingimage forgery detection and localization techniques: a review ", Australian Journal ofForensic Sciences, 2019.

[8] Y. Lui, X. Zhu, X. Zhao, and Y. Cao, "Adversarial learning for constrained image splicing detection and localization based on atrous convolution," IEEE Trans. Inf. Forensics Security, vol. 14, no. 10, pp. 2551–2566, 2019.

[9] J. Fridrich, D. Soukal, and J. Luka´s, "Detection of copy-move forgery ˇ in digital images," in Proc. of the 3rd Digital Forensic Research Workshop, 2019.

[10] Y. Zheng, Y. Cao, and C.-H. Chang, "A PUF-based data-device hash for tampered image detection and source camera identification," IEEE Trans. Inf. Forensics Security, vol. 15, pp. 620–634, 2020.