Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Software Department

An Ensemble Model for Image Forgery Detection based on Xception Networks

A Thesis

Submitted to the Council of the College of Information Technology for Postgraduate Studies of the University of Babylon in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology – Software

By

Ihsan Sahib Abd Alshahid Muhammed

Supervised by

Prof. Dr. Tawfiq Abedulkhaleq al-assadi

2022 A.C. **1443 A.H.**

(وَقُلْ عَسَىٰ أَن يَهْدِيَنِ رَبِّي لِأَقْرَبَ مِنْ هَـٰذَا رَشَدًا)

(٢٤ الكهف)

صَدَقَ اللهُ العَظيمُ

In the name of Allah, Most Gracious, Most Merciful

(And say, Perhaps my god will guide me to what is nearer than this to right conduct**)**

(24 Alkahf)

Great truth of God

Declaration

I hereby declare that this dissertation entitled "**An Ensemble Model for Image Forgery Detection based on Xception Networks**", submitted to the University of Babylon in partial fulfillment of requirements for the degree of Master in Information Technology \ Software, has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for experts and summaries whose source is appropriately cited in the references.

Signature:

Name: Ihsan Sahib Abd Alshahid

Date: / / 2022

Supervisor Certification

I certify that the thesis entitled "**An Ensemble Model for Image Forgery Detection based on Xception Networks**" was prepared under my supervision at the department of Software / College of Information Technology / the University of Babylon as partial fulfillment of the requirements of the degree of Master in Information Technology - Software.

Signature:

Supervisor Name: Prof., Dr. Tawfiq Abedulkhaleq al-assadi

Date: / / 2022

The Head of the Department Certification

Given the available recommendations, I forward the thesis entitled "An Ensemble Model based on Deep Learning Xception Networks for Image Forgery Detection" for debate by the examination committee.

Signature:

Assist. Prof. Dr. Ahmed Saleem Abbass

Head of Software Department

Date: / / 2022

Certification of the Examination Committee

We, the undersigned, certify that (Ihsan Sahib Abd alshahid) candidate for the degree of Master in Information Technology - Software, has presented his thesis of the following title (An Ensemble Model for Image Forgery Detection based on Xception Networks) as it appears on the title page and front cover of the thesis that the said thesis is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on:

Signature:
Name:
Title: Professor
Date: / / 2022
(Chairman**)**

Signature:
Name:
Title: Assistant Professor
Date: / / 2022
(Member**)**

Signature:
Name:
Title: Assistant Professor
Date: / / 2022
(Member**)**

Signature:
Name:
Title: Professor
Date: / / 2022
(Member**)**

Signature:
Name:
Title: Professor
Date: / / 2022
(Dean of Collage of Information Technology**)**

Dedications

This work is dedicated to…

The martyrs of Iraq of all sects and nationalities.

My mother,

I ask Allah to protect you from all evil and I will not disappoint you.

My father,

I will always be your son who is proud of you, and I will not disappoint you, my beloved father.

My beloved brothers, sisters, and wife

Who stands by me when things look very difficulties

My supervisor,

Who stands and gives me his effort and suggestions to pass the difficulties and achieve this work.

Acknowledments

Thanks and appreciation

Foremost, I am highly grateful to Allah (God) for His unlimited blessings that continue to flow into my life, and because of You, I made this through against all odds.

To my supervisor, Prof. Dr. Tawfiq Abedulkhaleq al-assadi: I feel highly indebted to you. I am deeply grateful for your suggestions on this topic, and without your support, comments, and guidance, it would be difficult to finish this work. I will not forget the contribution and assistance of Prof. Dr. Esraa Hadi Ali in accomplishing this work.

To my wonderful parents: Words cannot express my appreciation for you, the best mother and father. I would like to give you all my thanks and love for your guidance, advice, prayers, and endless support.

Thanks to my brothers and my wife for the endless support. Words cannot express my appreciation for them, and I find in my heart nothing but gratitude for what they have given me throughout my studies.

To my friends: Thank you all for your belief in me and for your continued support, encouragement, and cooperation at all times.

To my College Staff: Thank you all for your efforts, support, and cooperation at all times.

Special thanks to the staff of the Babylon Governorate Education Directorate for the great assistance they provided me.

Finally, I would like to thank all the kind, helpful and wonderful people who helped me in completing this work and apologize to them for not being able to mention them by name here, but they are in my heart.

Ihsan Sahib Abd alshahid Muhammed

Abstract

Image forgery of the images can be defined as tampering or falsely or imitation altering of a photo, therefore digital image forgery nowadays is one of the most problems that need to be a solution, especially the most digital platforms deals with the data using the images, at the same time, the image manipulation softwares were developed, on other hand, the new technologies of artificial intelligence and deep neural networks of Generative Adversarial Networks(GANs) can create images by computer. Where the new Architectures of GAN networks such StyleGan could make fake images and videos that cannot be detected by the naked eye. Therefore the systems of GAN images forgery detection are important to overcome this problem but these systems suffer from a lack of high performance Because of the high precision and high consistency of images generated from GANs, which is the aim of this proposed methodology, in other words, how to detect fake images better than humans.

The proposed methodology consists of five major phases as follows: pre-processing, preparing the dataset, extracting image features, decision, and localization. The first phase is preprocessing which includes resizing the images, converting the image color system from RGB to YCbCr color system, and making another dataset for image partitions. The second phase is preparing the dataset includes splitting the original images dataset and images partitions dataset of into training, validation, and testing. The third phase is features extraction based on two deep Convolutional Neural Networks (CNNs) to represent the image as a vector of features by learning the networks on the features of the real

and fake images to make predictions to classify the images in the testing phase based on the learned features. Then training the localization Convolutional Neural Network (CNN) to learn the network on the features of the partitions of the images to detect the faked parts of the fake images. The fourth phase is making the decision in the testing phase based on Average Ensemble for each prediction of the two classification networks to classify the image as real or fake, where the ensemble method improves the performance of the system. The fifth phase is localization as a postprocessing step after the decision on whether the image is fake, to detect and frame the faked partitions of the fake image using the learned localization network.

This thesis adopted “140k Real and Fake Faces” dataset, which it contains 70K of real images, and 70k fake images generated using StyleGan network, where most of the fake images are difficult to detect with the human vision system. Lastly, the classification accuracy for the proposed methodology has achieved %99.99 in addition to %92.13 for the localization process. Finally, after applying the learning enhancement idea to the networks, the two image classification networks achieved an accuracy of 100% and the localization network achieved 92.20%.

Declaration Associated with this Thesis

Some of the works presented in this thesis have been published or accepted as below.

First Paper:

Title: Deep Learning for Image Forgery classification based on Modified Xception Net and Dense Net

Author: Ihsan Sahib Abd Alshahid, and Tawfiq Abd Alkhaliq AlAsady

Journal: AIP Conference Proceedings,

Volume:

Number:

Pages:

Year: 2022,

Publisher: AIP Conference Proceedings.

List of Contents

List of Figures

List of Tables

List of Algorithms

Table of Abbreviations

Abbreviation	Meaning
Acc	Accuracy score
AI	Artificial Intelligence
ANN	Artificial neural networks
BN	Batch-Normalization
BP	Back-Propagation
CM	Confusion Matrix
CNN	Convolutional Neural Network
DBN	Deep Belief_Network
DCNN	Deep Convolutional Neural Network
DNN	Deep Neural Network
FN	False Negativ
FP	False Positive
GAN	Generative adversarial network,
GPU	Griphics Processing Unit
Loss	Loss function score
MSE	Mean Square Error
MTXception	Modified Trimmed non trained Xception Net
ProGAN	Progressively Growing GAN
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Units
SGD	Stochastic Gradient Descent
SLNetwork	Segments Localization Network
SNNs	Synthetic Neural Networks
Std	standard deviation
Tanh	Hyperbolic Tangent
TN	True Negative

TP	**True Positive**
Trn	**Training set**
Tst	**Testing set**
Val	**Validation set**

Chapter One

Introduction

1.1 Overview

The “forgery” definition according to Merriam -Webster Dictionary defined as “The crime of falsely and fraudulently making or altering a document”. Where Image forgery of the images can be defined as the tampering or falsely and imitation altering of a photo. The history of the fake images can be traced back to the 1840's when Hippolyte Bayrad created the first fake photograph Figure (1.1), in which he appeared to commit suicide [1]. Another fake photo appeared in the 1860s, in which the head of Abraham Lincoln (President of the United States) was replaced by the corpse of rival politician John Calhoun [2] Figure (1.2). Many other examples can be found in History.

Figure 1.1 First fake image [1].

Figure 1.2 President Abraham Lincoln and Southern politician John Calhoun [2].

The problem of detecting false images required a wide range of analyzes and examinations by researchers after the advent of digital photography and the development appeared in image manipulation programs of computers until fraud became an easy matter for anyone who possesses the ability to use such programs like Adobe Photoshop, Pixir and Gimp and use these images for scandalous

Previously, forensic techniques were mainly concentrated on identifying particular modifications. It was a popular approach for analyzing the process of a single operation and the model of statistical characteristics can discover the trace of specific forging [3], However, because of the limitations of the statistical analysis approach of each attack, therefore the different types of image manipulation can change the statistical fingerprint of various operations [4]. As a result, the reliability of the images is becoming weak and is not certified safely. At the same time, in recent years, most platforms use images As a means of dealing and transmitting the information.

The topic of computer vision has grown to include everything from collecting raw data, extracting image patterns, and analyzing data [5]. It combines artificial intelligence, image processing, pattern recognition, computer graphics concepts, techniques, and ideas [6]. The important tasks in computer vision considered Extracting information on events or describing from input scenes (digital images) and conducting feature extraction are two of the most.

Image processing and pattern recognition are integrated in computer vision [7]. A Computer Vision concept yields image understanding as a result. The advancement of this field based on the adaptation of human vision's ability to process information. In contrast to computer graphics, computer vision is the science of extracting information from images. Whether its enhancement the image quality or recognize the image, the growth of computer vision is controlled by the computer technology system. On basic

techniques, there is some overlap with Image Processing, and some authors use both words interchangeably [8]. In order to improve perception and decision-making, image processing has become more important and basic in many information access systems, for example image generation, image compression, encryption, image de-aliasing, ultra-high-resolution, image classification, segmentation, and object recognition, image annotation, image retrieval, and other image processing techniques are commonly used in pattern recognition based on machine learning techniques [9].

Artificial Intelligence (AI) and deep learning approach using the Convolutional Neural Networks (CNN) applied to medical imaging has the potential to be the most common diagnostics technology since the introduction of digital imaging. Most academics expected that deep learning applications will overtake humans in the next 15 years, and that intelligent robots will not only do most diagnostics, but will also help predict sickness, prescribe drugs, and lead therapy, pathology, cancer detection [10].

On the other hand, the Deep learning approach used for image manipulation in a Generative Adversarial Network (GAN), which produce computer-generated images of multi manipulation images, most the network generations focused on the faces side more than other sides in life, where the face is the most identifier part of the human, where the modern generations of the generative adversarial network can be produced images of the faces of non-existent people, and it became difficult to detect forged images by the naked eyes of humans as shown in Figure (1.3).

Figure 1.3 Sample of computer-generated images by StyleGan Network.

1.2 Research Problem

Due to the recent developments in forgery techniques of GAN networks and difficulties in the used techniques to detect the fake images of Deep Fake. Therefore the main problem addressed in this thesis is the detection of Deep Fake images forgery or in other words how to detect fake images better than humans.

1.3 Related Works

Most previous studies concerning forgery detection focus on Keypoint-based multi-scale analysis performed on the obtained key-points. Alternatively, may use a block-based matching approach, as well as the various properties of the picture areas. Recently, a number of works appeared that have relied on machine learning and learning methodologies.

1. H. S. Shad et al (2021). Present comparative research by implementing several methods to detect styleGAN images to make a analysis. the comparative study of using convolutional neural networks (CNN) to identify real and deepfake pictures, eight different CNN models are trained. Three of DenseNet architecture (DenseNet121, DenseNet169, and DenseNet201); two were trained using the VGGNet architecture (VGG16, VGG19); one was with the ResNet50 architecture, one with the VGGFace, and one custom model. Amongst all the models, VGGFace performed the best, with 99% accuracy. Besides, ResNet50

obtained 97%, 96% from the DenseNet201, 95% from the DenseNet169, 94% from the VGG19, 92% from the VGG16, 97% from the DenseNet121 model, and 90% from the custom model [11].

2. N.-K. Ngo and X.-N. Cao proposed proposed a deep learning approach for the U-net classifier to detect the fake images of StyleGAN, by determining if the image is real or fake based on integrating local and global information about the image on the pixel-level where The greatest accuracy of the model was %98.43 [12].

3. J. Sharma and S. Sharma (2021) proposed simple CNN to classify the deep fake of face images generated StyleGAN of contain of tow convolutions and four fully connected layers which get an accuracy for training of %99 and testing of %94,41[13].

4. Zhengzhe Liu and Xiaojuan Qi. (2020) proposed an improvement on ResNet called GramNet to detect the fake images based on the difference gap between a texture of fake and real images by adding Gram block to ResNet at the input image and before each layer of downsampling. it consists of two convolution, batchNormalization, and Rectified LinearnUnit (Relu) Layers to capture the texture of the images and compute the differentiation between real and forged images. The accuracy of this work is 98.96% [14].
5. Chih-Chung Hsu et.al (2020). Proposed a deep learning-based method for detecting fake images using pairwise learning based on the idea of contrastive loss, by generating real–Fake images pairs. Then, using the DenseNet developed and reduced to a two-stream network of Siamese architecture to allow information in pairs as inputs. Finally, training the network based on the pairwise learning to distinguish the difference

between the real and fake images features. The accuracy of this work is 99.1% [15].

6. Minh Dang , et.al (2020) proposed an Extreme gradient boosting tree, which is a learning approach to create many classifiers where a final classifier is a group of weak classifiers. At first, a simple classifier is trained on the training dataset, and the classified sample improperly are recorded. Then, the next classifier is trained to repair the erroneous predictions of the first classifier. After that, many weak classifiers are constructed to fix the errors of predictions that previous classifiers. The accuracy of this work is 83% [16].
7. Philip H. S. et.al. (2020) proposed DeepFake Face Image Detection based on improved VGG Convolutional Neural Network. It has an enhanced VGG network and named NAVGG to detect fake faces of DeepFake, based on the noise and image enhancement. The SRM filter (Steganalysis Rich Model) is utilized in the image to high spot the noise features, then the noise map is enhanced to weakening the face features to learn more about manipulation artifacts that may be unseen in RGB channels. Finally, these images are sent via the network, which is used to train and determine whether the image is fake or not. The accuracy of this work is 98.96% [17].
8. Z. Mi, X. Jiang, et.al. (2020) proposed a method to detect the images generated by Gan Network based on learning the difference between Gan image texture pattern and actual images using CNN's deep learning approach and self-attention mechanism with two dimensions by dragging the aggregated information into the images. This information was used to detect Gan Generated images. The accuracy of this work is 99.3% [18].
9. Lakshmanan Nataraj, et.al. (2019) proposed an approach to detect forged images generated by the GANs, where a mixture of matrices of

recurrence and deep learning was used. Using a deep (CNN) framework, extract co-occurrence matrices on color three channels in the pixel domain and train deep CNN model. The accuracy of this work is 93.42% [19].

10. Peisong He, et.al. (2019) proposed using different color systems (YCbCr, HSV, and Lab) to extract different deep representations from a convolutional neural network (CNN), and use a random forest classifier for determining the reality of an image. The accuracy of this work is above 99% [20].

The related works are summarized in the below in Table (1.1).

the work presented in this thesis achieved an accuracy of 99.99 which outperformed the other works which adopt same or another dataset of earlier GAN networks

Table 1.1. Related works summary

No. of works	Processing method	Dataset	results
1	Comparing for several CNN architectures	140k Real and Fake Faces	VGG19 %94 VGG16 %92 VGGFace %99 DenseNet169 %95 DenseNet201 %96 DenseNet121 %97 ResNet50 %97 Custom CNN %90
2	U-net classifier	140k Real and Fake Faces	%98.43
3	CNN	140k Real and Fake Faces	%94.41

4	**GramNet (Gram block + ResNet)**	**StyleGAN, PGGAN, DRAGAN, DCGAN, StarGAN, CelebA-HQ, FFHQ, CelebA**	**%98.96**
5	**Pairwise Learning**	**DCGAN, WGAP, WGAN-GP, LSGAN, PGGAN**	**%99.1**
6	**Extreme gradient boosting classifiers tree**	**MANFA PGGAN,**	**%83**
7	**NA-VGG(SRM filter + VGG CNN)**	**DeepFake face image**	**%85.7**
8	**CNN + self-Attention mechanism**	**CelebA-HQ, PGGAN, LSUN-Bedroom**	**%99.3**
9	**CNN(Co-occurrence Matrices)**	**CycleGAN, StarGAN**	**%93.42**
10	**Multi deep CNN with multi color**	**PGGAN, CelebA**	**Above %.99**

1.4 Thesis Motivation

Many researchers motivate to solve the problem of images forgery detection based on GAN network where the new generations such StyleGAN can lead to an assortment range of losses and social troubles for example:

1. Image forgery and deep fake may mislead the public opinion by targeting political personages, presidential candidates, or religious personages by creating fake images of them and saying things that were not spoken in their tongues.
2. Defaming celebrities and people and exposing them to scandal when posting a fake images or video about them for the purpose of fraud or extortion

3. Presenting them in a law court as evidence, may Doing mislead the court could cause Unfair judgments.

1.5 Aim of Thesis

The aim of this thesis presents a better solution for the problem of image forgery produced by StyleGAN network, through the following steps:

- Building a system with effective classification accuracy for the problem of the image forgery undetected by human vision system.
- Develop a model for Extracting the image features using Separable CNN with high accuracy, minimal complexity, fewer parameters, and a shorter implementation time.
- Detect and localize the forged portions of the fake image.
- suggest an idea of enhancing learning for the proposed models to get better accuracy with less time, effort, and resources.

1.6 Thesis Contributions

The important contribution in this thesis is to present an effective model for detecting the forged images which generated using StyleGAN network and localizing the forgery in the image.

1.7 Challenges of the Research Problem

The research has addressed the issue of image forgery based on GANs, and it also has drawbacks and limitations that make it impractical and ineffective. The main challenges of this technique are as follows:

1. High consistency of forged images in the latest generations of Gan networks compared to real images.
2. the forged image (faked image) contains many and overlapping manipulations where faces are swapped and merged the features of several faces styles are combined in the fake image.

3. Dimensionality, reducing the dimensions of the image may result in the loss of some details that can help to detect and localize the forgery in the fake image.
4. Computer resources where deep learning approach needs large data with high dimensions for training to obtain better accuracy and this requires more resources.
5. Multi-face images some images contain part of another face this may affect the final decision.

1.8 Thesis organization

This thesis includes five chapters and each chapter begins with an overview and then some details about the problem that this thesis addresses, as follows:

The first chapter explains a general introduction to the problem and its history, in addition to the reasons and Aim of the thesis. The content of the other chapters are as follows:

Chapter Two: Presents a comprehensive description of the main principles of "theoretical background", which were used later in this thesis.

Chapter three: Under the title of “The proposed system”. Explains the proposed system and the algorithms used in it.

Chapter Four: This chapter clarifies the results of the proposed system and research experiments. It also discusses the evaluation of the system's performance.

Chapter Five: mentioned precisely the conclusions, it includes what was discovered in this thesis, and potential future research directions to improve this work.

Chapter Two
Theoretical Background

2.1 overview

Deep learning is the most interested field in machine learning, neural networks and now deep learning presents solutions that are the most efficient to a wide range of problems in an image, audio, and text recognition, time series issues, video analysis, and natural language processing. Because deep learning is a smarter way than shallow algorithms of machine learning [21].

Nowadays, deep learning has attained wonderful performance as a result of recent swift advances in deep convolutional neural networks. Specifically, in the field of computer vision, introduced non-classic and effective solutions for a number of difficult problems that rested for a long-without solving, due to this promising performance, it is gaining more and more attention and is being applied widely in computer vision for several tasks such as object detection and recognition, object segmentation, pedestrian detection, aerial imagery registration, video processing, scene classification, autonomous driving, and robot localization as well as medical image related applications [22].

2.2 Fake or Forgery Detection

Since more aspects of life nowadays is in a digital world where all kinds of advancements are possible, and the use of images in of daily lives is growing by the day, the motivation to manipulate images is growing. This form of picture forgery is becoming more popular by the day [23].

Images edited using digital tools go through many processing stages and are so photorealistic that human eyes never identify the forgery in the image. As a result, manipulated images are emerging at a growing space, causing people to lose faith in visual content. As a result, the image's authenticity isn't taken for granted. Technology has been innovated to verify the originality of image details with the creation of forgery software [24].

2.2.1 Image Forgery Techniques

Images forgery methods are divided into two types, i.e. active methods and passive methods, which consist of a variety of techniques, as shown in Figure (2.1). Overlapping forgery is containing the image on multi forgery techniques.

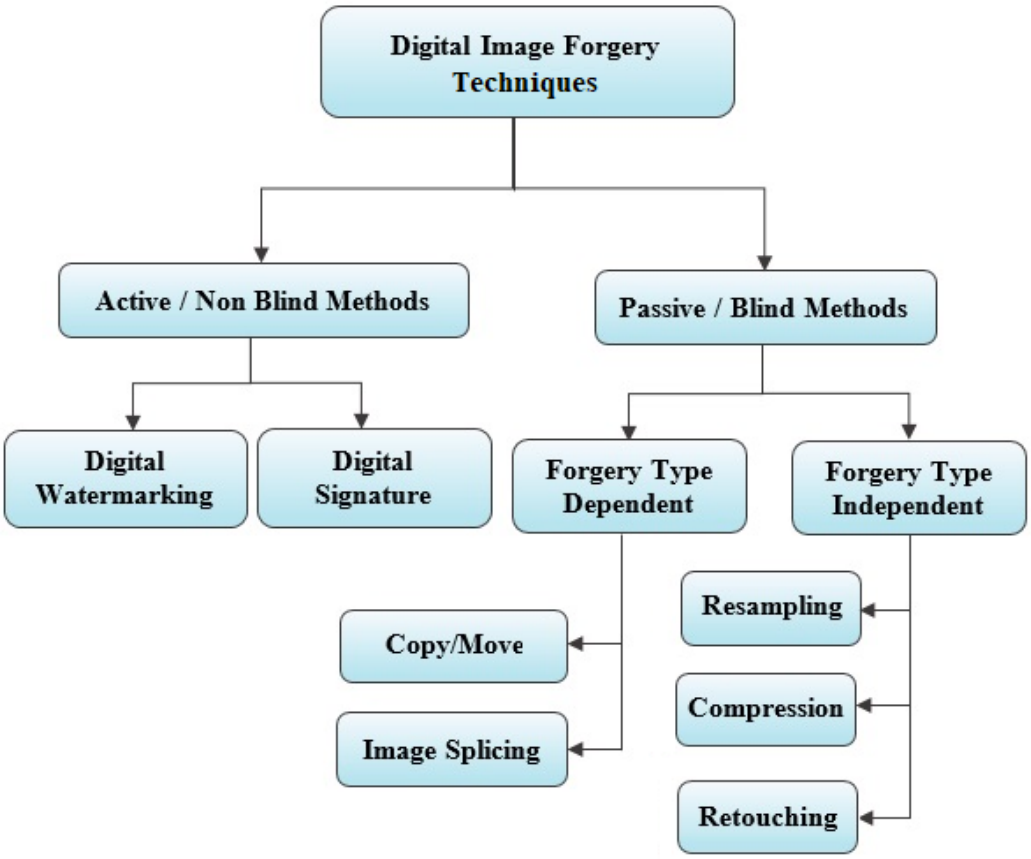


Figure 2.1: Classification of Image Forgery techniques [25].

a) Active Approach

Active forging techniques require some prior information of the image, which might be included in it.

- Watermarking and steganography are two aspects of the active approach. These are applied at the time of acquisition of the image. To signify the authenticity of the digital image, a special hardware implementation is required. Digital watermarking is the act of hiding a critical message in a photograph. Steganography is the method of disguising a mark or message in an image in order to retain its copyright at the time of acquisition [26].
  - A digital signature is a mathematical technique for ensuring the validity and legitimacy of an electronic message, software, or text. It's the digital version of a handwritten or sealed signature [26].

b) Passive Approach

In image processing methods, passive image forensics is typically a major challenge of Researchers because of the absence of prior information about the image. There isn't a single system that can handle all of these circumstances, but there are several that can each detect a unique forgery in its own way. To find image tampering, the passive tampering detection stream analyzes the raw image using several statistics and semantics of image content to locate image tampering. Unlike active techniques, no construct is embedded in or associated with the image for protection [23]. The types of passive image forgery as follows:

1. Independent Forgery: the forgery is the manipulation in some of the characteristics of the same image. The main types of independent forgery are as follows:

- Resampling

  Resampling the 2-Dimension image has a complexity more than linear or a 1-Dimension correlation. Where the 2-Dimension image is interpolated by the vertically and horizontally directions together. The method of resampling an image has three steps: up-sampling vertically and horizontally, interpolation in two directions, and down-sampling also in two directions. In the end, each pixel color correlates harmonically in the resampled image with its around pixels [27].

- Image compression

  The most used image compression in the cameras is JPEG format, besides, it is very common and popular with manipulators, for example, when alters the manipulator with the content of a previously compressed image with JPEG format, then resaving the image with the same format. The untampered parts is appear compressed two times and the tampered part of the image appears compressed one time because the JPEG artifacts of the first compression have been destroyed after manipulation, but is difficult to distinguish because the JPEG format considered a lossy compression [28][3].

- Image Retouching

  Image retouching is a form of digital image forgery that is considered less dangerous than other forms. The original image is not substantially changed in the case of image retouching, although some aspects of the original image are enhanced or reduced. Photo editors in magazines widely use this method. This form of image forgery can be seen on almost all magazine covers, which use this technique to enhance certain aspects of an image to make it more appealing. In reality, such enhancement is ethical in the first place as shown in Figure (2.2), but on the other hand, in some places used for unethical purpose [29].

a b

Figure 2.2 Retouching Forgery a) Original image b) retouched image [29].

2. Dependent Forgery: One image can be tampered with one or many images can be merged (image splicing) to create a persuasive composite image. The types of dependent forgery are as follows:

- Image splicing or photomontage

This forgery image generation process is more violent than retouching. Splicing an image is a simple process that involves copying and pasting portions from the same or separate sources. This approach entails gluing photographs together using digital programs such as Photoshop to produce a paste-up. The Image Splicing technique involves combining two or more photographs to create a fake image. Several well-known news reporting cases in which faked images were used are examples. Figure (2.3) shows how to create a forged image by copying a spliced section of the source image into a target image, creating a composite picture of scenery known as a forged image [23].

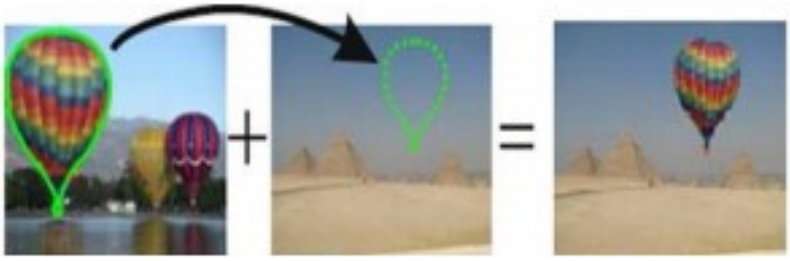
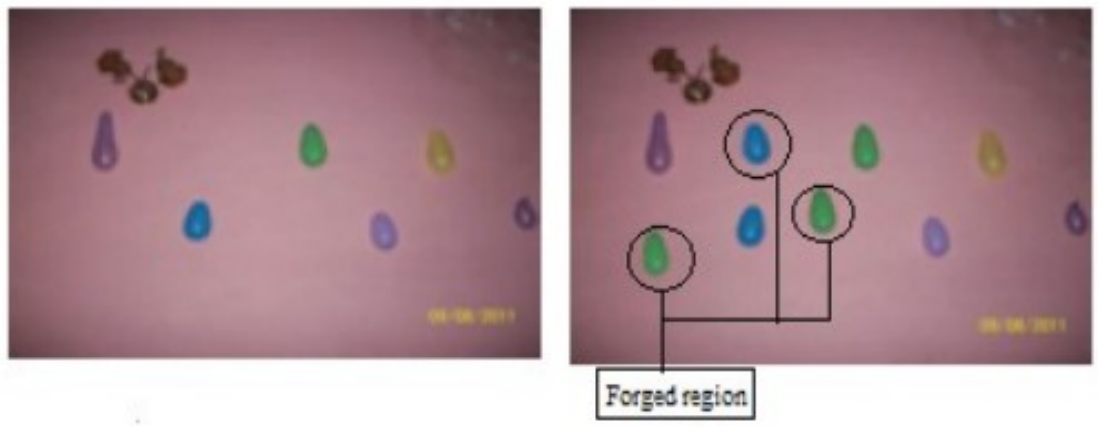


Figure 2.3: Example of Image splicing [23].

- Copy-Move Attack

One of the most challenging and often used types of image altering is copy-move forgery. To add or delete content, this approach used to cover a piece of the image, by copy a component from an image then pasted into different part or parts in same image, in the Copy-Move image processing method. A copy-move attack tries to hide anything in the original image by leveraging another image component [30]. Figure (2.4) and Figure (2.5) shows an example of the Copy-Move form of action. The original picture only has three warheads, while the right-hand Copy-Moved version has four.



A B

Figure 2.4: copy move Forgery A. Original image , B. Tampered image [31].

A B

Figure 2.5: Copy-Move Attack on Images [23] A. Original image, B. Tampered image

2.3 Face Datasets

When evaluating an algorithm's accuracy, it's best to use a standard test data set so that researchers can compare results directly. Over the years, many face data set have been published to test human face-based computer vision applications like face detection and recognition.

This section will mention part of the most important face databases used in previous times

- Flickr-Faces-HQ Dataset (FFHQ)

  FFHQ is a human face dataset that provides more diversity in terms of age, race, and photo history than the CELEBA-HQ dataset, as well as greater representation of accessories like eyeglasses, sunglasses, and hats. Flickr photos are crawled, then auto-aligned and cropped [32].

- 140k Real and Fake Faces

This dataset includes 70k REAL faces from the Flickr dataset, in addition to 70k fake faces (produced using StyleGAN) selected by the provider from 1 Million FAKE faces [33].

- Diverse Fake Face Dataset (DFFD)

Diverse Fake Face Dataset termed (DFFD) is one of the datasets having a variety of fake faces (DFFD). DFFD has a wider range of features, which is useful for detecting and locating facial manipulations. It comprises real-life portraits of people. CelebA and FFHQ datasets, and frames from 1000 real videos of FaceForensics++ as additional real faces. And their corresponding 3,000 manipulated videos, Images are modified at random. Where three false images Created for each face in FFHQ, and 40 fake images for each face in CelebA. In all, 100k and 200k high-quality fake images were created, respectively, using the pre-trained models of StyleGAN and PGGAN [34].

- Tufts-Face-Database

The Tufts Face Database is wide or the most inclusive and extensive face dataset available, with seven image styles: visual, thermal, near-infrared, computer graphic, LYTRO, and 3D images, recorded video [32].

- Labelled Faces in the Wild Home (LFW) Dataset

The LFW dataset (Labeled Faces in the Wild) is a face-image database created to investigate the issue of unregulated face recognition. Face authentication, also known as pair matching, is a general norm for faces with labels in the wild, in other words, this dataset is related to Biometric systems [32].

The Biometric verification systems are used in many areas of life. A biometric verification system uniquely identifies or verifies an individual based on one or more biometric features of the person to be identified. In other words, the biometric verification system answers the question of "who you are" in both digital and physical scenarios. Each person has multiple unique biometric identifiers such as iris, fingerprint, ears, DNA, and face, etc. These identifiers are impossible to replicate in another person. So companies, organizations, governments, and law enforcement agencies adopt biometric verification systems in their work as a means to achieve safety and security. One of the most important biometric verification systems is the face verification system [35].

2.4 Features Extraction

The extraction of the features is a crucial step in detecting forgery by copy transmission. Many studies have looked into this topic in various ways. It has been split into more sections by some scholars. To detect forgery, (Ashwini, Siddharth) concentrated on the contents of colors. The function is primarily extracted using the automatic color graph, color space, and HSV color

moment [36]. Also, (Nooraini, Loai) divided the methods for extracting experimental traits into two classes. Dimensional reduction and base point are two types of categories. On the other hand, Keypoint methods, extract features from points of interest in the image [37]. On another side, Artificial Intelligence and artificial neural networks like convolutional neural networks (CNN) using the deep learning approach create feature maps of the input data like the images and apply these feature maps to the classifier to solve the problems of data. Each data set has a unique problem or multi problems and strategies are applied differently to solve those problems. The advantage of a CNN classifier is including the list of layers that convert the input size into the output size with simplest method. Each layer of CNN converts inputs to outputs through an adjustable function [38].

2.5 Neural Networks

Artificial neural networks (ANNs) and synthetic neural networks (SNNs) gets their name and shape from the human brain, and they look like biological neurons communicating with one another [39].

(ANNs) are commonly used machine learning methods and techniques that model biological organisms' learning processes. Neurons are cells that make up the human central nervous system. Synapses are the connecting areas between axons and dendrites. The frequency of synaptic connections varies a lot in response to external sensory influences. This change occurred in the way living beings think [40].

2.5.1. The Basic Architecture of the Neural Networks

The basic Form of simple Neural Network are:

- Perceptron neural network**:**

The first model of the ANN is called a Perceptron neural network. Perceptron is a single artificial neuron, which takes multiple inputs (x1, x2, x3, ..., xn), each input, has its weight, and generates one output (yes, or no, 0 or 1, etc.). Figure (2.6) shows a single neuron (perceptron). The assembly symbol ($\boldsymbol{\Sigma}$) denotes the sum of the dot product of the input xi, and the weights wi and the bias neurons b.The bias is a constant value always equal to 1 and finally, add the output of the product to the output node to be aggregated by applying the signal activation function denoted by the symbol (⌐¬) to return the final class label corresponding to the input data. As shown in the sum of the product account shown in equation (2.1), the calculation of the output by applying the activation function shown in equation (2.2) [41].

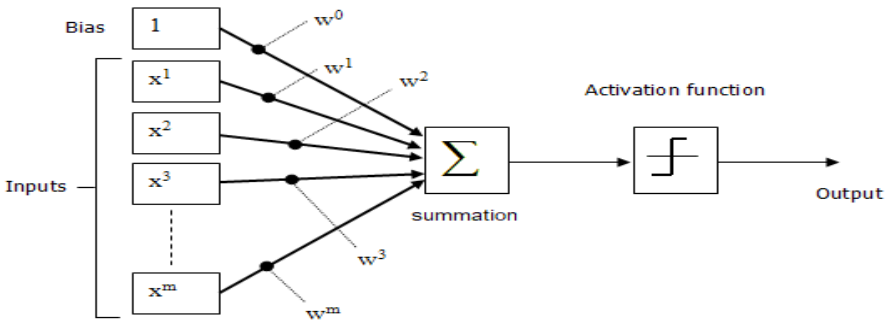


Figure 2.6: Sample perceptron Neural Networks [42].

$$\mathbf{net} = \sum_{i=1}^{n}(x_i w_i) + b \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

$$f(\mathbf{net}) = \mathbf{O} = f(\sum_{i=1}^{n}(x_i w_i + b)) \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

- Multi layer perceptron:

Another architecture of neural is a shallow network (contained at least one hidden layer) can be used to solve easy problems, as the structure is depicted

in Figure (2.7). This neural network is made up of three layers: an input layer, a hidden layer, and an output layer. that is flows from left to right) [41].

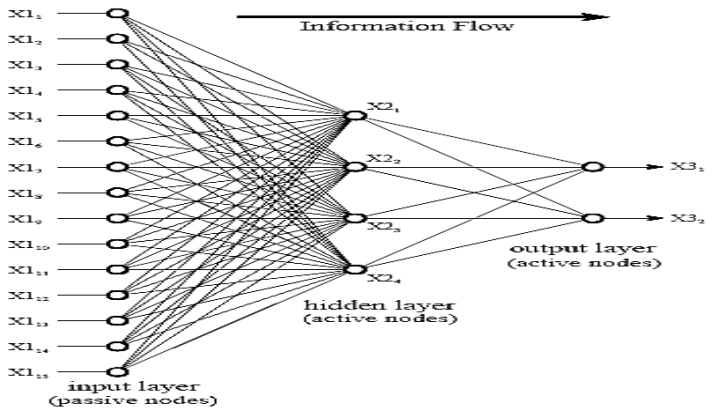


Figure 2.7: Structure for Neural Networks: Three Layers with Full Interconnection [41].

If there is a complex pattern that needs to be recognized, for example, a visual pattern, many hidden layers are required in this state, it must be called a deep neural network (DNN) [41]. as shown in Figure (2.8).

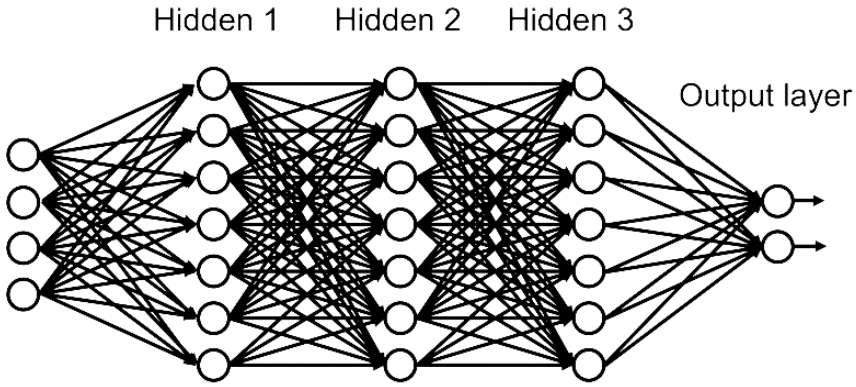


Figure 2.8: Deep Neural network (DNN) [43].

2.5.2 Activation Functions

Activation functions in artificial neural networks convert an input signal to an output signal that is then fed as an input to the next layer in the stack. In an artificial neural network, measure the number of input products and their corresponding weights, then add an activation function to obtain the output of that layer and supply it as an input for the next layer [44].

The activation functions are principally divided into two types, linear and nonlinear activation functions. The linear activation function is used then the output is similar to the input based on linear changes, it is used to solve simple problems, since the derivative value of the input to be constant for the backpropagation process [45]. The nonlinear activation functions that have more than one degree when curvature is plotted. It is represent and process any data and any complex problems which perform non-linear mappings the inputs to the outputs. An important advantage that it can perform a backpropagation strategy in order to reduce errors [44].

The most three activation functions that are important for use in hidden layers:

a. Rectified Linear Activation (ReLU)

Rectified Linear Units (ReLU). Widely used in Neural networks especially Deep learning models, it produces 0 when x is zero or negative value and return same value in another states [46]. the function represented by the equation (2.3). Figure (2.9) represents the ReLU Activation Function plot.

$$ReLU(x) = \max(0, x) \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.3) [46]$$

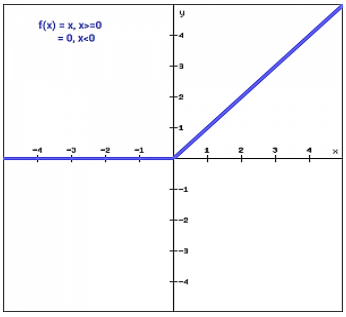


Figure 2.9: ReLU Activation Function [47].

b. Sigmoid

Since sigmoid is a non-linear function, it is the most commonly used especially in binary classification. The sigmoid function transforms

values in the range 0 to 1 range [48]. As shown in Figure (2.10), it can be summed up as equation (2.4):

$$f(x) = 1/(1+e^{-x}) \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.4) [48]$$

The function's derivative as equation (2.5):

$$f'(x) = 1\text{-sigmoid}(x) \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.5) [44]$$

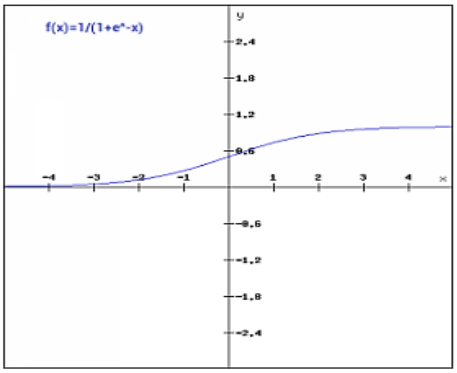


Figure 2.10: Logistic (Sigmoid) Activation Function [44].

c. Softmax

The softmax function is one of the activation function types, it is widely used in neural computing at the last layer to calculate the probability of multi classes of more by using a vector of real numbers. The output of Softmax function values in the range between 0 and 1, where the summation of the probabilities is equal to 1, and the target class which have the highest value, the equation (2.7) represent softmax function [45].

$$\boldsymbol{P(class(i)) = j(net))} = \frac{e^{net_j}}{\sum_{k=1}^{n} e^{net_k}} \quad \ldots\ldots\ldots (2.7) [42]$$

2.5.3 Loss function

In early models of neural networks, the error is measured by formulas for calculating the difference between the real output and the

expected output, these formulas are called Loss Functions [49]. There are three main types of loss functions: Classification Loss, Regression Loss, and Embedding Loss Functions [50].

The most tow Loss Functions that are used for the classification process:

a. binary cross entropy

the binary classification issues in deep neural networks, the objective function that is the binary cross entropy [51] as shown in equation (2.8).

$$\boldsymbol{l = \frac{1}{N}\sum_{n=1}^{N}[y_n log(\hat{y_n}) + (1 - y_n) log(1 - \hat{y_n})]} \quad \text{..... (2.8) [51]}$$

Where $\boldsymbol{y}$ and $\boldsymbol{\hat{y}}$ are the true label and the estimated output probability from the neural network respectively.

b. Categorical Cross-Entropy

This loss function uses in Multi-class classification problems. The target can only belong to one out of many possible categories [51]. The equation (2.9) illustrates the categorical cross-entropy.

$$\boldsymbol{L = -\sum_{n=1}^{N}[y_n \log(\hat{y}_n)]} \quad \text{..... (2.9) [51]}$$

Where **y** one hot encoding vector $\boldsymbol{\hat{y}}$ probability distributions, and N output size.

2.6 Back Propagation in Neural Networks

Back-Propagation (BP) is at the heart of neural net preparation. It's a method for boosting and generalization the model's by fine-tuning the weights, making it more accurate [52]. The BP algorithm aims to decrease the resulted error among the desired and outputs by a function named optimization function [53]. The optimization algorithms typically measure

the gradient, which the partial loss function derivative concerning weights, and the weights are adjusted in the reverse direction of the measured gradient [50]. This loop is repeated while the model reaches a minimal loss of function is reached. Figure (2.11) shows the operation of computing the gradient and the relationship of the loss function with the network weights.

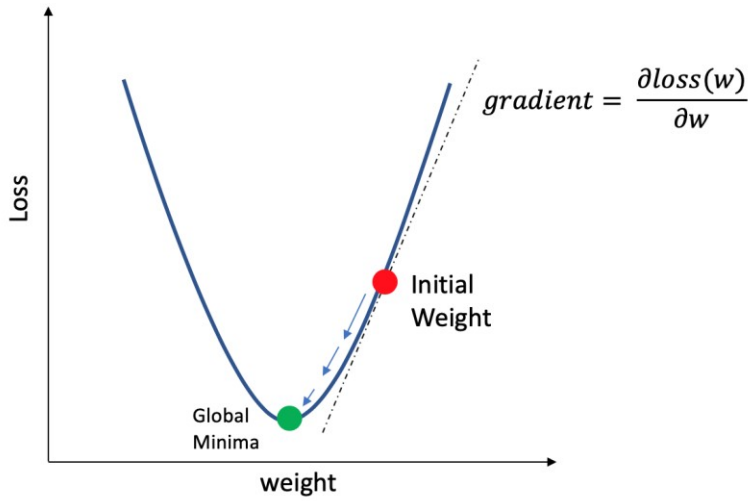


Figure 2.11: computing the gradient relationship of the loss function [54].

BP algorithm involves two phases which are forward and backward phase.

a. Forward phase: the process is shown in Figure (2.6) of perceptron.

1- Input patterns fed into the network in forward formula.

2- Compute predicted output.

- Compute the *net* which is the sum of product weights with input, as in equation (2.1).
- Compute *f(net)* which is the predicted output referred to as *out* or *o* for the net as in equation (2.2), applying activation function.

3- Compare desired *(target)* and predicted *(out)* outputs, error differences between the desired and predicted output for neural output must be calculated using loss function such as cross-entropy loss function as shown in equation (2.8), and equation (2.9)

b. Backward phase: the process is shown in Figure (2.12).

1- Pre-calculated errors have to be propagated in backward form.

2- One of the most popular optimization technique usually used with back-propagation algorithm, gradient descent, used to make the predicted output more close to the desired output. The Total error is partially derivated E(total) to adjust The error in relation to a specific weight, by applying the chain rule [56], as the equation (2.10).

$$\frac{\partial E(total)}{\partial w1} = \frac{\partial E(total)}{\partial out(o1)} * \frac{\partial out(o1)}{\partial net(o1)} * \frac{\partial net(o1)}{\partial w(i)} \quad \ldots. (2.10) [56]$$

3- Weight difference value (ΔW) can be achieved by multiplying the resulted gradient descent of specific weight value with the *learning rate (η)* as the equation (2.11) below.

$$\Delta W = \eta \frac{\partial E(total)}{\partial w(i)} \quad \ldots\ldots\ldots\ldots (2.11) [55]$$

4- The current weight will be adjusted by subtracting the weight difference from it as clarifying in equation (2.12).

$$W(i)_{new} = W(i) - \Delta W \quad \ldots\ldots\ldots\ldots (2.12) [56]$$

5- Weight adjustment process is still going back to the hidden layer in multi-layer network to the input layer until the error become smaller that the upper bound of error allowed (E_{max}) [49][55].

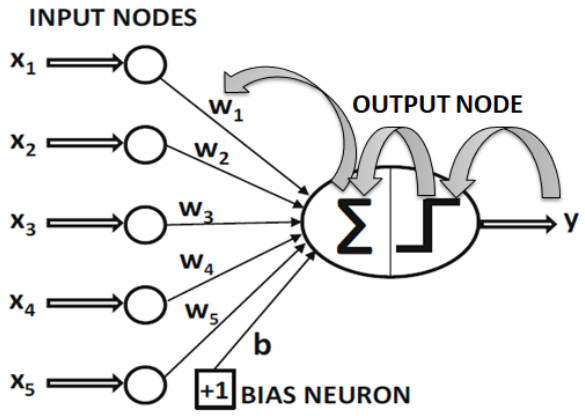


Figure 2.12: Backward phase of back-propagation algorithm for single perceptron [55].

2.7 The optimization algorithms

One of the most essential phases is choosing the algorithm to use to optimize a neural network. The Main types of optimization methods in machine learning are batch or static gradient methods, which handle all training instances in a big batch or mini-batch when $N_i < N$ instances, at the same time, and stochastic or online methods, which work with only one instance at a time [50].

There are two types of optimization algorithms: algorithms of adaptive learning and algorithms with constant learning rate, such as SGD In the first group, the learning rate η is chosen manually. When choosing a relatively small learning rate, the learning process is slowed, and the training time becomes too large. While choosing a relatively large learning rate, it can lead to fluctuation in the loss value around the minimum value, and this hinders the convergence process. While the algorithms of the second group, they do not need to set the learning rate manually, but they have a heuristic approach that takes care of adjusting the learning rate value, several algorithms appeared that belong to these two categories [56], the most important of which are illustrates as follows:

a) Stochastic Gradient Descent(SGD)

SGD is one of Constant Learning Rate Algorithms that tries to update the network parameters more repeatedly. In this algorithm, the network parameters are modified on each training sample after the loss calculation. Therfore, if the dataset contains 900 samples, the network parameters will be modified 900 times in one cycle of the dataset [56]. The SGD equation that is used to update parameters in a neural network is illustrated as equation (2.13):

$$W^{(k+1)} = W^{(k)} - \eta * (\Delta J(W)) \quad \ldots\ldots\ldots (2.13) [56]$$

W is network a parameter (weights, biases), η is the learning rate, $\Delta J(W)$ is the gradient, which is taken of loss function.

b) Adam

Adam that empirically appear to significantly reduce the time and resources needed to tune the initial learning rate[57], it is considerd one of adaptive learning rate optimization algorithms, it measures individual learning rates for various parameters [56]. Adam sets the learning parameter automatically, this was done by using the first and second-moment estimation. But what's the moment?. The moment is the expectation of a random_variable at the power of n [58]. The moment can illustrate in equation (2.14).

$$m_n = E[X^n] \ldots\ldots\ldots\ldots\ldots\ldots (2.14) [58]$$

Where: m is the_moment, and X is a random_variable.

The equations (2.15) and (2.16) uses to estimates, the first and second moment Adam [58].

$$\acute{m} = \frac{m_t}{1-\beta_1} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.15) [58]$$

$$\tilde{v}_t = \frac{v_t}{1-\beta_2} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.16) [58]$$

Where m_t the previous first moment and v_t ,the previous second moment and they are initialized with 0 in the first step.

β1,β2 are new parameters inserted into the algorithm. They have default values of 0.9 and 0.999 respectively.

After calculating the value of the first and second moments, it is used to update the network weights according to the equation (2.17) [58].

$$W_t = W_{t-1} - \eta \frac{\boldsymbol{m_t}}{\sqrt{\tilde{v}\boldsymbol{t}+\epsilon}} \quad \ldots\ldots\ldots\ldots\ldots \text{(2.17) [58]}$$

Where W is network weights, η is Stepsize, $\boldsymbol{\epsilon = 10^{-8}}$

2.8 Deep learning Approach

Deep learning is a field of artificial intelligence and both are an elements of machine learning [59]. It consists of hierarchical architecture, where each higher layer builds on its previous lower layer, which makes it popular for the first time as hierarchical learning [60]. The beginning of deep learning was in 2007. Deep learning works well with a massive amount of data to solve a complicated problem. Many applications improved with deep learning such as Object Recognition, Object detection, Automatic Machine Translation, Investment Modeling, and drug discovery [62].

Deep learning can extract high-level features from the input data. For example, in image processing, deep learning can be used in several applications such as edge extraction, image segmentation, shape recognition, and face verification [61].

Deep learning algorithms such as the Convolution Neural Network (CNN) boost its efficiency on the conventional algorithms due to the automated function learned without the need for human computational effort as it does in conventional machine learning techniques. The conventional machine learning technique requires a feature vector corresponding to the raw data (intensity pixel values in the case of an image) as an input to detect

and process it so the features must be extracted and later fed into a classifier. Deep learning algorithms can learn these features on their own. The goal of designing deep learning algorithms is to overcome the problems that traditional machine learning algorithms have been unable to solve. The most important problems that deep learning overcame can be described as follows: [60].

- Curse of Dimensionality

Many fields such as numerical analysis, data processing, and machine learning are faced with the problem of curse dimensional. The popular theme of curse dimensional problems is that the amount of space increases so rapidly that the data available are sparse with the dimensionality increases. The deep learning has overcome this problem due to the enormous ability of its algorithms to handle multiple dimensions [60].

• Local Constancy and Smoothness Regularization

Machine learning algorithms require to be driven by previous beliefs about the type of function they can learn. The smoothness or the local consistency is among the most commonly used beliefs. An algorithm is said to have smoothness if within a small region, the learned function should not be changed very much. Many simpler Machine learning algorithms depend on local consistency beliefs to generalize well, and so that these algorithms fall to scale statistical challenges in AI tasks [60].

2.8.1. Deep learning Types

There are three main types of deep learning models, each has its specific network architectures inside and has specific use.

a. Supervised deep_learning

In Supervised Deep Learning algorithms, the model is trained on pre-labeled data. Then the model is used the learning algorithm to adjust itself, and during the testing phase, the model should determine the correct answer without relying on any label. Supervised deep_learning has two main fields: classification problems and regression problems. One of the most frequently supervised models used is a convolution neural network (CNN). Some of the CNN-based supervised deep learning architectures are AlexNet, LeNet-5, VGGNet, GoogleNet, resnet, InceptionNet, and others [50] [40].

b. Unsupervised deep learning

In the unsupervised Deep Learning algorithms, training the model based on unlabeled data, and the model tries for extracting patterns and features on its own. Some of the unsupervised deep learning architectures are Deep Belief_Network (DBN), Restricted_Boltzmann_Machine (RBM) [60].

c. Semi-Supervised Deep Learning

Semi-Supervised Deep Learning takes on an intermediate stage between supervised and unsupervised learning. It takes a lot of categorized data to support a larger collection of unlabeled data. This approach is especially useful when it is difficult to extract relevant data features, and when labeling the samples is takes a long time. The common method that uses this approach is Generated Adversarial Networks (GANs) [62].

2.8.2 Convolution neural network (CNN)

CNN is type of deep discriminative architecture that has been demonstrated to be effective in processing two-dimensional data such as images and videos, with a grid-like layout. The architecture of CNNs is based on brain cortical architecture in animals. During the 1960s, CNNs is the first successful architecture for deep learning Because of the excellent training of the hierarchical layers. The CNN architecture takes advantage of spatial relationships to reduce the parameters number of the network, which improves performance when typical BP algorithms are used. The growth of computation techniques and GPUs-accelerated, have been used to train CNNs more effectively. The CNN performs two functions, the extraction function then the prediction function. Each function used certain layers to achieve a particular purpose [63].

2.8.3 Basic components of CNN architecture [64].

The basic structure of CNN in the mainly from five layers. In addition to functions between the layers for modifying and normalizing the data.

a. The input layer: The computer reads the input data as such an array that will be predicted in the end.

b. Convolutional layer: In the convolutional layer, every single neuron connects with neurons from neighboring layers and often includes many feature mappings. There are frequently several feature maps in a convolutional layer. Each feature map is made up of a rectangle-pattern arrangement of numerous shared weights. These shared weights are convolution kernels that will be modified on a regular basis during the training phase. The convolutional results will be used as input to the following layer.

c. The activation layer: The activation function is used to improve the model's non-linear features by compensating the shortcomings for the lack of linear computational expression.

c. The pooling layer: the most commonly used approaches in the pooling layer are Maximum and the average pooling. Pooling may be described as a simpler formulation of convolution in other words a sampling layer.

d. Flatten: The depth of the convolution layers output might become more than one. The flatten transforms the output of the convolution layers to generate a flat structure that may subsequently be provided as input to a fully connected layer

e. The fully connected layer: (classification layer). A fully connected multilayer classifier. Usually placed at the end of the network makes up where the Back-propagation will begin When the forward propagation process accomplishes the final fully connected layer of classification, weights and biases are updated to decrease losses and enhance the accuracy of the classification.

2.8.4 Two Dimensional Convolution Neural Network Architecture

2D CNNs are designed with the assumption that the input will be images. This topic focuses on the architecture that should be put in place to deal with the specific type of data [65]. Convolutional Neural Networks (CNN) is a biologically inspired variation of feed-forward networks in which the connections between neurons have a tendency to capture patterns invariances for the input data in the distortion or shift. The majority of CNN architectures considered that the networks will work with two dimensional input data (usually images). A standard CNN is constructed with a stack of layers, each of which transforms

one volume of activations into another. Figure (2.13) shows an example of CNN architecture that recognizes objects within an image [60].

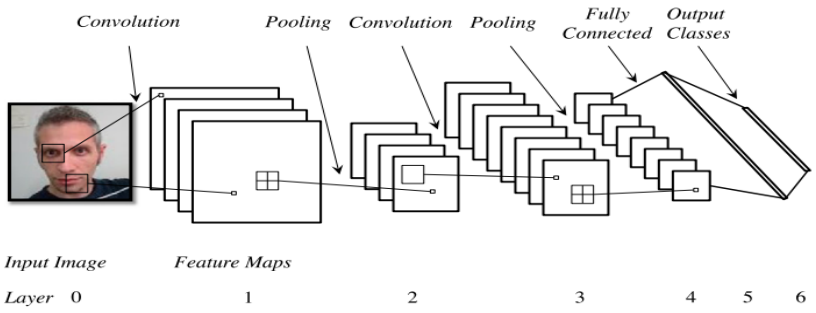


Figure 2.13 example of CNN architecture that recognize an image [60].

a) Convolution Layer

In computer vision field concentrate on discrete 2D-convolution because it's the most common form of convolution in digital image processing. The convolution procedure determine the value of pixel intensity in an image. To measure the corresponding pixel [66][65]. The spatial dimensionality of the input (height and the width) and the depth. The depth refers to the third dimension of an activation volume. the neurons within any given layer will only connect to a small region of the layer preceding it [67]. A convolution layer is a vital element of the CNN architecture that is does the operation of feature extraction. It usually comprises of a blend of nonlinear and linear processes, such as convolution and activation functions [68]. Convolutional is a mathematical process to combine two groups of information. It is used convolution filters to extract features from the input image and learn these features using input data arrays whereas allowing the cultivation of the spatial relationship of each feature in the image by using these filters to generate feature maps.

The convolution process was accomplished by sliding the filter/kernel across an input image. On each spatial location, the element-wise array is multiplicated and computes the result. Then this result will go to the feature map. The 2D convolution layer applies a 2D kernel (filter) on the input image to extract features and produce feature maps [69].

For instance, in Figure (2.14) and Figure (2.15) an input image 5 x 5 and applies a 3 x 3 kernel and convolved over the image and convolution operation output will be a 3 x 3 image matrix and visualize it.

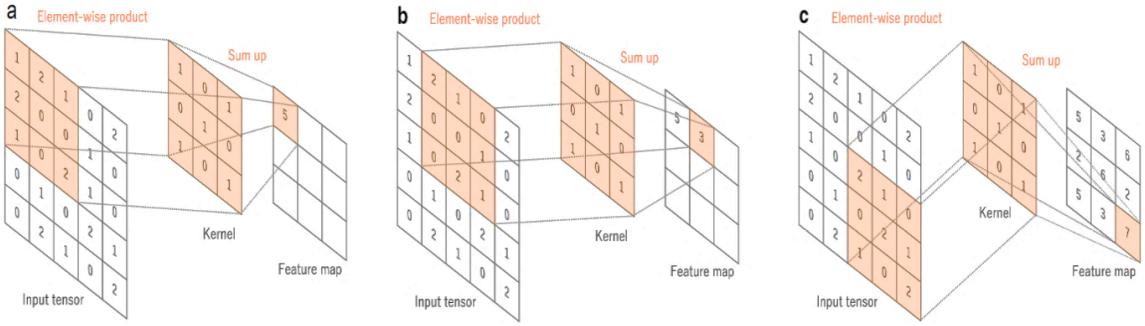


Figure 2.14: Convolution Operation Calculates [68].

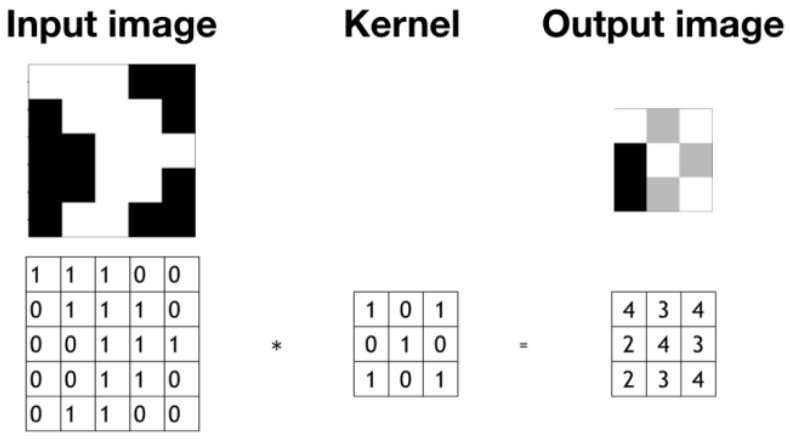


Figure 2.15 convolution kernel-matrix on input image and output [70].

b) Depthwise Separable Convolutions

Depthwise separable convolutions work with kernels that can't be "factored" into two smaller kernels. As a result, it's becoming more popular.

The depthwise separable convolution gets its name from the fact that it works for both the spatial and depth dimensions the number of channels. Three channels are possible in an input image: RGB stands for red, green, and blue. Each channel represents a different interpretation of the image; for example, the "red" represents the "redness," the "blue" represents the "blueness," and the "green" represents the "greenness" of each pixel [71][72].

The depthwise separable convolution involves two phases, a depthwise convolution and a pointwise convolution. In the depthwise convolution, a spatial convolution is conducted separately on every input channel, and the outputs from the depthwise convolutions are then combined via pointwise convolution. Additionally as shown in Figure (2.16), depthwise separable convolution layers can produce more deep information characters. Simultaneously, they demand lower computational costs and fewer parameters number and provide comparable (or somewhat higher) performance and scalability. Combining numerous simple convolutions into a block increases the neural network total depth and provides for more accurate advanced features extraction [73].

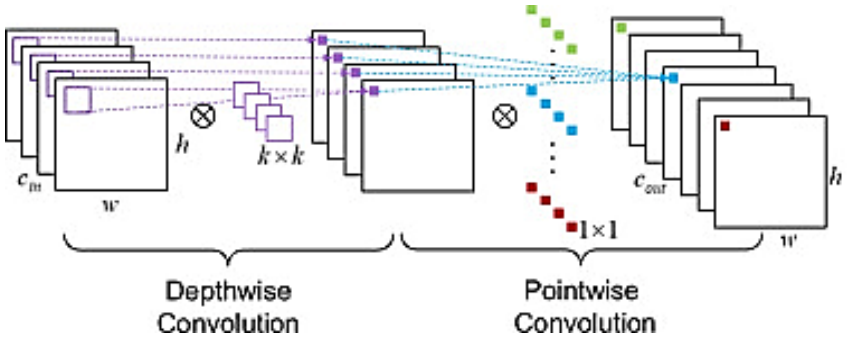


Figure 2.16. Depthwise separable convolution Architecture [56].

For instance [74], there are convolution layer of size DR*DR*NI*NO, feature map of size DF*DF*NI as input map and DH*DH*NO output feature map where:

• DR: the filter size;

• DF: the size of input feature maps;

• DH: the size of output feature maps;

• NI: the number of depth dimension of input feature maps;

• NO: the number of depth dimension of output feature maps.

The complexity of standard convolution is:

DR*DR*NI*NO*DF*DF …… (2.18) [74]

In depth-wise convolution, extracts information from every channel of feature map and does not merge them such as standard convolution. Therefore, depth wise convolutions computational cost dispose of NO and is calculated as: [74]

DR*DR*NI*DF*DF …… (2.19) [74]

Point-wise convolution is much the same standard convolutional except that the filter size is 1 x 1, and is calculated as:

NI*NO*DF*DF ……..… (2.20) [74]

Depth-wise separable convolution represents convolution operation as a two-step process of extracting and compressing to have a complexity of:

DR*DR*NI*DF*DF + NI*NO*DF*DF ……..... (2.21) [74]

The Comparing of standard convolution with the depth wise technique gets a reduction of:

$$\frac{\mathbf{D}\boldsymbol{R}*\mathbf{D}\boldsymbol{R}*\mathbf{N}\boldsymbol{I}*\mathbf{D}\boldsymbol{F}*\mathbf{D}\boldsymbol{F} + \mathbf{N}\boldsymbol{I}*\mathbf{N}\boldsymbol{O}*\mathbf{D}\boldsymbol{F}*\mathbf{D}\boldsymbol{F}}{\mathbf{D}\boldsymbol{R}*\mathbf{D}\boldsymbol{R}*\mathbf{N}\boldsymbol{I}*\mathbf{N}\boldsymbol{O}*\mathbf{D}\boldsymbol{F}*\mathbf{D}\boldsymbol{F}} = \frac{\mathbf{1}}{\mathbf{N}\boldsymbol{O}} + \frac{\mathbf{1}}{\mathbf{D}^{\mathbf{2}}\boldsymbol{R}} \text{ ……... (2.22) [74]}$$

the CNN architecture has many parameters that use to control the output size of the convolution layer, these parameters called hyperparameters. Some of the important CNN's hyperparameters are as follows: [75]

- Number Filters: Different reasonable number of filters can be used with different sizes.
- Filter size: Filter or kernel size must be squared k*k, smaller than the input imag.
- padding: is an efficient technique for controlling the dimensionality of output volumes by performing the process of filling at the input boundary.
- Stride: number of cells (pixels) to move the kernel through or down the input map at the same time.

c) Activation layer (Non-Linear Layer)

Non-Linear layers apply many functions such as 'Rectified Linear Units' (ReLUs) functions. These functions are used to determine the distinctive features of every hidden layer. The feature maps output from the convolution layers becomes input to Non-Linear layers to convert the linear output into nonlinear [69].

d) Pooling layer

The pooling layer transforms the Representation of the common feature into a more useable one that keeps essential information while eliminating irrelevant information, then in the pooling layer the resolution of the feature maps is decreased in addition to improving the stability to deformation on the inputs [76]. The two most frequently used pooling approaches are maximum pooling and average pooling. In Max

Pooling Returns the greatest value of the input map portion that covered by the Kernel, in a Similar manner, the average pooling returns the average of the values of the image portion covered by the Kernel [77]. Another types of pooling layer is the Global Max Pooling layer and Global Average Pooling. Where the pool size is equal to the input size, and the maximum value of the whole input map is calculated to be as the output in Global Max, while the average of the whole input map values is calculated to be as output value in Global Average Pooling [78]. As shown in Figure (2.17).

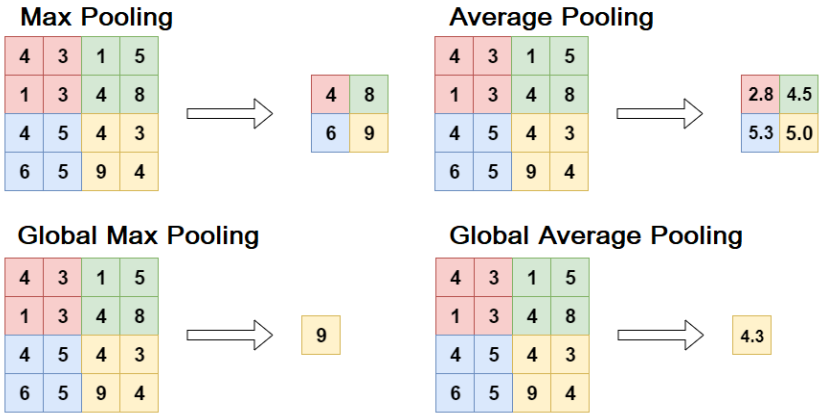


Figure 2.17 Types of Pooling layer [78].

e) Regularization layers

Overfitting is one of the important issues that face the network model, this issue occurs in both following cases: when a model is complex, and when a dataset is poor. The overfitting cause's difficulty in generalizing the model on the unseen samples, therefore, the accuracy of the model will be high during the training phase and low during the testing phase. The regularization layers are one of the important methods to avoid the overfitting problem. The most two used regularization layers and can be illustrated as follows:

- Dropout layer

Dropout is a technique that addresses the overfitting issue. The word "dropout" refers to removing (both hidden and visible) units.

Where dropping a unit out in a neural network means logically removing it or in other word disconnecting its incoming and outgoing connections in the network. These units are chosen randomly based on value, for Example can be set at 0.5 [79] as shown in figure (2.18).

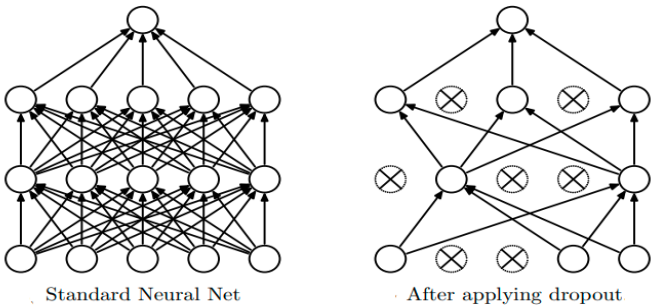


Figure 2.18 The dropout process [79].

- Batch Normalization layer activation

Deep networks need layers of normalization and activation functions as building a blocks for stable generalization improvement and optimization. Batch normalization is one of the techniques that regularize the neural network and avoid overfitting. It makes to solve the Internal Covariate Shift problem which is defined as the changes in the distributions of inputs to the current layer due to the changes in the parameters of the preceding layer. Readjusting the current layer to new distribution constantly is required accordingly, In order to increase training performance. Normalizing the output of the previous layers by fixing it to be zero mean and one standard deviation(std) [80]. Suppose input data x, batch normalization of it is as the equation (2.23):

$$X1 = x - mean, \quad x = X1/std. \quad \ldots\ldots\ldots\ldots (2.23) [80]$$

By continuing training the network, layer by layer, it becomes more intelligent to recognize shapes difficult than those recognized by the previous layer. Where the beginner layers capable to recognize colors,

lines, corners, etc. The later layers are capable to recognize components of shapes and the overall shapes [80] finally. The visualization of CNN layers can be clarifying in Figure (2.19).

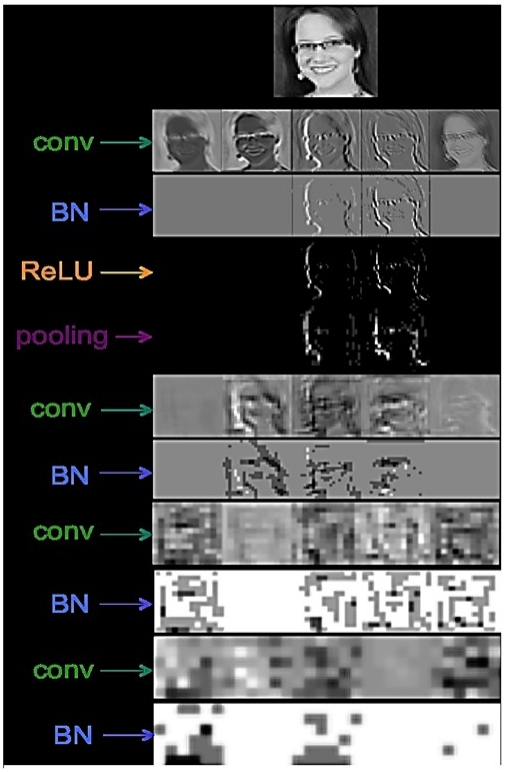


Figure 2.19 visualization of CNN layers [81].

2.8.5 Xception Network

Xception is a Depthwise Separable Convolutions-based deep convolutional neural network design. The concept was devised by Google researchers. Google researchers describes inception modules as shown in Figure (2.20) to be as a convolutional step between regular convolution and depthwise separable convolution. In this way, the suggestion of new deep architecture is based on Inception modules.

2.10.1 The Xception architecture

Francois Chollet proposes a depthwise separable architecture based on convolution layers. Proposing the spatial correlations in convolutional neural network feature maps and mapping go cross-channel can effectively dissociate. The proposition of Xception architecture, which is short for

"Extreme Inception, "because this idea is a powerful version than the assumption of Inception architecture [71].

Figure (2.21) shows a detailed outline of the network's Architecture. The feature extraction based on Xception architecture is made up of 7 modules of 14 blocks. The experimental evaluation will focus solely on image classification, the 14 blocks are divided into 36 convolutional layers, all of the blocks have connections with residual layers around the blocks excluding the first and last modules [71].

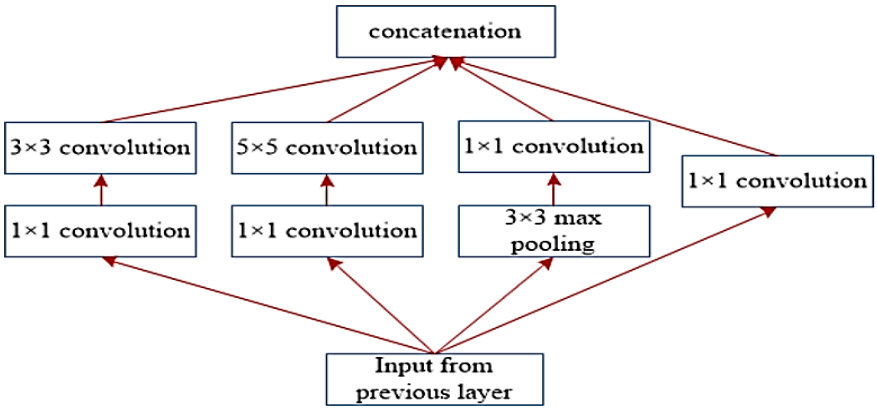


Figure 2.20 Simplified Inception Architecture [82].

2.9 DenseNet

DenseNet is a breakthrough in neural networks for visual object recognition. Every layer receives features from all preceding layers. In Densenet the output of the previous layer is concatenated with the output of the current and next layers. DenseNet was designed with the objective of resolving the effect of a vanishing gradient on the accuracy of high-level neural networks. in Simplest terms, due to the longer pathway between input and output layers, information vanishes before it reaches its destination [83][84]. The diagram in Figure (2.22) depicts the DenseNet Architecture of a 5-layer dense block with a k = 4 growth rate [85].

The DenseNet is separated into DenseBlocks, with its own set of filters but same dimensions, which the Transition Layer utilizes to down sample and perform batch normalization at the end of each dense block [79] As shown in Figure (2.23).

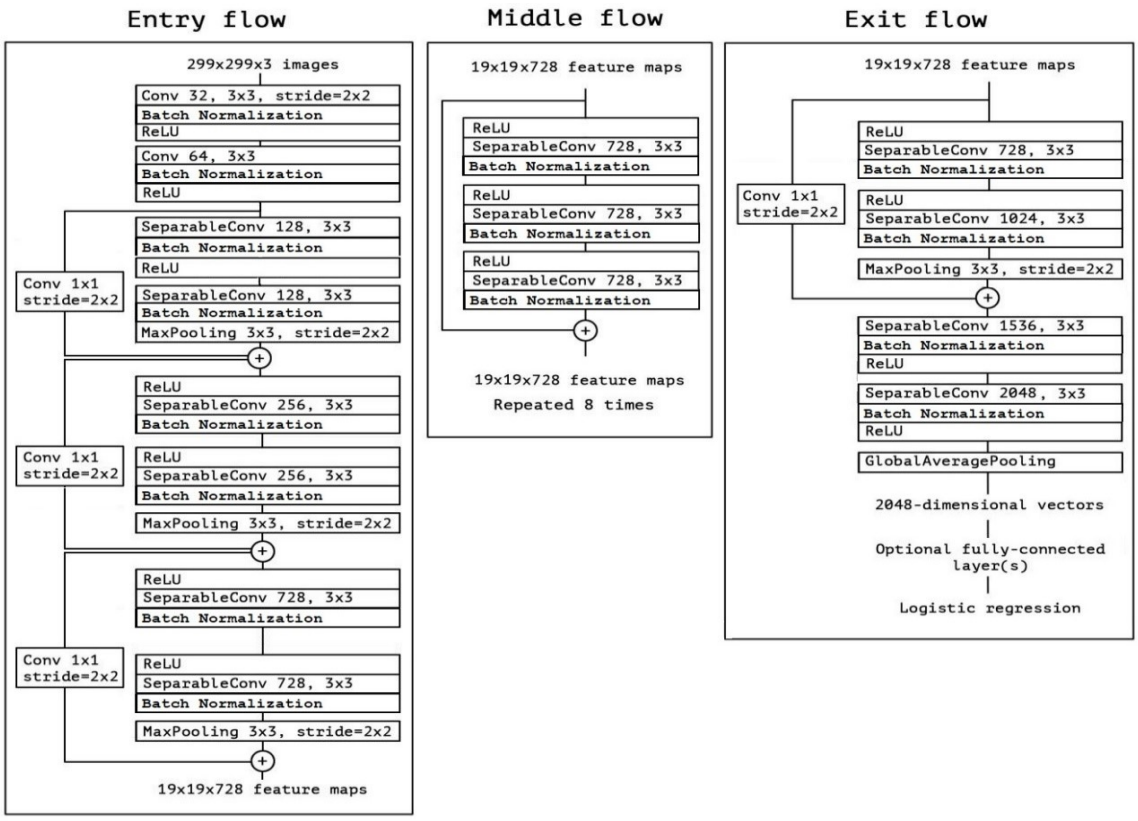


Figure 2.21: The Xception architecture [71].

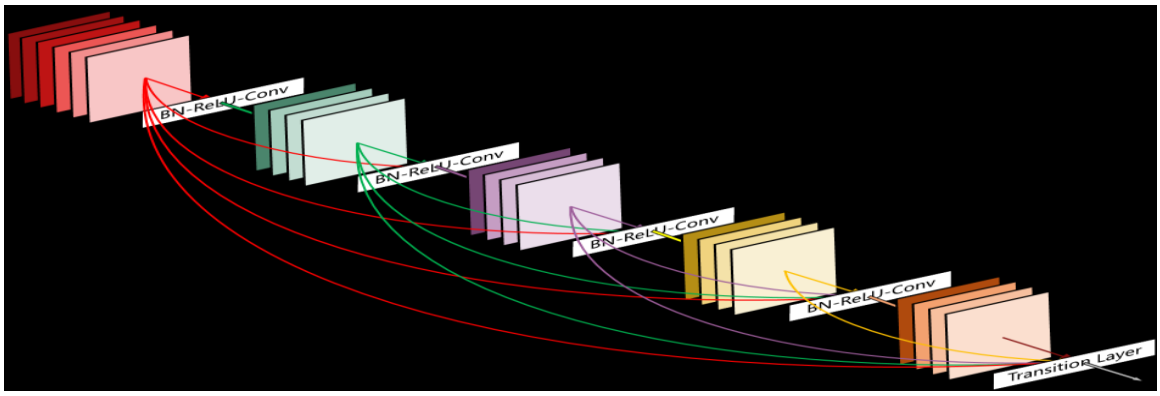


Figure 2.22: A DenseNet 5-layer block with a growth rate k=4 [85]

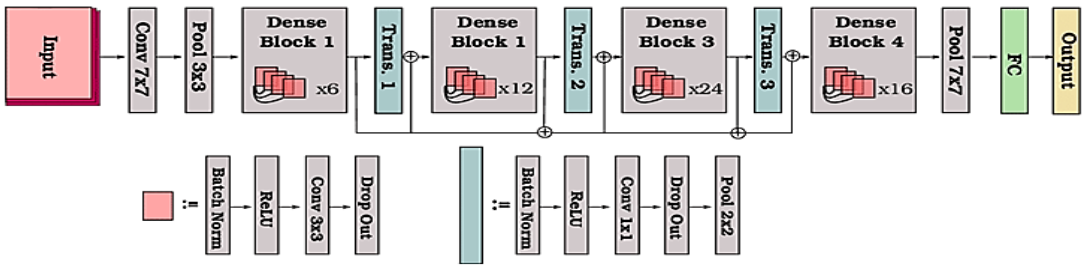


Figure 2.23: A DenseBlock and transition layer [85]

2.10 Generative adversarial networks (GANs)

GANs (Generative adversarial networks) are algorithmic structures that play two neural networks one against the other (thus the word "adversarial") to create new, synthetic samples of data that may be passed as real data. They're commonly employed in the creation of images, videos, and audio.

GAN works by training a two networks discriminator and a generator at the same time as shown in Figure (2.24). The discriminator is taught to differentiate between actual dataset samples and generator-generated fake samples. The generator is trained to generate fake samples using input from an easy-to-sample random source that the discriminator can't tell an element from real data samples [86].

Because GANs can adapt to imitate every data delivery, they have enormous potential for both good and bad. GANs, can be taught to construct worlds that are uncannily in every domain: pictures, music, and voice. In other words, GANs can be used to create fake media material, and this is the technology that underpins Deepfakes [87].

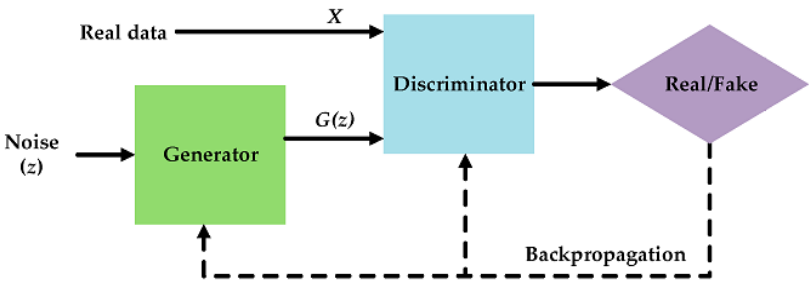


Figure 2.24 simple Architecture of GAN network [88].

2.10.1 StyleGAN

The big improvements have been done in the Generator portion of the "Progressive Growing GAN" architecture. The traditional and style-based generator networks are shown below in Figure (2.25).

The latent vectors in a traditional network pass directly into the block after normalization, while the latent vectors in the StyleGAN network cross through to the mapping network (network of eight fully connected layers), where their outputs are transfigured (A stands for affine transfiguration, that is a collection of linear and translation) and passed into the block(AdaIN short for Adaptive instance normalization, equation (2.24) [89].

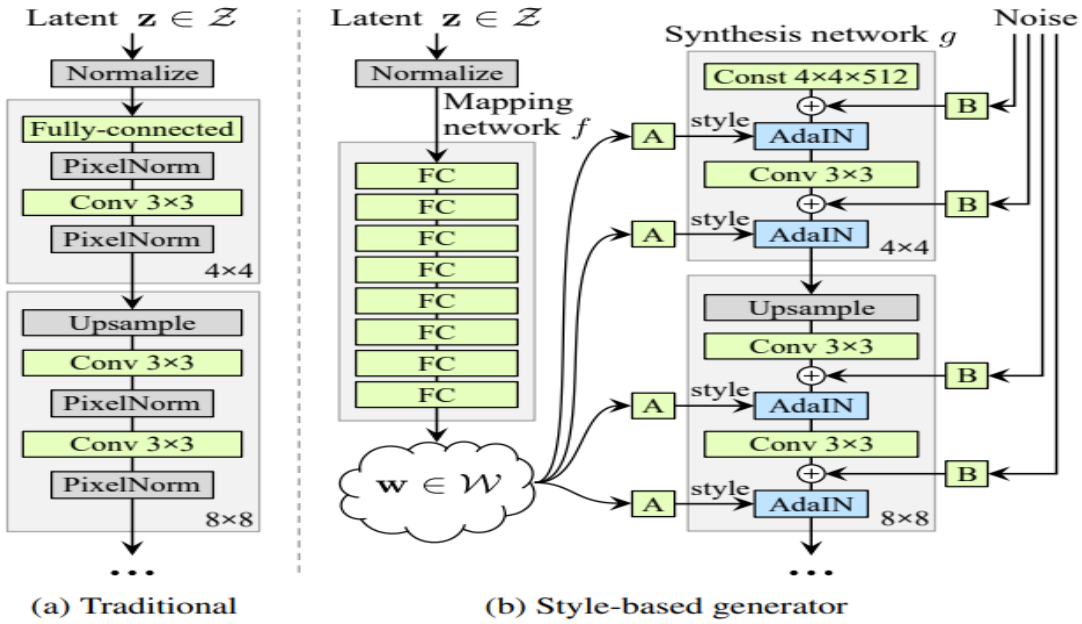


Figure 2.25: StyleGAN architecture [90].

$$\text{AdaIN}(x_i, y) = y_{s,i} \frac{\mathbf{x}i - \boldsymbol{\mu}(\mathbf{x}i)}{\boldsymbol{\sigma}(\mathbf{x}i)} + \mathbf{y}\boldsymbol{b}, \boldsymbol{i}, \ldots \quad \ldots\ldots\ldots (2.24)\ [89]$$

The AdaIN formula is shown above, with x coming from the conv net and y coming from the left side network. After normalizing x, y(s, I) is used for scaling, and y(b, I) is used for the transformation as a bias [90],

as shown in the equation (2.29). Figure (2.26) show examples for performance of StyleGan images and the ovals represent the forgery evidence [91].

Figure 2.26: StyleGAN Image [91].

2.11 Ensemble methods [92]

Ensemble methods are strategies of training several learning algorithms, where in combination produces significantly more high accuracy outcomes than single classifier. Standard methods comprise boosting, stacking, bagging, and a combination of base learners .

• Boosting takes the trained model on data then gradually creates new models by concentrating on the classifying errors produced by the prior model, for example, XGBoost, where it is effective in the implementation of gradient boosting.

• Stacking feeds the models set outputs into another arbitrary algorithm to combine the outputs of used learners then construct the final prediction.

• Bagging applies using random sample sets to train models, then the models votes for classification based on equal weight. For Example, The bagging approach is used in Random Forest to enable selecting of features set randomly at every internal node.

• The combination handles the models predictions and sums them to produce a weighted or simple average.

2.12 Transfer Learning

Transfer learning is a technique in which developing models to be as the basis of models for different tasks, because it has the capability to provide perfect accuracy even with a small dataset, therefore this approach is popular in the deep learning field. The initial training is usually accomplished on a big data set, such as ImageNet. ImageNet is commonly used for training such deep learning models. It has millions of images under thousands of different categories. Several of the transfer learning models available such as VGG net, ResNet, Inception net, Densenet, and Xception net [93].

2.13 Image Segmentation

There are two primary types of image segmentation methods or techniques. layers Based methods and block based methods. The methods of Layer Based Segmentation. The Layer based model used for image segmentation and object recognition, it combines a set of object detectors for the creation of a mask-like structure and explains the appearance, as well as depth order, evaluating both class and instance segmentation. On the other hand, the methods of block based segmentation, are based on several different attributes of the image. This could be color information used for constructing histograms, or texture information, or pixels information that indicate borders or edges [94]. The simplest method of the block based method starts by partitioning the input image into overlapped or non-overlapped blocks that can be circular or rectangular, then extracting characteristics of each block. Finally, the retrieved characteristics are sorted or organized, using a convenient data structure [95].

2.14 Color Systems

The physical process that occurs when combining shades or colors is the basis for the color system. Colors may be created in various methods,

including painting, digital images on a computer or phone screen, and printed pictures.

There are several different types of color systems, and will go over the most important ones.

- RGB

The RGB color system is a set of color models that blend red, green, and blue light in a variety of ways to create a wide spectrum of colors. Each of the R, G, and B components has a value between 0 and 255. The name of the system is formed from the initials of three colors: Red, Green, and Blue. The RGB color model is generally used in electronic devices such as televisions and computers to sense and display images, but it is also employed in traditional photography [96].

- YCbCr

The YcbCr color system, which was determined by the International Telecommunication Union, for utilized for digital video and is extensively used for television in Europe [67]. The Y in YcbCr represents the luminance component with values in range 16 to 235, and Cb and Cr represent the chrominance factors with values in range 16 to 240 for each [97]. The YcbCr color system was chosen in this thesis after performing many tests in the RGB, HSV, lab, and other color systems.

- HSV

HSV is a cylindrical color model, the dimensions are hue, saturation to represents chrominance channels, and value lighting component. The Hue comes with values around 0 to 360, Saturation S value, and value V with values in the range of 0 to 100 [98].

- Lab

A Lab color system is a family from color space, the publication in the CIE (international commission on illumination). The base of the Lab color system is the spectral sensitivity of the human visual system. The L in Lab refers to the lightness channel with value 0 for black to 100 for white, The color channels are represented by the letters a and b. with values -128 to 128, where a channel refers to red for -128 to green for 128, and b channel refers to blue for -128 to yellow for 128 [99].

2.15 Evaluation Measures

Evaluating performance and efficiency is an important aspect when designing a machine learning model. In order for the machine learning model to be reliable, an evaluation tool must be chosen to commensurate with the nature of the model's work. Often, when evaluating machine learning models more than one scale is used to ensure correct evaluation of the model. The evaluation measures in machine learning are divided into three main types: the measures that use to evaluate classification tasks, clustering tasks, and regression tasks [100].

2.5.1 Evaluation Measures for Classification Tasks

There are several kind of evaluation measures based on the tool of Confusion_Matrix, are available for Classification tasks; such as Accuracy, Recall, Precision, and F1 Score[100].

a. Confusion Matrix (CM):

CM is one of the most important tools that give a comprehensive description of the performance of the classification model. Figure (2.27) show the confusion matrix for a binary classification model [100].

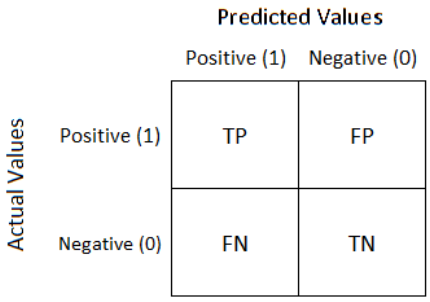


Figure 2.27: The confusion matrix [100].

Each prediction it will belong to one of the four categories [100] **:**

- True positive (TP): correct positive prediction.
- False positive (FP): incorrect positive prediction.
- True negative (TN): correct negative prediction.
- False negative (FN): incorrect negative prediction.

b. Accuracy measure

The accuracy is referred to as the average of right predictions to the aggregate of the input samples as shown in equation (2.30). The matrix accuracy can be calculated by computing the average of the main diagonal values [100] according to the equation (2.25).

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+FP+FN+TN}} \quad \ldots\ldots\ldots\ldots \text{(2.25) [100]}$$

c. Recall

The recall is the ratio of right positives to the total of right positives and erroneous negative measures, it is used to measure the classifier's ability to detect all positive cases, and it is one of the most important metrics used with models that contain unbalanced dataset [100]. This metric calculated according to the equation (2.26).

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \quad \text{..... (2.26) [100]}$$

d. Precision

Precision is defined as a classifier's ability to not label a negative instance as positive and described by the ratio of right positives to the total of right and erroneous positives [101]. Precision metric calculated according to the equation (2.27).

$$\mathbf{Precision} = \frac{\mathbf{TP}}{\mathbf{TP+FP}} \quad \text{.....(2.27) [101]}$$

e. F1-score

The F-score measure can be represented as a united mean of recall and precision. The F1 tries to balance between recall and precision [101], and use to measure a test's accuracy which determines the precise of how many instances it classifies correctly and robust by cannot ignoring a significant number of instances of the model. F1 Score has [0, 1] ranges and the greater value indicated to the better performance. The equation (2.28) represents the F-score measure [100].

$$\mathbf{F_1\ score} = \frac{\mathbf{2\times Precision\times Recall}}{\mathbf{Precision+Recall}} \quad \text{..... (2.28) [100]}$$

Chapter Three

The Proposed System

3.1 Overview

The proposed system passes through several stages to be capable to detect real and fake images then detect the faked partitions. These stages are resizing the images, convert the image color system to YCbCr system color, training two deep convolutions neural networks to ensemble them for classifying the image to real or fake, training a third neural network to localize the faked partitions, and testing phase. This chapter explains the main concepts and algorithms related to the model stages, besides the cause of choosing the dataset that is used in the system.

3.2 The Proposed system

The proposed system includes five phases as follows: pre-processing, preparing the dataset, extracting image features, decision making, and postprocessing. The first phase preprocessing before training the data includes resizing the images, converting the image color system to an YCbCr color system, and make another dataset for image pa. The second phase is preparing the dataset includes splitting the original dataset and partitions dataset of the images into training, validation, and testing. The third phase is feature extraction, based on two deep CNNs of pretrained and modified Xception net, to represent the image as a vector for the features by learning the networks on the features of the images for each class in the training phase to make predictions for the images of testing set based on the learned features. Then training Localization CNN to learn the network the features of the partitions of the images to detect the faked partitions of the fake images. The fourth phase is to make an Average Ensemble for each prediction in the testing phase to make a decision within the classifications whether the image is real or fake. The fifth phase is a postprocessing step after the decision that the image is

fake, to localize the faked partitions of the fake image using the learned Localization CNN, as shown in Figure (3.1) and Algorithm (3.1)

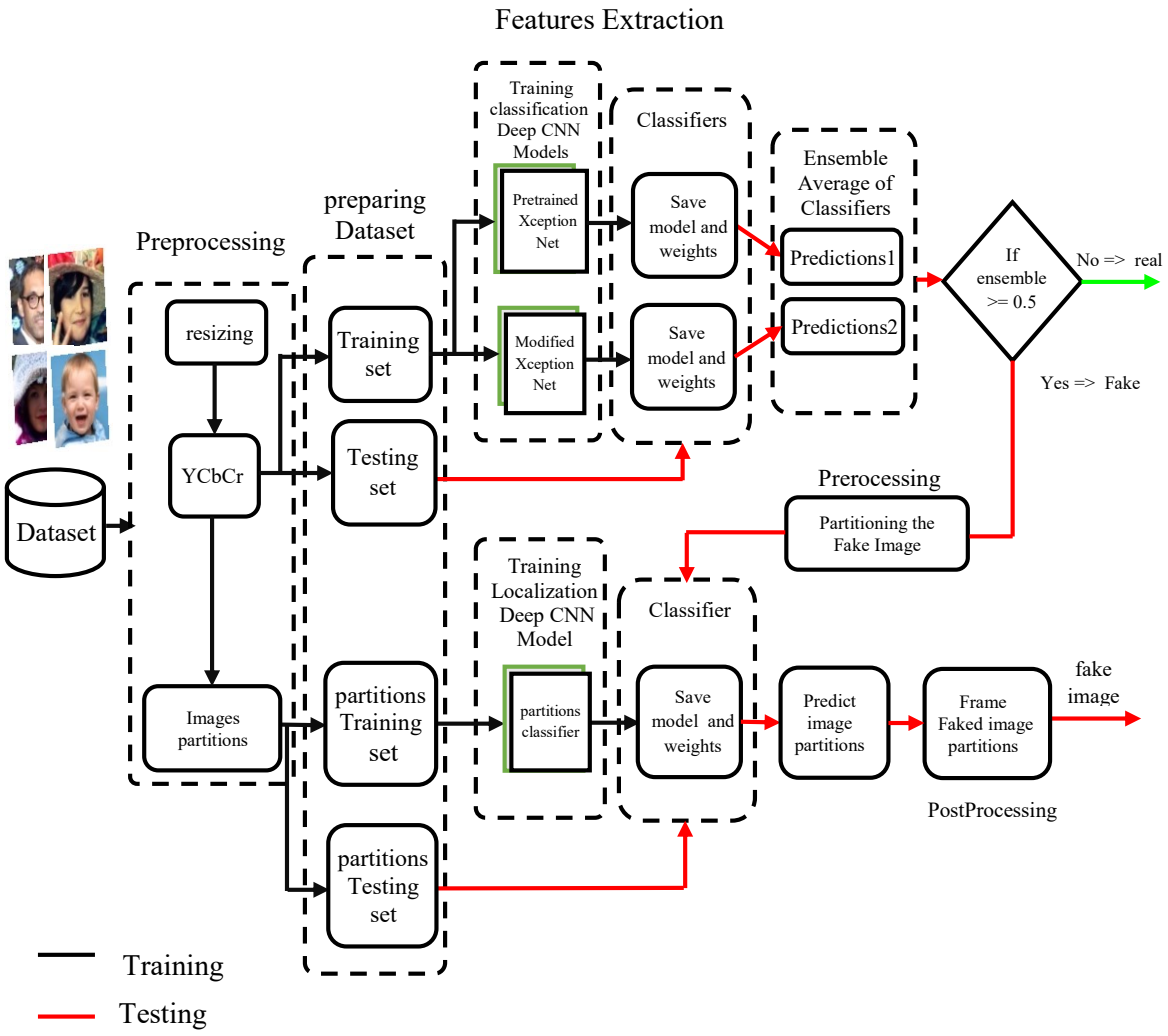


Figure 3.1 the proposed System.

Algorithm 3.1: The proposed System
Input: face dataset
Output: Decision of Real or Fake , Frame the faked partitions of forged image

step1: call pre-processing algorithm

- Reduce image dimensions to 128*128
- Convert color system to YCbCr
- partitioning dataset images to make partitions dataset

step2 : preparing dataset (face dataset)

- Splitting images dataset to Training, Testing, and Validation
- Splitting images partitions dataset to Training, Testing, and Validation

step3: call training algorithm.

- Training the Pre-trained Xception net on the features of images training set.
- Training the modified augmented Xception net using random weights on the features of images training set.
- Training localization Deep CNN using random weight on the images partitions features of the training set based on 64 partitions for each image of 16*16 pixels for each partition .

// step4, step5, step6, and step7 are Testing phase

step4: Testing algorithm.

- Extract the Features of the testing set as vector using pre-trained Xception model
- Extract the Features of the testing set as vector using the randomly trained modified Xception net model

step5: Call Ensemble method.

- Calculate the average of predictions for each image classification models.
- Compare the result with threshold of 0.5
- Return decision fake if ensemble >=0.5 else return decision real.
- If decision fake
  - partitioning the faked image
  - Extract the features of each partition of fake image to detect the fake partitions using Localization CNN model

Step6: postprocessing

- Framing the faked partitions of the fake image
- View the localized faked image

Step7: Evaluate the models **END**

3.2.1 The dataset

The machine learning model relies on disparate data and resulting accuracy. The process of searching for meaningful and useful data for forgery forensic detection may be somewhat difficult. There have been many previous studies based on datasets from various generative adversarial network architectures and many of the used GAN network architectures resulting images can be detected with the naked eye as a fake. Therefore, the choice was for a newer and more difficult dataset. Therefore, The used dataset is "140k Real and Fake Faces" [33] in this thesis, where the generating of the fake images based on the StyleGan network, where most of the fake images are difficult to detect with the human vision system, as shown in Figure (3.2).

figure 3.2 samples of dataset A: real images B: fake images.

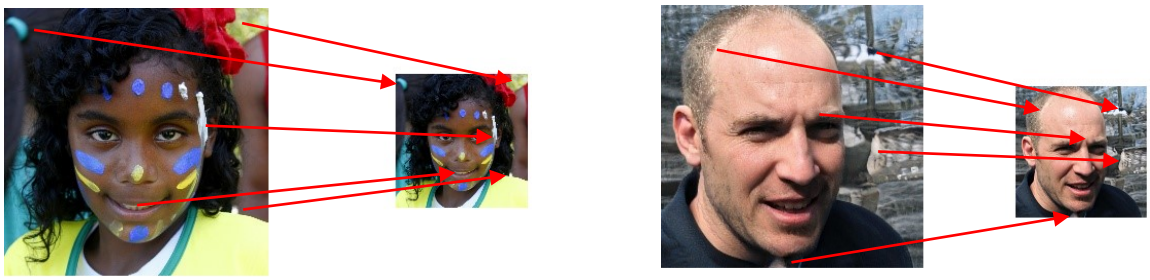
3.2.2 Pre-processing Dataset

The pre-processing phase includes different stages. The preprocessing phase returns a list of resized images of YCbCr color system to classify the images as fake or real. In addition to another list for partitions of each image

to train the images partitions classifier on the partitions of the training set of images partitions classifier, Besides a lists of classes label for these image lists. The testing phase returned the list of resized images of YCbCr color system, in addition to the labels list. On the other hand, the partitioning process implemented after the image classifiers detect the image as fake as explained in the steps in Algorithm (3.2).

a. images resizing:

In this step, the RGB image of the dataset will be resized from 256 * 256 to 128 * 128 pixels. Where it is considered an appropriate size, it is not very expensive from the computational complexity, besides that, it preserves the image details from the Decay, which shown in the arrows of some small detailes in Figure (3.3) A and B.



(A) a Real image (B) a Fake image

Figure 3.3 : Reducing Image Dimensions image step.

b. Converting color system

This step includes converting the color system of the images from RGB to YCbCr color system, it gave a better than RGB and other color systems by obtaining the evidence of Forgery by taking evidence of forgery in the Luminance part (Y Channel), in addition to the evidence of forgery in the chrominance part (Cb And Cr channels), where notice the lack of fusion in some places of the fake image as shown in Figure (3.4) of a fake image. Where the fusion appears in most parts of the real image although with

different colors in the image which shown in the ovals of some inconsistent areas in the channels of images in Figure (3.5), which appear in YCbCr more than RGB color system.

RGB

YcbCr

Figure 3.4 RGB to YcbCr conversion for a faked image with channels.

RGB

YcbCr

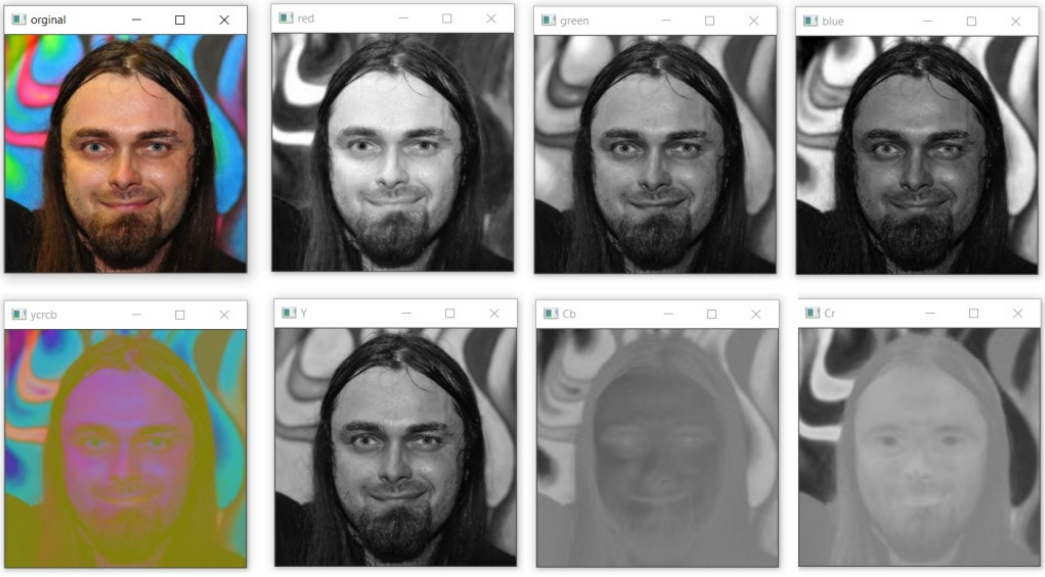


Figure 3.5 RGB to YcbCr conversion for a real image with channels.

c. Image segmentation

In this step, the images in the dataset of "140K Real and Fake Faces" is divided to 8 columns and 8 rows to be 64 partitions based on 16 * 16 pixels for each partition to be the total number of partitions 8960000 partitions, and each of these partitions holds the original image label as shown in Algorithm (3.3), then to train the Localization network of classifying the partitions, but when implementing the overall system, then this step is implemented after detecting the image as a fake. As presented in Algorithm (3.2).

Algorithm 3.2: Preprocessing phase
Input: images dataset
Output: List of images dataset, list of images partitions, List of images class labels, list of partitions class label

```
1: for each image in training and validation sets
2:      read one RGB image at a time
3:      resize the RGB image to 128*128 pixels
4:      convert the image color system From RGB to YCbCr
5:          for i= 0 to 128  i+=16 //number of columns
6:            for j= 0 to 128  j+=16 // number of rows
7:                 partition = image([ i , j ] , [ i+16 , j+16 ])
8:                append  partition to  partitions list
9:                append image label to  partitions labels list
10:           End for-j
11:         End for-i
12:    append the image to image list
13:    append label 1 for fake images and 0 for real   images to labels list
14:End for
15:for each image in testing set
16:      read one RGB image at a time
17 :     resize the RGB image to 128*128 pixels
```

```
18:     convert the image color system From RGB to YCbCr
19:     append the image to image list
20:     append label 1 for fake images and 0 for real   images to labels list
21:     call image classifiers
22:     if label=1
23:         for i= 0 to 128  i+=16 //number of columns
24:             for j= 0 to 128  j+=16 // number of rows
25:                 partition = image([ i , j ] , [ i+16 , j+16 ])
26:                 append  partition to  partitions list
27:                 append image class label to  partitions labels list // 1 for fake, 0 for real
28:             End for-j
29:         End for-i
30:     Endif
31:Endfor
End
```

3.2.3 Preparing Dataset for training purposes

Preparing the dataset is an important step, where the images are selected randomly. Therefore, sometimes the selected data cannot give good learning for the networks. In any case, the data is split into three main sets, training of 80%, validation of 5%, and testing of 15% for neural network models of classifying the images into a real or fake image as shown in Figure (3.6), and the image partitions dataset is split to 85% for training, 5% for validation, and 10% for testing for localization neural network model as shown in Figure (3.7).

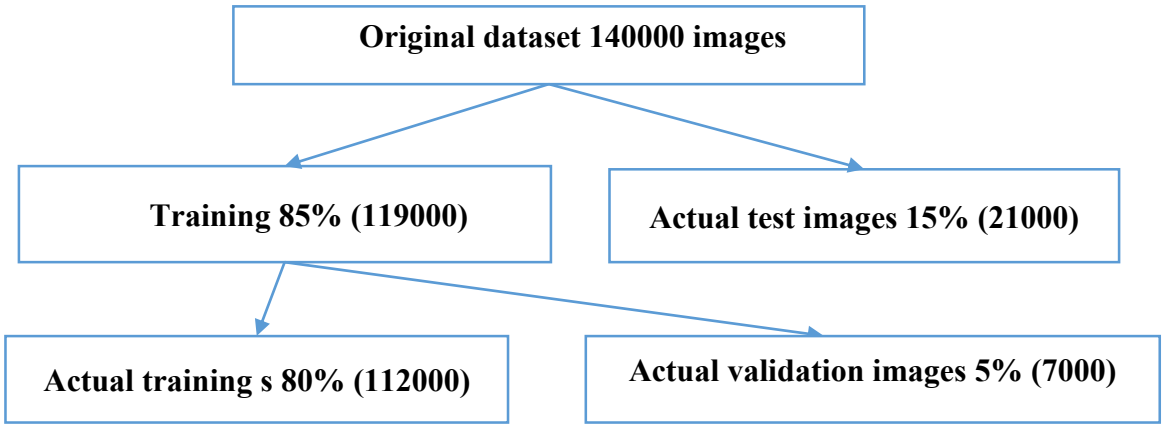


Figure 3.6 preparing dataset for image classification.

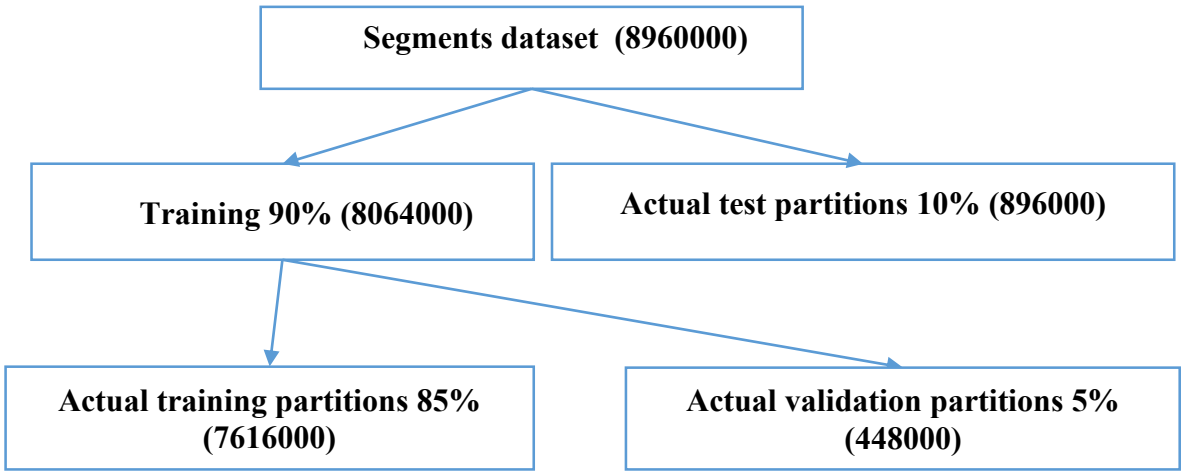


Figure 3.7 preparing image partitions dataset.

3.2.4 Features Extraction

The features extraction phase uses three networks, two for image classification, and one for partitions classification. The first network is pre-trained Xception net, and the second is a Modified Trimmed non trained Xception net, and it is trained on random weights to extract and learn the networks on features of the images dataset. On the other hand, the third network was trained on the partitions of the image of training set. The third network is used in the testing phase after the previous two networks detect the fake image to classify the image partitions to frame the detected fake parts of a fake image.

A. Pretrained Xception Net

The first network depends on transfer learning of Xception net with Global_ Max_Pooling to extract robust feature maps of the images for better accuracy. It consists of 14 models distributed on 7 blocks, where the fifth block is repeated 8 times. The first block is of the network based on Conv2D, and the rest blocks of the network mainly depend on SeparableConv2D with filter size 3*3. Each of the convolutional layers precedes the "Rectified Linear Unit" (Relu) Activation function layer, and each convolution layer follows the BatchNormalization layer. There are max_pooling2d at the end of each block except the first block. In addition to Conv2D layer with filter 1*1 and stride 2*2 to create residual data except for the first and last block without residual data. At the end of the blocks, there are operations of adding the residual data with the output of the MaxPooling2D layer.

B. Modified Trimmed non trained Xception Net (MTXception)

The second network is similar to the first network, with some modification as bounded in red color in Figure (3.8) at the first block depend on SeparableConv2D layer instead of Conv2D, the fifth model not repeated as the first network, and at the end of the net by replacing the last Globalpooling2D layer with MaxPooling2D of (2,2) output feature map instead of GlobalPooling layer. Then the MT-Xception net consists of 7 blocks distributed on 7 models. Then to increase the feature map of images, therefore the output of the MaxPool2D layer of shape 2*2 feature map will be as input to three layers globalmaxpooling2D, globalaveragepooling2D, and Flatten layers as the concept of Densenet network. The feature maps extracted from these Layers will combine to produce a vector of feature maps for each image as shown in Table 3.1. The MTXception is better than pretrained Xception in accuracy, less a parameters number with accuracy of 99.93, total parameters 9539217, and time of execution with 420 seconds per epoch,

compared to accuracy of 99.89, parameters 20873769, and 670 seconds for the pretrained Xception.

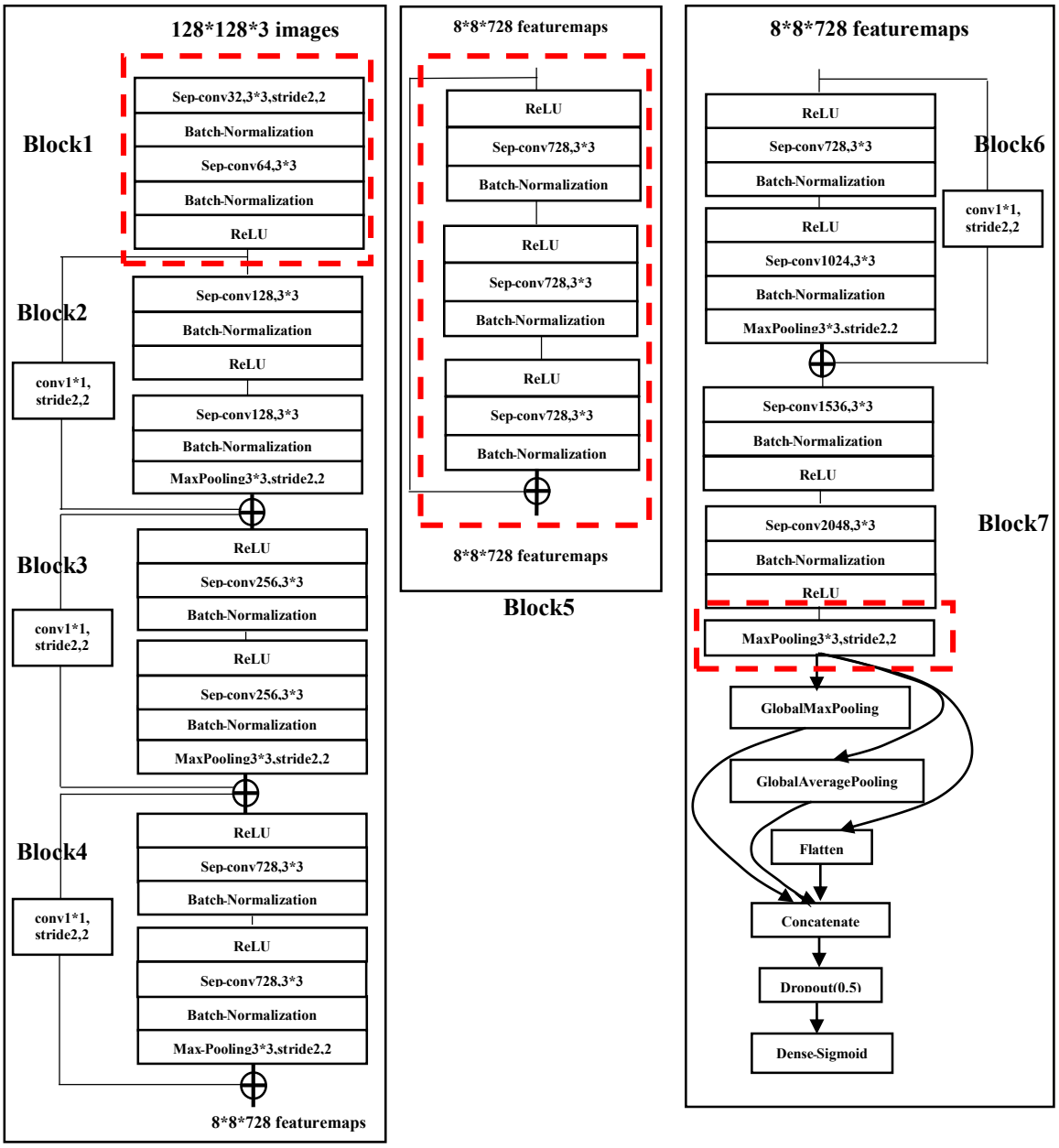


Figure 3.8 MTXception net Architecture.

Table 3.1 The details of MTXception net.

Block number	Layer	Layer information	image / map shape	Parameters number
Input image			(128,128,3)	0
1	Separable-Convolution-2D	32,filter-size(3,3), stride(2,2)	(63, 63, 32)	123
	Batch-Normalization		(63, 63, 32)	128
	Activation	ReLU	(63, 63, 32)	0
	Separable-Convolution-2D	64, filter-size(3,3)	(61, 61, 64)	2336
	Batch-Normalization		(61, 61, 64)	256
	Activation	ReLU	(61, 61, 64)	0
residual	Convolution-2D (residual)	128,filter-size(1,1), stride(2,2),padding=same	(31, 31, 128)	8768
	Batch-Normalization	Axis=0	(31, 31, 128)	512
2	Separable-Convolution-2D	128,filter-size(3,3)	(61, 61, 128)	17536
	Batch-Normalization		(61, 61, 128)	512
	Activation	ReLU	(61, 61, 128)	0
	Separable-Convolution-2D	128,filter-size(3,3)	(61, 61, 128)	17536
	Batch-Normalization		(61, 61, 128)	512
	MaxPooling-2D	(3,3), stride(2,2)	(31, 31, 128)	0
	add	Pooling + residual	(31, 31, 128)	0
residual	Convolution-2D (residual)	256,filter-size-(1,1), stride(2,2),padding=same	(16, 16, 256)	8192
	Batch-Normalization	Axis=0	(16, 16, 256)	512
3	Activation	ReLU	(31, 31, 128)	0
	Separable-Convolution-2D	256,filter-size(3,3)	(31, 31, 256)	33920
	Batch-Normalization		(31, 31, 256)	1024
	Activation	ReLU	(31, 31, 256)	0
	Separable-Convolution-2D	256,filter-size(3,3)	(31, 31, 256)	67840
	Batch-Normalization		(31, 31, 256)	1024
	Max-Pooling-2D	(3,3),stride(2,2)	(16, 16, 256)	0
	add	Pooling + residual	(16, 16, 256)	0
residual	Convolution 2D (residual)	768,filter-size(1,1), stride(2,2),padding=same	(8, 8, 728)	32768
	Batch Normalization		(8, 8, 728)	1024
4	Activation	ReLU	(16, 16, 256)	0
	Separable-Convolution-2D	728,-filtersize(3,3)	(16, 16, 728)	188672
	Batch-Normalization		(16, 16, 728)	2912
	Activation	ReLU	(16, 16, 728)	0
	Separable-Convolution-2D	768,filter-size(3,3)	(16, 16, 728)	536536
	Batch-Normalization		(16, 16, 728)	2912
	Max-Pooling-2D	(3,3),stride(2,2)	(8, 8, 728)	0
	Add	Pooling + residual	(8, 8, 728)	0
residual	Convolution-2D (residual)	768,filter-size(1,1), stride(2,2),padding=same	(4, 4, 1024)	186368
	Batch-Normalization		(4, 4, 1024)	4096

5	Separable-Convolution-2D	728,filter-size(3,3)	(8, 8, 728)	536536
	Batch-Normalization		(8, 8, 728)	2912
	Activation	ReLU	(8, 8, 728)	0
	Separable-Convolution-2D	728,filter-size(3,3)	(8, 8, 728)	536536
	Batch-Normalization		(8, 8, 728)	2912
	Activation	ReLU	(8, 8, 728)	0
	Separable-Convolution-2D	728,filter-size(3,3)	(8, 8, 728)	536536
	Batch-Normalization		(8, 8, 728)	2912
	add	Residual+ BN block5	(8, 8, 728)	0
residual	Convolution-2D (residual)	1024,filter-size(1,1), stride(2,2),padding=same	(4, 4, 1024)	745472
	Batch Normalization	Axis=0	(4, 4, 1024)	4096
6	Activation	ReLU	(8, 8, 728)	0
	Separable-Convolution-2D	728,filter-size(3,3)	(8, 8, 728)	536536
	Batch-Normalization		(8, 8, 728)	2912
	Activation	ReLU	(8, 8, 728)	0
	Separable-Convolution-2D	1024,filter-size(3,3)	(8, 8, 1024)	752024
	Batch-Normalization		(8, 8, 1024)	4096
	MaxPooling-2D	(3,3),stride(2,2)	(4, 4, 1024)	0
	add	Pooling + residual	(4, 4, 1024)	0
7	Separable-Convolution-2D	1536,filter-size(3,3)	(4, 4, 1536)	1582080
	Batch-Normalization		(4, 4, 1536)	8192
	Activation	ReLU	(4, 4, 1536)	0
	Separable-Convolution-2D	728,filter-size(3,3)	(4, 4, 2048)	3159552
	Batch-Normalization	Axis=0	(4, 4, 2048)	8192
	Activation	ReLU	(4, 4, 2048)	0
	Max-Pooling-2D	(3,3),stride(2,2)	(2, 2, 2048)	0
	Global-max-pooling2d	(2,2),stride(2,2)	(2048)	0
	Global-average-pooling2d	(2,2),stride(2,2)	(2048)	0
	Flatten	Max Pooling-2D	(2048)	0
	Concatenate	Global-max-pooling2d Global-average-pooling2d Flatten	(12288)	0
	Dropout	0.5	12288	0
	Dense	activation="sigmoid"	1	12289

Total params: 9531436.

Trainable parameters: 9506028.

Non-trainable parameters: 25408.

C. Segments Localization Network (PLNetwork)

The feature extraction part uses Deep 2D CNN models to extract the feature of the partitions of the images. This network begins with a 2D convolution layer with a (ReLU) activation function, then a

BatchNomalization layer. This convolution is followed by eight Separable convolution layers with a (ReLU) activation function. The following of each convolutional layer is one BatchNormalization layer, and between each two Separable convolution layers, there is a max-pooling layer with pooling size (2,2). A number of convolution filters used with convolutional layers are (32, 32, 64, 128, 256, 512, 512, 1024, 2048) respectively with kernel size (3,3), Except for the first and last layer, use kernel with (1,1), and the eighth separable layer use kernel with (2,2). Then the network ends with maxpooling2D, Flatten, and dense layer with sigmoid activation function to learn the network on classes of the partitions based on extracted features from the convolutional layers. Table 3.2 shows the model layers and Figure (3.9).

Table 3.2 overall details of Model3 p classifier.

Block number	Layer	Layer information	image / map shape	Parameters number
Input partiotion			(16,16,3)	0
1	Convolution-2D	32,filter-size (1,1),Relu	(16, 16, 32)	96
	Batch-Normalization		(16, 16, 32)	128
	Separable-Convolution-2D	32,filter-size (3,3),Relu	(16, 16, 32)	1312
	Batch-Normalization		(16, 16, 64)	128
	Separable-Convolution-2D	64,filter-size(3,3),Relu	(16, 16, 64)	2336
	Batch-Normalization		(16, 16, 64)	256
	MaxPooling-2D	(2,2),stride(2,2)	(8, 8, 64)	0
2	Separable-Convolution-2D	128,filter-size(3,3),Relu	(8, 8, 128)	8768
	Batch-Normalization		(8, 8, 128)	512
	Separable-Convolution-2D	256,filter-size(3,3),Relu	(8, 8, 256)	33920
	Batch-Normalization		(8, 8, 256)	1024
	MaxPooling-2D	(2,2),stride(2,2)	(4, 4, 256)	0
3	Separable-Convolution-2D	512,filter-size(3,3),Relu	(4, 4, 512)	133376
	Batch-Normalization		(4, 4, 512)	2048
	Separable-Convolution-2D	512,filter-size(3,3),Relu	(4, 4, 512)	266752
	Batch-Normalization		(4, 4, 512)	2048
	MaxPooling-2D	(2,2),stride(2,2)	(2, 2, 512)	0
4	Separable-Convolution-2D	1024,filter-size(2,2),Relu	(2, 2, 1024)	526336
	Batch-Normalization		(2, 2, 1024)	4096
	Separable-Convolution-2D	2048,filter-size(1,1),Relu	(2, 2, 2048)	2098176
	Batch-Normalization		(2, 2, 2048)	8192
	MaxPooling-2D	(2,2),stride(2,2)	(1, 1, 2048)	0
	Flatten	Max Pooling-2D	(2048)	0
	Dense	activation="sigmoid"	1	2049

Figure 3.9 SLNetwork Architecture.

The details of the layers in the three used networks can be represented as follows:

- **Convolution Layer**

According to the background of theoretical presented in Chapter Two Section 2.9 of the convolution process used in the networks, it can be represented as an algorithm. Suppose the input image has i,j indices, and received k weights Kernel W of size k1*k1 of m,n indices, and it is move

with strides S1,S2, the convolution process needs 4 nested loops for 2D convolution shown as following in the Algorithm (3.3).

Algorithm 3.3: The convolution layer.
Input: Image img(D1,D2), filter size(k1,k1).

Output: The feature maps of image.

```
1.  For k from 1 to no. of filters
2.      Initialize Kernal (W)
3.      b equal to 1  // b represent bias which is a constant
4.      For i from 1 to row D1  // stride =S1
5.          For j from 1 to column D2 // stride =S2
6.              net=0 , fr =1 fc=1 // net= initial feature map , fr, fc = initial filter dimensions
7.              For m from i to i +k1     // rows and cols. of kernel, k1 specify kernel size
8.                  For n from j to j+k1   // d=1 k1 equal to kernel of size 3*3
9.                      net = net +[ img(m,n) * W(fr,fc) + b]
10.                     fr= fr +1
11.             End-n
12.             fc= fc +1
13.         End-m
14.         F(net) = Max(net, 0)   // Relu activation function
15.         Feature-map[i,j,k] = f(net)
16.     End-j
17.   End-i
18. End-k
19. End
```

Depthwise separable convolution

A depthwise separable convolution is comprised of two convolutional phases depthwise and pointwise convolutions. Where depthwise focuses on spatially relationship mapping with Kernels of 2D convolutions, one kernel for every channel in the input map, and then the pointwise convolution focuses on cross-channel relationship modeling as shown in

Algorithm (3.4) with n ×1 convolution for the output of the pointwise convolution where n is a number of channels. This factorization form, denoted by depthwise and pointwise convolutions.

Algorithm 3.4: The depthwise separable convolution layer.

Input: Image img(D1,D2), filter size(k1,k1).

Output: The feature map of image.

```
b equal to 1  // b represent bias which is a constant
For c from 1 to no. of channels
  Initialize Kernal (W) 2D kernel of size
    For i from 1 to row D1  // stride =S1
      For j from 1 to column D2 // stride =S2
        net=0 , fw =1 fc=1 // net= initial feature map , fr, fc = initial filter dimensions
        For m from i to i +k1     // rows and cols. of kernel, k1 specify kernel size
          For n from j to j+k1   // d=1 k1 equal to kernel of size 3*3
            net = net +[ img(m,n) * W(fr,fc) + b]
            fr= fr +1
          End-n
          fc= fc +1
        End-m
    F(net) = Max(net, 0)   // Relu activation function
    Feature-map[i,j,c] = f(net)
    End-j
  End-i
End-c
For k from 1 to no. of filters
   Initialize point_wise_Kernal(W) // 1D Kernel
   net=0
   For i from 1 to length.row Feature-map
      For j from 1 to length.column Feature-map
         For c from 1 to length.channels
                net = net +[ Feature-map (i,j) * W(c) + b]
                net = Max(net, 0)   // Relu activation function
```

```
27.                    net = F(net)
28.                    Batch normalize the net //according to equation (2.28).
29.              End-c
30.              Output_FeatureMap[i,j]=net
31.          End-j
32.      End-i
33. End-k
34. End
```

- **Residual Data**

The residual layer contains convolution2D with a kernel (1,1) and stride (2,2). The benefit of this layer is to keep some data from the input map to add it to the output of the pooling layer as shown in Figure (3.10). This data prevents vanishing the data of input map data to get better results. When removing the residual blocks from the networks of this thesis cause reducing the accuracy is by about 10%, where get an accuracy of more than 89%.

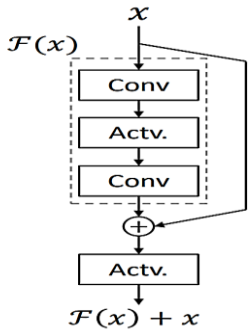


Figure 3.10 Residual Layer [102].

- **Batch-Normalization (BN)**

The Batch normalization layers perform stable generalization improvement, optimization and increase the performance and learning of training neural network models. It keeps the mean activation close to zero

and standard deviation close to one batch normalization equation presented in equation (2.28) in section 2.9.3.d

- **Max Pooling Layer**

The process of pooling is a sample-based operation. The target of the pooling layer reduces the input dimensions between the layers, and it also decreases the computational cost by reducing the learning parameters. The MaxPooling level is in charge of making the characteristics sturdy to noise by selecting the robust features. The MaxPooling operation explained in Algorithm (3.5).

Algorithm 3.5: Max_ pooling layer.
Input: input_map
Output: lower output_map

```
1. For i = 1 to feature_map.rows
2.     For j =1 to feature_map.columns
3.         Max = feature_map(i,j)
4.               For  m = i to i+1
5.                     For n = j to j+1
6.                       If  feature_map(m,n) > Max
7.                            Max = feature_map(m,n)
8.                       End-n
9.                End-m
10.         output_map[L1, L2] = max // add result to index output_map
11.         L2 = L2 + 1
12.      End-j
13.   L1 = L1 + 1
14. End-i
15. End
```

- **Global Max Pooling**

GlobalMaxPooling is another kind of pooling process. Where the size of the pool is set to the entire input map size, therefore that the maximum value of the whole input is calculated as the one output value as shown in Figure (3.11) and Explained in Algorithm (3.6).

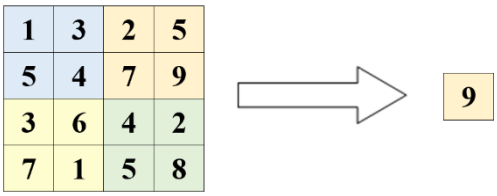


Figure 3.11 Global Max Pooling operation.

Algorithm 3.6: Global Max_ pooling layer.

Input: input_map

Output: output_map

1. For i = 1 to feature_map.rows
2. For j =1 to feature_map.columns
3. If feature_map(i,j) > Max
4. Max = feature_map(i,j)
5. End j
6. End i
7. output_map = max *// result output_map*
8. Return output_map
9. End

- **Global Average Pooling**

GlobalAveragePooling process, the size of the pool as well is set to the entire input map size, but it computes one value of the average of the input map as shown in Figure (3.12) and Explained in Algorithm (3.7)

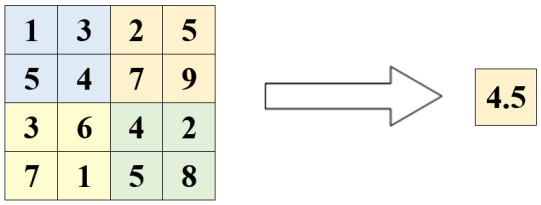


Figure 3.12 Global Average Pooling operation.

Algorithm 3.7: Global Average_ pooling layer.

Input: input_map

Output: output_map

1. AVG=0 // initial value of avarage
2. For i = 1 to feature_map.rows
3. For j =1 to feature_map.columns
4. AVG = AVG+feature_map(i,j)
5. End-j
6. End-i
7. output_map= AVG / (feature_map.rows * feature_map.columns) // *result of output_map*
8. Return output_map
9. End

- **Fully connected layer**

Fully connected layers basically harmonize with the convolution layer. All neurons of the fully connected layer is densely connected to the neurons of the previous and next layer. In a typical CNN, full-connected layers are usually placed at the end of the network architecture, but not mean doesn't placed in the hidden layers.

3.2.5 Training Phase

the pretrained Xception net and Nontrained MTXception net are trained on the features of the real images and the fake images from the training set to

extract the weights of each network in order to build the classification models based on training data to use them in matching the images of the test set. On the other hand, the third proposed network of partitions classification is trained on the features of the partitions of the real and fake images of the training set to extract weights in order to build the faked images detection model as shown in Figure (3.13) and Algorithm (3.8) and (3.9) to use it in the testing phase.

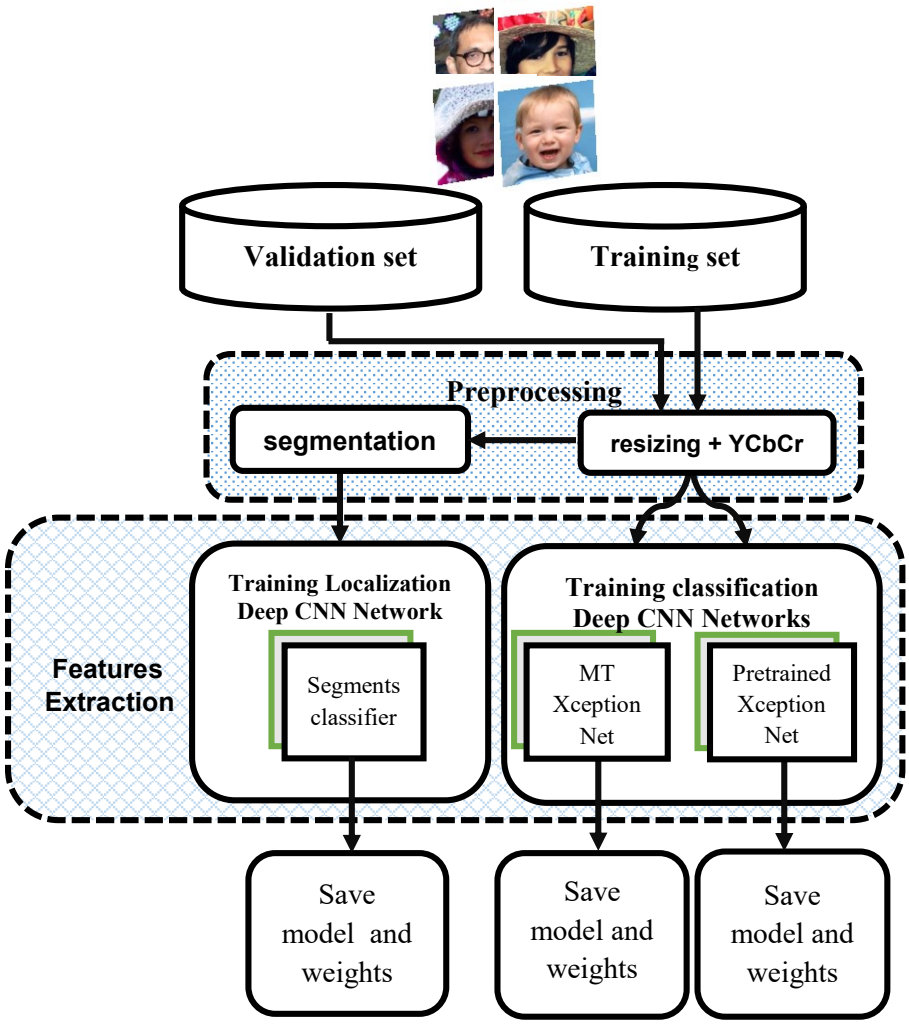


Figure 3.13 Training phase.

Algorithm 3.8: Training phase of image classification models

Input: training images, training labels.

Output: forgery classification models.

Step 1: read one YCbCr image at a time.

Step 2: Extract features vector of images using pretrained Xception neural network.

Step 3: every 32 (batchsize) image reweights the connections between neurons of the neural network.

Step 4: Extract features stream of image using modified nontraind Xception neural network.

Step 5: every 32 (batchsize) image reweights the connections between neurons of the neural network.

End

Algorithm 3.9: Training phase of partitions classification model

Input: training images partitions , training partitions labels.

Output: forgery detection model.

Step 1: read one YCbCr image partition at a time.

Step 2: Extract features vector of images partitions .

Step 3: every 256 (batchsize) partitions reweights the connections between neurons of the neural network

End

3.2.6 Testing phase

Testing phase includes at the first step extract the features of the unseen image using the pretrained Xception net and MTXception net and make the predictions for the image based on the weights of the training two image classification models then ensemble the predictions using average to predict the class label of the images either fake or real, then if the image is predicted as fake then the image will partitioned to make prediction for each of the 64 partitions of the fake image using partitions classification model and framing the partition that predicted as faked, as shown in Algorithm (3.10) .

Algorithm 3.10: Testing phase
Input: Testing images.

Output: forged image detection.

1: read one YCbCr image at a time.

2: Extract features vector of image using pretrained Xceptin net

3: predict the decision real or Fake image.

4: Extract features vector of image using MTXception net.

5: predict the decision real or Fake image.

6: Ensemble the predictions *// calculating the average of predictions for image.*

7: Classifying the image to fake or real *// based on threshold 0.5.*

8. if prediction=1 then *// fake image class label*

9. for each partition in image

10. Extract features vector of partition *// using segments classification model*

11. predict the decision real or fake partition

12. if prediction = 1 *// fake partition label*

13. Frame the partition with red color

14. else Frame the partition with green color

15. End for

16.End if

End

3.2.6.1 Ensemble method

The ensemble method was used to get better results, as the aggregation of two classifiers in one predictor gave a high performance compared with the performance of each one classifier separately, where each classifier has a weak point in the classification process. Therefore adopt the ensemble method to classify the images, by calculating the average for the predictions of the classifiers as shown in Algorithm (3.11), which is one of the most common approach to enhance the performance of classifiers on huge and complex data.

Algorithm 3.11: Ensemble Method.

Input: prediction of models

Output: decision of ensemble predictions

1.pred1= model1 prediction *// pretrained Xception net*

2.pred2= model2 prediction *// nontrained MTXception net*

3.if (pred1+pred2)/2 >= 0.5 then

4. enspred= 1 *// for Fake image*

5.else enspred= 0 *// for Real image*

6.End if

7. Return enspred *// final prediction*

End

3.2.7 Post processing

The post-processing process includes localizing and framing by the red color for the partitions detected by the segments classification network as fake and framing the partitions that were undetected by the network by green color to distinguish them as shown in Figure (3.17).

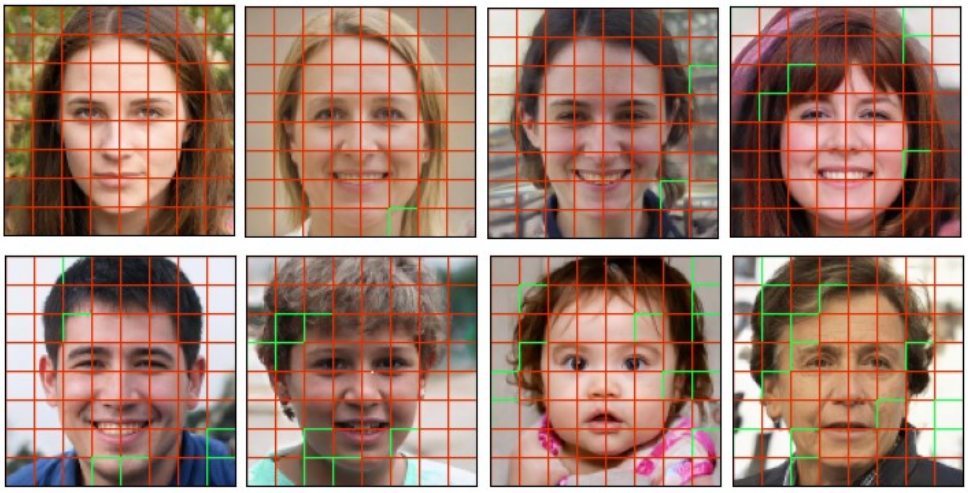


Figure 3.17 samples of Fake images after postprocessing.

3.3 Enhance Learning:

After performing the Ensemble method and getting a high accuracy of %99.99, we thought of another idea of improving learning performance and getting more Accuracy by reducing the cost of training from scratch in resources, time, and effort. Therefore, the learning can start from the end of the last learned model and weights saved in the model based on a smaller learning rate of 0.1 or less than the priorly specified learning rate of the network to reduce the changing on the saved weights.

Chapter Four

Experimental Results and Discussion

4.1 Overview

This chapter includes an explanation of the experiments and results for each step of the proposed model. In addition to explaining the used dataset in the proposed system, the hardware, and software used to implement the proposed system. The results of the stages are sorted as their appearance in Chapter Three.

4.2 System specification

Implementing image processing systems using machine learning with a deep learning approach requires high computer resources to apply flexibly, especially with huge datasets. Therefore, the proposed model was implemented using the following resources:

4.2.1 Hardware

- Central Processing Unit (CPU): Intel(R) Core i7-10750H.
- RAM: 32 GB.
- Graphics Processing Unit (GPU): NVIDIA GTX 1660 TI 6GB.
- Hard Disk: 512 GB

4.2.2 Operating system

- Windows10, 64 bit

4.2.3 Programing language

- Python 3.7 with PyCharm 2021 IDE

4.3 The experimented dataset

This thesis uses the dataset mentioned in the previous chapter in Section 3.2.1 when training and testing the proposed model, the proposed-system uses a dataset (140 Real and Fake Faces). Which consists of 70000 REAL faces collected from the Flickr dataset, as well as 70000 fake faces sampled from the 1 Million fake face imges generated by StyleGAN of dimensions 1024*1024. The provider combined both datasets then resized the images into 256*256 px with extension jpeg.

The number of images of this dataset is large, and the quality of forgery is high. Therefore, it satisfies the forensic forgery demands and don't need additional datasets to enhance the performance of the proposed system.

4.4 Results of Data Preprocessing

The preprocessing stage mentioned in Section 3.2.2 includes three steps and all dataset images go through it. These preprocessing steps include:

4.4.1 Resizing image dimensions

This step is necessary when considering computational resources, especially with large datasets such as the used dataset in this thesis. Resizing the RGB images of the dataset to 128 * 128 pixels as shown in Figure (4.1), and in the previous chapter in section 3.2.2. which is considered the appropriate size, as it is not very expensive from a computational point, and to preserve the details of the image information from decay. It is considered better than the minimum image dimensions accepted by the networks of 71 * 71 pixels as shown in the Table (4.1) for measures of the training (Trn), testing (Tst), and validation (Val) sets per 10 and 20 epochs using the MTXception net which gave better accuracy.

Table 4.1 Results of image dimensions per epochs

Images dimensions	Epochs	Trn Samples	Trn Acc	Trn Loss	Val Samples	Val Acc	Val Loss	Tst Samples	Tst Acc	Tst Loss
71*71 px	10	112000	0.9836	0.0291	7000	0.9747	0.0694	21000	0.9737	0.0764
	20	112000	0.9955	0.0122	7000	0.9791	0.0821	21000	0.9802	0.0738
128*128 px	10	112000	0.9937	0. 0183	7000	0.9871	0. 0364	21000	0.9815	0.0490
	20	112000	0.9969	0.0094	7000	0.9843	0.0580	21000	0.9815	0.0616

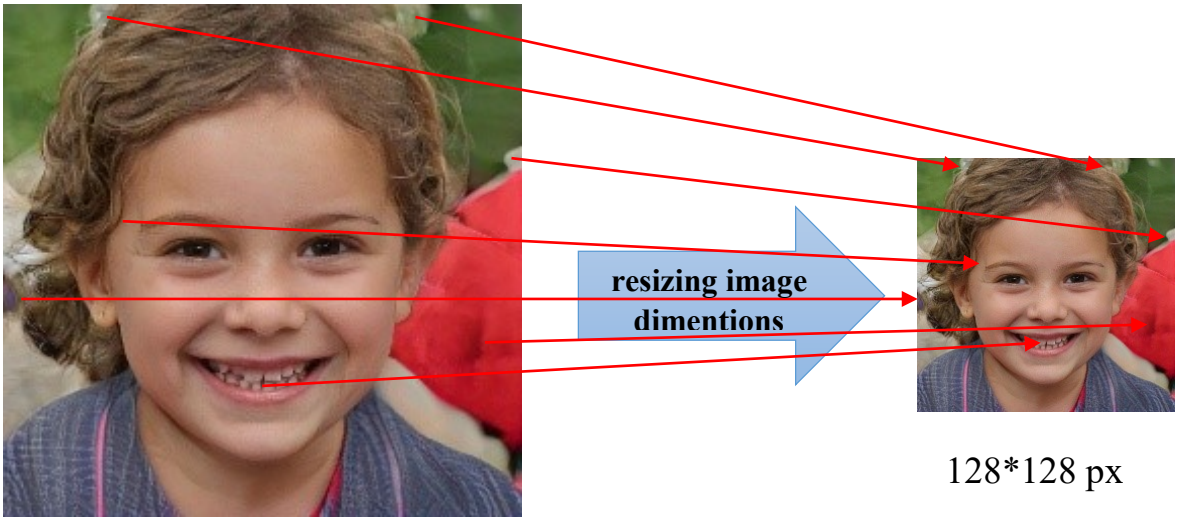


Figure 4.1 result of reducing image Dimensions step for fake image.

4.4.2 Converting Color System

Converting the image color system to the YCbCr system gave somewhat better results than the RGB color system and much than other color systems such as lab or HSV as mentioned in section 3.2.2, and shown in Table (4.2) based on splitting the dataset as mentioned in Figure 3.6 in section 3.2.3, where when looking at the nature of the images in the dataset used in the thesis, that it are very consistent. Therefore, it requires using a color system with correlated color values to each byte

in the pixel and this is available in the RGB color system as well as in the YCbCr color system, but YCbCr outperform the RGB and other color systems. When tested on MTXception net, it is possible to obtain evidence of forgery by taking advantage of the luminance part (Y Channel) can represent the gray level of images. Beside the clues in the coloring channels (Cb and Cr channels),

Table 4.2 Results of 10 epochs for tested Color Systems.

Color System	Epochs	Trn Samples	Trn Acc	Trn Loss	Val Samples	Val Acc	Val Loss	Tst Samples	Tst Acc	Tst Loss
YCbCr	10	112000	0.9937	0.0183	7000	0.9871	0.0364	21000	0.9881	0.0375
HSV	10	112000	0.9935	0.0186	7000	0.9834	0.0512	21000	0.9860	0.0437
RGB	10	112000	0.9924	0.0210	7000	0.9814	0.0532	21000	0.9819	0.0590
lab	10	112000	0.9949	0.0142	7000	0.9726	0.0830	21000	0.9757	0.0679

4.4.3 Image segmentation

In this step, the images of size 128 * 128 are divided into 8 columns and 8 rows to be 64 partitions based on 16 * 16 pixels for each partition as mentioned in Chapter Three section 3.2.2.c to be the total number of partitions 8,960,000 partitions each of these partitions hold the original image label, these partitions split to training, evaluation, and testing set to train the partitions localization net. But when implementing the overall system this step is implemented after classification the image as a fake.

When looking at the issue of the analog the face and the convergence of adverse parts of the face, such as the eyes, nose, and lips, and this may cause an imbalance in the weights of the network when passing these counteractive sections horizontally to the network with sequential selection. Therefore, using the vertical partitioning instead of the horizontal partitioning of the image for scatting these adverse parts, where the best accuracy during the first epoch of the horizontal

division was %69.36, while the vertical division achieved %71.43. On the other hand when using the shuffle selection for partitions then the horizontal splitting is better because the vertical splitting may converge these adverse parts of the face, where the horizontal partitioning achieved best accuracy %71.75 during the first epoch while the vertical partitioning achieved %69.83.

4.5 Preparing the dataset

The dataset, as mentioned in Chapter Three, there are two phases of preparing the dataset the first for image classification to real or fake where the images dataset split to 80% for training, 5% for validation, and 15% for the test set. The second phase for classification the partitions of the images, where the dataset after partitioning the images split into 85% of training 5% for validation, and 10% for testing.

4.6 Results of image classification models

Two networks are used to classify the image as fake or real by extracting the features of the image, where each of them takes the same input image every time, then calculates the prediction average based on a threshold of 0.5. These networks are as follows:

4.6.1 pretrained Xception net

This network is used to extract features for the input data. It is consists of 14 blocks as indicated previously in Chapter Three Section 3.2.4.A, The network uses ImageNet weights as initial weights, then the network trained on 80% of the used dataset, 5% for validation, and tested on 15% of the dataset.

The main parameters that control the results of the training process of this network are three parameters, the number of epochs, batch size, and learning rate. The experimental results of the proposed model on the values of these parameters are shown in the following tables during the stages of training, validation, and testing based on the "Adam" optimizer with initial learning rate of 0.0001 where it is better than 0.001 as shown in Table (4.3) . Table (4.4) shows the batch size adjustment process by comparing the accuracy and loss function values during training, validation, and testing per 10 epochs. Table (4.5) shows the difference between achieved results for the accuracy and loss function values for different epochs.

When looking at the results of Table (4.5), we find that the best accuracy of the model when training it for 17 epochs, Figure (4.2) shows curve of accuracy, and Figure (4.3) shows the Loss function curve for 17 epochs.

When implementing the final model in ".hdf5" file format of 17 epochs on the used dataset was 15% for the test phase, which is equivalent to 21000 images with shuffle selection from the images of the used dataset, the results ranged from 0.9986 to 0.9993. And when its implementation on 139999 images, which is the most test limit to find the maximum number of undefined images, the total number is 152 undefined images which gives an accuracy of 0.9989.

Table 4.3 Results pretrained Xception network of 17 epochs for different learning rate

Learning rate	Epochs	Trn Samples	Trn Acc	Trn Loss	Val Samples	Val Acc	Val Loss	Tst Samples	Tst Acc	Tst Loss
0.001	17	112000	0.9976	0.0070	7000	0.9920	0.0241	21000	0.9916	0.0286
0.0001	17	112000	0.9993	0.0019	7000	0.9954	0.0233	21000	0.9946	0.0184

Table 4.4 Results pretrained Xception network of 10 epochs for different batch size

Batch size	Trn Acc	Trn Loss	Val Acc	Val Loss	Tst Acc	Tst Loss
16	0.9976	0.0066	0.9736	0.1044	0.9710	0.1188
32	0.9978	0.0061	0.9783	0.1214	0.9740	0.1292
64	0.9980	0.0065	0.9603	0.1910	0.9588	0.1880

Table 4.5 Results pretrained Xception network of 32 batch size for diffrent epochs

Epochs	Trn Acc	Trn Loss	Val Acc	Val Loss	Tst Acc	Tst Loss
10	0.9978	0.0061	0.9783	0.1214	0.9740	0.1292
13	0.9986	0.0045	0.9887	0.0410	0.9886	0.0398
17	0.9993	0.0019	0.9954	0.0233	0.9954	0.0184
18	0.9989	0.0033	0.9894	0.0390	0.9910	0.0357
19	0.9975	0. 0080	0.9911	0 0332	0.9928	0.0267

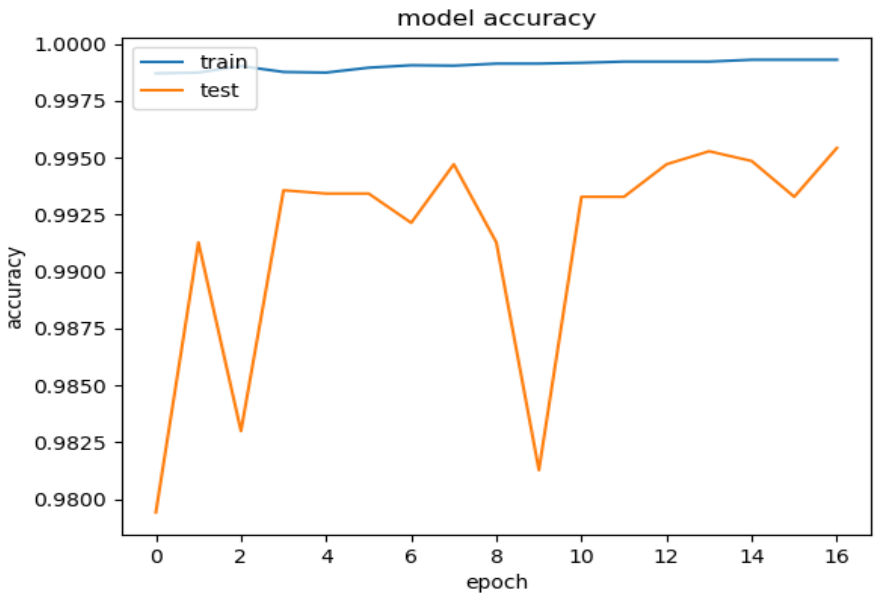


Figure 4.2 The Accuracy curve of 17 epochs of pretrained Xception net.

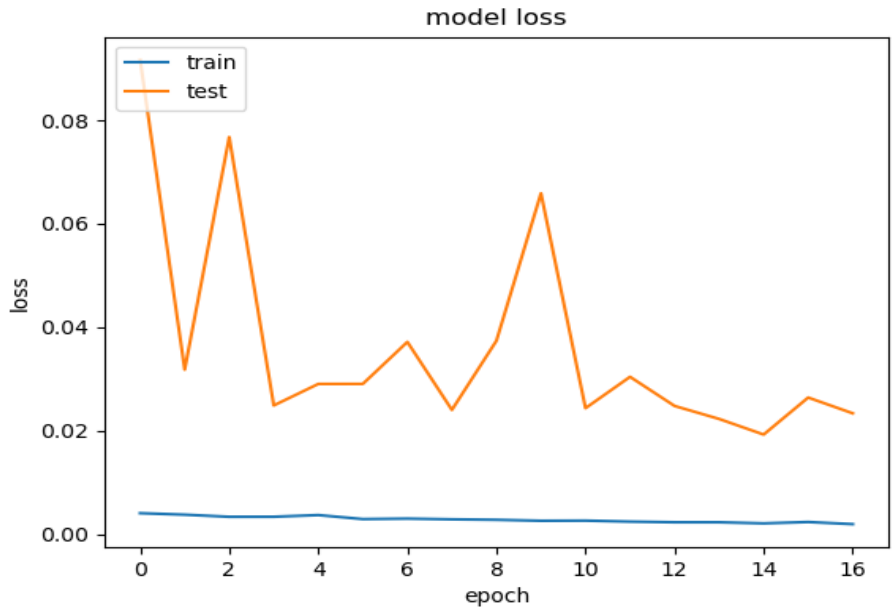


Figure 4.4 The Loss function curve of 17 epochs of pretrained Xception net

4.6.2 Non trained MTXception net

This network is used to classify the images as forgery or real besides the first network based on a modified Xception network and dense connectivity used to extract features for the input data, This model consists of 7 blocks as indicated previously in Chapter Three Section 3.2.4.2. The network trained on 80% of the used dataset, 5% for validation, and tested on 15% of the dataset.

In general, the main parameters that control the results of the training process in this network are two parameters, the number of epochs, and batchsize. The experimental results of the proposed model on the values of these parameters are shown in the following tables during the stages of training, verification, and testing based on optimizer function “Adam” with a learning rate of 0.001. Table (4.6) shows the batch size adjustment by comparing the accuracy and loss function values during training, validation, and testing phases per 10

epochs, while the table (4.7) shows the difference between achieved results for the accuracy and the loss function values for different epochs.

Table 4.6 Results MTXception network of 10 epochs for different batch size

Batch-size	Epochs	Trn Acc	Trn Loss	Val Acc	Val Loss	Tst Acc	Tst Loss
16	10	0.9929	0.0159	0.9836	0.0538	0.9816	0.0580
32	10	0.9937	0.0183	0.9871	0.0364	0.9881	0.0375
64	10	0.9932	0.0193	0.9803	0.0593	0.9814	0.0566
128	10	0.9951	0.0703	0.9740	0.0847	0.9741	0.0849

Table 4.7 Results MTXception network of 32 batch size for different epochs

Epochs	Trn Acc	Trn Loss	Val Acc	Val Loss	Tst Acc	Tst Loss
10	0.9937	0.0183	0.9871	0.0364	0.9881	0.0375
20	0.9972	0.0083	0.9873	0.0580	0.9823	0.0616
30	0.9980	0.0065	0.9872	0.0507	0.9885	0.0533
50	0.9991	0.0029	0.9930	0.0329	0.9909	0.0493
100	0.9994	0.0017	0.9949	0.0298	0.9944	0.0322
150	0.9996	0.0012	0.9949	0.0325	0.9949	0.0449
175	0.9997	0.0006	0.9968	0.0229	0.9970	0.0314
200	0.9997	0.0008	0.9946	0.0549	0.9943	0.0498

When looking at the results of Table (4.7), the best accuracy of the model when training it for 175 epochs, Figure (4.4) shows an accuracy curve, and Figure (4.5) shows the loss function curve for 175 epochs.

Then the final parameter values for training the network are as follows:

The number of epochs is 175, and batchsize of 32 images

When implementing the network in ".hdf5" file format of 175 epochs on the used dataset was 15% for the test phase, which is equivalent to 21000 images with shuffle selection from the images of the used dataset, the results ranged from 0.9990 to 0.9996. And when its implementation on 139999 images, which is the most test limit to find the maximum number of undefined images, the total number is 97 undefined images which gives an accuracy of 0.9993.

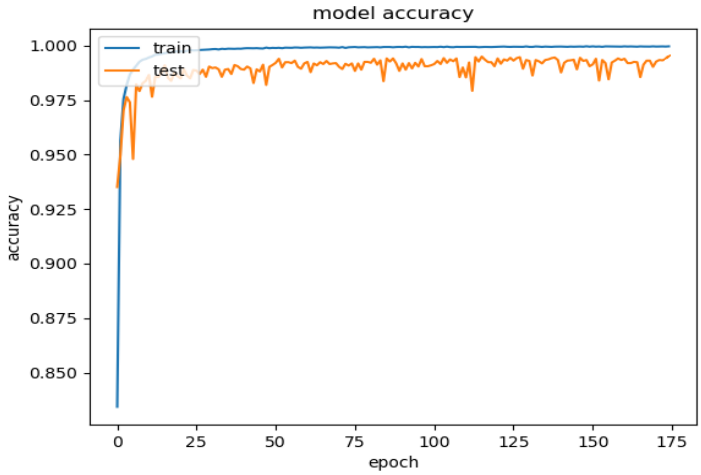


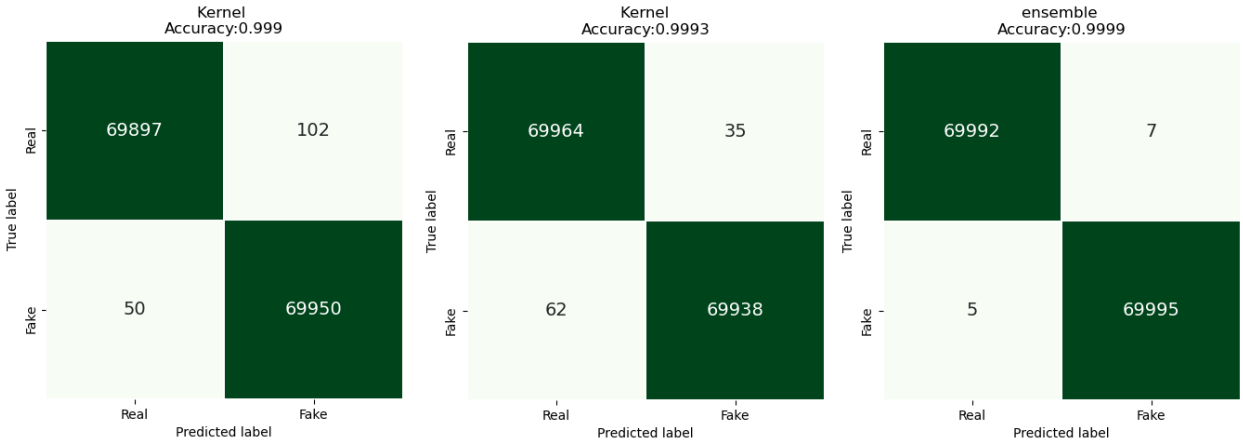
Figure 4.4 The Accuracy curve of 175 epochs of MTXception net.

Figure 4.5 The Loss function curve of 175 epochs of MTXception net.

4.6.3 Ensemble method

Ensemble method is used to overcome the weak point of the two classifiers based on threshold 0.5. The Ensemble method obtained a very high accuracy nearest to

optimal, where the maximum number of images that are not recognized is 12 images from all images in the dataset as shown in Figure (4.6) which shows the confusion matrix for the results of each network and ensemble them.



a) Pretrained Xception Net (b) MTXception Net (c) Ensemble method

Figure 4.6 confusion matrix of compared models for 139999 sample.

4.7 Results and Experiences of SLNetwork

Partitions Localization Network is used to extract the partitions features of the images as described earlier in Chapter Three, Section 3.2.4.C. This network was trained on image partitions with dimensions of 16*16. This network adopts Adam optimizer with a learning rate of 0.002.

There are two main parameters that control the training process for this network, which are the batchsize, and the number of epochs. The following tables show the experimental results of the network for adjusting the values of these two parameters for the training, validation, and testing phases. Table (4.8) shows the batchsize adjustment by comparing the accuracy and loss function values during training, validation, and testing for different batchsizes for 10 epochs, and Table (4.9) shows the experimental results of the network for adjusting the epochs number during the training, validation, and testing phases for static batchsize of 256. Figure (4.7) shows

curve of accuracy for the network, and Figure (4.8) shows curve of loss function for the network.

Table 4.8 Results of SLNetwork of 10 epochs for different batch size.

Batch-size	Trn Samples	Trn Acc	Trn Loss	Val Samples	Val Acc	Val Loss	Tst Samples	Tst Acc	Tst Loss
64	7,616,000	0.8303	0.3680	448,000	0.8153	0.4076	896,000	0.8164	0.4065
128	7,616,000	0.8136	0.3976	448,000	0.8102	0.4061	896,000	0.8100	0.4058
256	7,616,000	0.8317	0.3663	448,000	0.8154	0.4001	896,000	0.8167	0.3960
512	7,616,000	0.8266	0.3751	448,000	0.8050	0.4129	896,000	0.8044	0.4134

Table 4.9 Results of SLNetwork of 256 batch size for different epochs.

Epochs	Trn Samples	Trn Acc	Trn Loss	Val Samples	Val Acc	Val Loss	Tst Samples	Tst Acc	Tst Loss
30	7,616,000	0.8553	0.3196	448,000	0.8310	0.3957	896,000	0.8316	0.3904
50	7,616,000	0.8560	0.3331	448,000	0.8351	0.3739	896,000	0.8275	0.4025
100	7,616,000	0.8945	0.2420	448,000	0.8564	0.2940	896,000	0.8582	0.2919
300	7,616,000	0.9153	0.2007	448,000	0.8957	0.2366	896,000	0.8959	0.2334

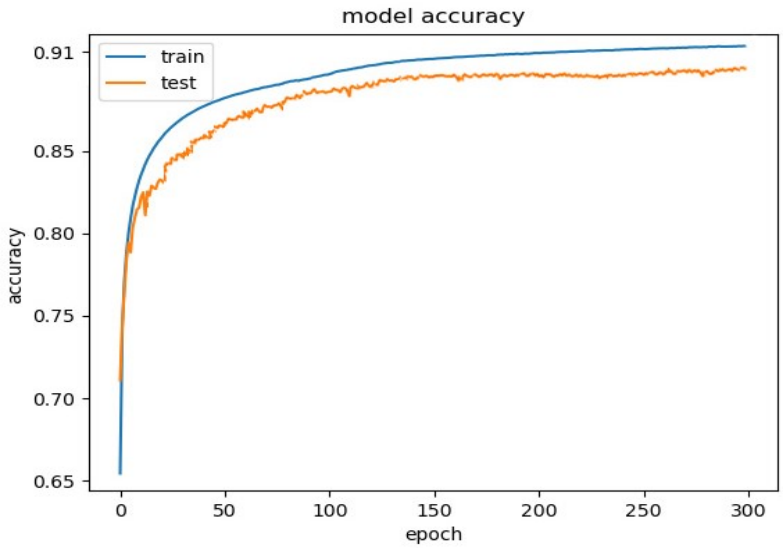


Figure 4.7: the accuracy curve SLNetwork.

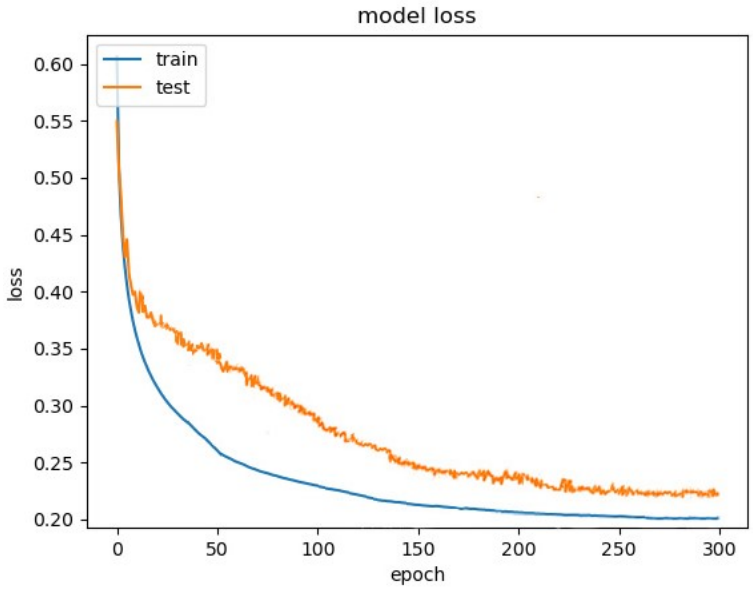


Figure 4.8: the loss function curves of SLNetwork for 300 epoch.

When implementing the model in ".hdf5" file format of 300 epochs on 8959999 partitions, the most test limit to find the maximum number of undefined partitions, the total number is 705671 undefined partitions which gives an accuracy of %92.124 as shown in Figure (4.9). and when tracing 1000 images tested on the model the undetected partitions of the 64 partitions are around between 0 and 17 partitions in the worst case, and most of the undetected partitions if not all of them appear in the background or in the hair partitions not in the actual face partitions in the image which is the main forgery objective in the dataset.

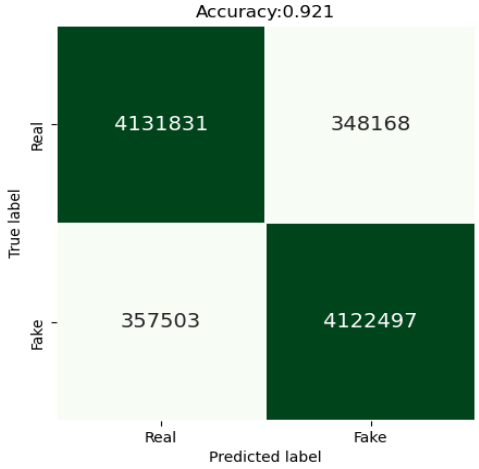


Figure 4.9 confusion matrix of SLNetwork for 8959999 samples.

4.8 postprocessing step

The post-processing stage represents the result of the two processes, The first of classifying the image and the second of classifying the partitions of the image, where the partitions of the images classified as fake are passed to SLNetwork, the detected partitions as fake are framed in red color and the undetected partitions in green color as shown in Figure (4.10).

Figure 4.10 postprocessing step for fake images localization.

4.9 System Evaluation

As explained in Chapter Three, the overall system goes through two phases the initial phase is the image classification process using the ensemble method for the predictions of pretrained Xception and MTXception nets, where the result is based on threshold value and condition to make a decision for the image is fake, or real. Therefore the threshold value and the condition form control the classification accuracy. The threshold based on the average of two classes which is equal to 0.5 controls the accuracy, but the condition is chosen according to the principle of the experiments in practical tests by comparing the classification accuracy of another classifier of MTXception which is undefined for 136 images based on testing all

dataset of 140000 images. Table (4.10) shows the process of selecting the condition form.

Table 4.10: The process of selection condition form.

Condition form	TP	FP	TN	FN	Accuracy	Pricision	Recall	F1_score
x >= 0.5	69994	6	69990	10	0.99991	0.99992	0.9998	0.99986
x > 0.5	69994	6	69986	14	0.99982	0.99991	0.99980	0.99985

Therefore according to the results that are elucidated in Table (4.9), the condition is if the ensemble result equal to or more than 0.5 then the image is fake otherwise the image is real.

The second phase is the localization of fake images partitions using SLNetwork. Therefore evaluating the overall forgery detection system performance requires evaluating the performance of each network in the system phases based on the testing dataset. The measures of Confusion matrix, accuracy, precision, recall, and F1-Measure are used in the evaluation. The best results of testing the performance of the system networks are shown in the confusion matrix in Table (4.11), and the performance metrics shown in Table (4.12), and When evaluating the time consuming of testing the proposed system based on the specified hardware and software in section 4.2, and the testing the predicting time of the images of the testing set 21000 images, and for 300 images which represent 10 seconds based on 30 frames per second for video, it can be described in Table (4.13), the Table (4.14) explain comparing the proposed system with other methods.

Table 4.11: The confusion matrix of best results of testing the proposed system networks.

network	Number of samples	Actual labels	Predicted labels	
			Real	Fake
Pretrained Xception	21000	**Real**	10409	9
		Fake	8	10574
MTXception	21000	**Real**	10555	4
		Fake	5	10436
Ensemble method	21000	**Real**	10503	0
		Fake	0	10497
SLNetwork	1344000	**Real**	619983	52043
		Fake	53068	618906

Table 4.12: The best Performance Metrics of proposed system networks.

network	Number of samples	Accuracy	precision	recall	F1-measure
Pretrained Xception	21000	0.9992	0.9991	0.9992	0.9991
MTXception	21000	0.9996	0.9996	0.9995	0.9996
Ensemble method	21000	1	1	1	1
SLNetwork	1344000	0.9218	0.9226	0.9216	0.9219

Table 4.13: The time consuming for testing the model.

network	Number of samples	Time (seconds)	Number of samples	Time (seconds)
Pretrained Xception	21000	24.46	300	0.37
MTXception	21000	14.29	300	0.24
Ensemble method	21000	38.7	300	0.6
SLNetwork	1344000	80.8	19200	1.17

4.10 Generalization

Another Evaluation side of generalization the system is when testing the system networks on handcrafted images based on the splicing forgery technique of two real images as shown in Figure (4.11) of some real samples of these images. Besides personal real images which predicted correctly.

Table 4.14: compare between proposed system with other method.

Authors	approach	Dataset	Accuracy
H. S. Shad et al [11]	deep learning for separate CNN architectures	140k Real and Fake Faces	VGG19 %94 VGG16 %92 VGGFace %99 DenseNet169 %95 DenseNet201 %96 DenseNet121 %97 ResNet50 %97 Custom CNN %90
N.-K. Ngo and X.-N. Cao [12]	Deep learning for U-net classifier	140k Real and Fake Faces	%98.43
J. Sharma and S. Sharma[13]	Deap learning for custom CNN	140k Real and Fake Faces	%94.41

Zhengzhe Liu and Xiaojuan Qi. [14]	Deep learning for GramNet (Gram block + ResNet)	StyleGAN, PGGAN, DRAGAN, DCGAN, StarGAN, CelebA-HQ, FFHQ, CelebA	%98.96
Chih-Chung Hsu et.al [15]	Siamese networks Deep learning the contrastive loss Features	DCGAN, WGAP, WGAN-GP(WGAN with Gradient-Penalty), LSGAN, PGGAN • CelebaA	Precision= %98.8, Recall= %94.8
Proposed system	Ensemble deeap learning of Xception networks	140k Real and Fake Faces	%99.99

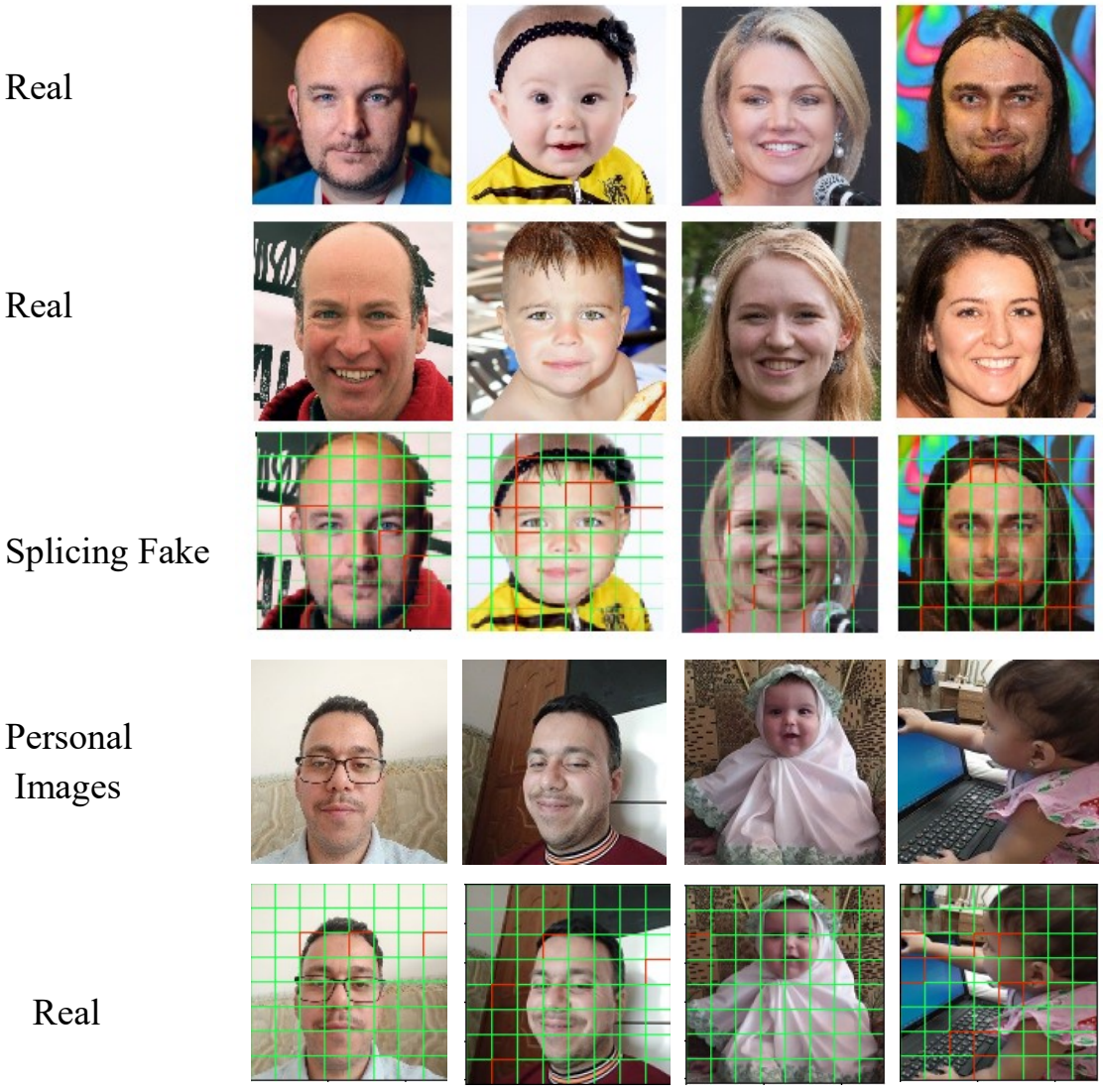


Figure 4.11 samples of detected images of forgery splicing technique and unseen personal real images.

4.10 Enhance Learning:

After performing the Ensemble method and getting a high accuracy of %99.99, apeared the idea of enhance learning performance and getting more Accuracy by reducing the cost of training from scratch in resources, time, and effort. Therefore, the learning can start from the end of the last learned model and weights saved in the model based on a smaller learning rate of 0.1 or less than the priorly specified learning rate of the network to reduce the changing on the saved weights, where when applying the enhancing on pretrained Xception net and MTXception for three times gave an accuracy of %100, as shown in figure (4.12) but the loss function value of pretrained Xception net is 1.2e-5 and 5.6e-7 for MTXception net.

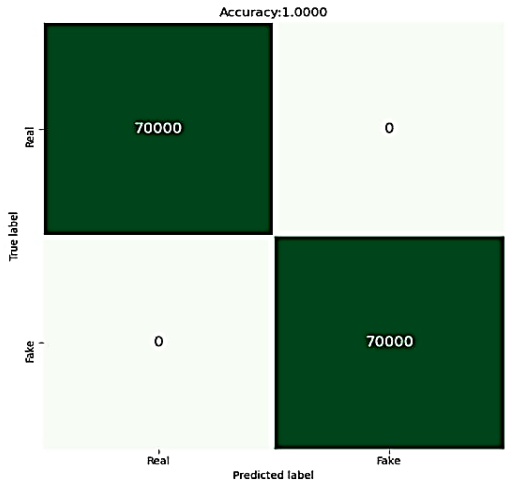


Figure 4.12 Results of enhance learning of MTXception net

Chapter Five

Conclusion and Future Works

5.1 Research Conclusions

Several conclusions during the design and implementation of the proposed system have been drawn. Following the results of this thesis:

1- Convolutional neural networks give better performance than the statistical methods and handcraft features in detecting image forging, especially with multi manipulation forgery, such as the image results from recent generations of GAN networks, for example, the StyleGan network used in this study.

2- When referring to the nature of the dataset images used, it includes not only the forgery for the faces but also the background where all parts of GAN generated image is fake but have the features of real images, so the entire image is processed, where the evidence of forgery may be seen in the image's background instead of only in the face part as presented in figures 2.26, 3.17, and 4.11.

3- When converting the image color system to YCbCr, the proposed model performed better than other color systems., as it deals with grayscale and two channels for color values, where the abnormality in the forgery regions appear more distinguished in this color system more than the channels for other color systems as presented in table 4.3 in section 4.4.2.

4- Depthwise sep arable convolution two dimensions are considered better than standard convolution in the accuracy of results, and the speed of execution, even though it goes through two convolutional steps with two filters, not such as standard convolution that goes through one step with one filter.

5- The idea of residual data affects the results of the model, wherein in the best cases of without adding these data, the accuracy decreased by about 10% of the results

achieved with these data, where the accuracy of the test phase with 175 epochs was above 89%.

6- After each convolutional layer, apply the BatchNormalization layer to balance the feature map values of the inputs, improving the representation of image features and boosting the learning operation.

7- Connecting the last block of the modified Xception network with the GlobalAveragePooling, GlobalMaxPooling, and Flatten layers using a dense approach led to increasing the feature maps in a reasonable ratio in order to obtain higher accuracy.

8- When using convolutional architectures with transfer learning, then should be set a small value for the initial learning rate to make small changes in the initial weights of the pretrained network as shown in the Table 4.3.

5.2 Future Works

1. Improving the training time of the network in addition to using an efficient function to control the learning rate gradually to obtain high accuracy with few epochs, as the lower the learning rate leads to an increasing in the efficiency of the model.
2. Applying the proposed model on another dataset created by other Gan network architectures.
3. Work to create a new architecture for the network that does not depend on the transfer learning and with higher performance in terms of speed in implementation and accuracy in results.
4. An attempt to process the images using a deep learning based on one-dimensional convolution, by taking the features of images to process them to reduce the time of training of the neural network with considering to keep the high accuracy.

References

References

[1] T. Qazi *et al.*, “Survey on blind image forgery detection,” *IET Image Process.*, vol. 7, no. 7, pp. 660–670, 2013, doi: 10.1049/iet-ipr.2012.0388.

[2] H. Farid, “Digital doctoring: How to tell the real from the fake,” *Significance*, vol. 3, no. 4, pp. 162–166, 2006, doi: 10.1111/j.1740-9713.2006.00197.x.

[3] S. Walia and K. Kumar, “Digital image forgery detection: a systematic scrutiny,” *Aust. J. Forensic Sci.*, vol. 51, no. 5, pp. 488–526, 2019, doi: 10.1080/00450618.2018.1424241.

[4] A. B. Z. Abidin, H. B. A. Majid, A. B. A. Samah, and H. B. Hashim, “Copy-move image forgery detection using deep learning methods: A review,” *Int. Conf. Res. Innov. Inf. Syst. ICRIIS*, vol. December-2, 2019, doi: 10.1109/ICRIIS48246.2019.9073569.

[5] B. G. Weinstein, “A computer vision for animal ecology,” *J. Anim. Ecol.*, vol. 87, no. 3, pp. 533–545, 2018, doi: 10.1111/1365-2656.12780.

[6] J. Clemons, C. C. Cheng, I. Frosio, D. Johnson, and S. W. Keckler, “A patch memory system for image processing and computer vision,” *Proc. Annu. Int. Symp. Microarchitecture, MICRO*, vol. 2016-Decem, 2016, doi: 10.1109/MICRO.2016.7783754.

[7] O. H. Babatunde, L. Armstrong, J. Leng, and D. Diepeveen, “A survey of computer-based vision systems for automatic identification of plant species,” *J. Agric. Informatics*, vol. 6, no. 1, pp. 61–71, 2015, doi: 10.17700/jai.2015.6.1.152.

[8] J. Zhuang, J. Yang, L. Gu, and N. Dvornek, “Shelfnet for fast semantic segmentation,” *Proc. - 2019 Int. Conf. Comput. Vis. Work. ICCVW 2019*, pp. 847–856, 2019, doi: 10.1109/ICCVW.2019.00113.

[9] M. I. Razzak, S. Naz, and A. Zaib, “Deep learning for medical image processing: Overview, challenges and the future,” *Lect. Notes Comput. Vis. Biomech.*, vol. 26, pp. 323–350, 2018, doi: 10.1007/978-3-319-65981-7_12.

[10] M. Hatt, C. Parmar, J. Qi, and I. El Naqa, “Machine (Deep) Learning Methods for Image Processing and Radiomics,” *IEEE Trans. Radiat. Plasma Med. Sci.*, vol. 3, no. 2, pp. 104–108, 2019, doi: 10.1109/trpms.2019.2899538.

[11] H. S. Shad *et al.*, "Comparative Analysis of Deepfake Image Detection Method Using Convolutional Neural Network," *Comput. Intell. Neurosci.*, vol. 2021, 2021, doi: 10.1155/2021/3111676.

[12] N.-K. Ngo and X.-N. Cao, "Pixel-Wise Information in Fake Image Detection," 2021, pp. 436–443.

[13] J. Sharma and S. Sharma, "Challenges and Solutions in DeepFakes," pp. 1–17, 2021, [Online]. Available: http://arxiv.org/abs/2109.05397.

[14] Z. Liu, X. Qi, and P. H. S. Torr, "Global Texture Enhancement for Fake Face Detection in the Wild," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 8057–8066, 2020, doi: 10.1109/CVPR42600.2020.00808.

[15] C. C. Hsu, Y. X. Zhuang, and C. Y. Lee, "Deep fake image detection based on pairwise learning," *Appl. Sci.*, vol. 10, no. 1, 2020, doi: 10.3390/app10010370.

[16] L. M. Dang, K. Min, S. Lee, D. Han, and H. Moon, "Tampered and computer-generated face images identification based on deep learning," *Appl. Sci.*, vol. 10, no. 2, 2020, doi: 10.3390/app10020505.

[17] X. Chang, J. Wu, T. Yang, and G. Feng, "DeepFake Face Image Detection based on Improved VGG Convolutional Neural Network," *Chinese Control Conf. CCC*, vol. 2020-July, pp. 7252–7256, 2020, doi: 10.23919/CCC50068.2020.9189596.

[18] Z. Mi, X. Jiang, T. Sun, and K. Xu, "GAN-Generated Image Detection with Self-Attention Mechanism against GAN Generator Defect," *IEEE J. Sel. Top. Signal Process.*, vol. 14, no. 5, pp. 969–981, 2020, doi: 10.1109/JSTSP.2020.2994523.

[19] L. Nataraj *et al.*, "Detecting GAN generated Fake Images using Co-occurrence Matrices," *IS T Int. Symp. Electron. Imaging Sci. Technol.*, vol. 2019, no. 5, pp. 1–7, 2019, doi: 10.2352/ISSN.2470-1173.2019.5.MWSF-532.

[20] P. He, H. Li, and H. Wang, "Detection of Fake Images Via the Ensemble of Deep Representations from Multi Color Spaces," *Proc. - Int. Conf. Image Process. ICIP*, vol. 2019-Septe, pp. 2299–2303, 2019, doi: 10.1109/ICIP.2019.8803740.

[21] K. Obaid, S. Zeebaree, and O. Ahmed, "Deep Learning Models Based on Image Classification: A Review," *Int. J. Soc. Sci. Bus.*, vol. 4, pp. 75–81,

2020, doi: 10.5281/zenodo.4108433.

[22] M. Hassaballah and A. I. Awad, *Deep Learning in Computer Vision*, no. March. 2020.

[23] S. K. Mankar and A. A. Gurjar, "Image Forgery Types and Their Detection: A Review," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 5, no. 4, pp. 174–178, 2015.

[24] C. N. Bharti, "A Survey of Image Forgery Detection Techniques," pp. 877–881, 2016.

[25] M. F. Hashmi, V. Anand, and A. G. Keskar, "Copy-move Image Forgery Detection Using an Efficient and Robust Method Combining Un-decimated Wavelet Transform and Scale Invariant Feature Transform," *AASRI Procedia*, vol. 9, no. Csp, pp. 84–91, 2014, doi: 10.1016/j.aasri.2014.09.015.

[26] A. Kaur, "Copy-Move Forgery Detection using DCT and SIFT," vol. 70, no. 7, pp. 30–34, 2013.

[27] M. Ali Qureshi and M. Deriche, "A review on copy move image forgery detection techniques," *2014 IEEE 11th Int. Multi-Conference Syst. Signals Devices, SSD 2014*, no. August 2015, 2014, doi: 10.1109/SSD.2014.6808907.

[28] M. C. Stamm and K. J. R. Liu, "Anti-forensics of digital image compression," *IEEE Trans. Inf. Forensics Secur.*, vol. 6, no. 3 PART 2, pp. 1050–1065, 2011, doi: 10.1109/TIFS.2011.2119314.

[29] P. R. Rothe, P. P. Asthankar, and J. P. Rothe, "Image Forgery Detection Based on SURF and Machine Learning Classifier," vol. 14, no. 1, pp. 38–43, 2019, doi: 10.9790/2834-1401023843.

[30] A. Gupta, N. Saxena, and S. K. Vasistha, "Detecting Copy move Forgery using DCT," *Int. J. Sci. Res. Publ.*, vol. 3, no. 5, pp. 3–6, 2013.

[31] M. D. Ansari, S. P. Ghrera, V. Tyagi, M. D. Ansari, S. P. Ghrera, and V. Tyagi, "Pixel-Based Image Forgery Detection : A Review Pixel-Based Image Forgery Detection : A Review," *IETE J. Educ.*, vol. 55, no. 1, pp. 40–46, 2014, doi: 10.1080/09747338.2014.921415.

[32] O. Amer and J. M. Al-tuwaijari, "Analysis of challenges and methods for face detection systems : A survey," vol. 13, no. November 2021, pp. 3997–4015, 2022.

[33] Bojan, “140k Real and Fake Faces | Kaggle.” https://www.kaggle.com/xhlulu/140k-real-and-fake-faces.

[34] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, “Deepfakes and beyond: A Survey of face manipulation and fake detection,” *Inf. Fusion*, vol. 64, pp. 131–148, 2020, doi: 10.1016/j.inffus.2020.06.014.

[35] A. K. Jain, B. Klare, and U. Park, “Face matching and retrieval in forensics applications,” *IEEE Multimed.*, vol. 19, no. 1, pp. 20–27, 2012, doi: 10.1109/MMUL.2012.4.

[36] A. V. Malviya and S. A. Ladhake, “Copy move forgery detection using low complexity feature extraction,” *2015 IEEE UP Sect. Conf. Electr. Comput. Electron. UPCON 2015*, 2016, doi: 10.1109/UPCON.2015.7456688.

[37] N. Yusoff and L. Alamro, “Implementation of feature extraction algorithms for image tampering detection,” *Int. J. Adv. Comput. Res.*, vol. 9, no. 43, pp. 197–211, 2019, doi: 10.19101/ijacr.pid37.

[38] M. Jogin, Mohana, M. S. Madhulika, G. D. Divya, R. K. Meghana, and S. Apoorva, “Feature extraction using convolution neural networks (CNN) and deep learning,” *2018 3rd IEEE Int. Conf. Recent Trends Electron. Inf. Commun. Technol. RTEICT 2018 - Proc.*, no. May 2018, pp. 2319–2323, 2018, doi: 10.1109/RTEICT42901.2018.9012507.

[39] B. Mehlig, “Machine learning with neural networks,” 2021.

[40] C. Aggarwal, *Neural Networks and Deep Learning*. Springer, 2018.

[41] S. W. Smith, “Neural Networks,” *Digit. Signal Process.*, pp. 451–480, 2003, doi: 10.1016/b978-0-7506-7444-7/50063-7.

[42] R. Dastanian, E. Abiri, M. R. Salehi, and A. Akbari, “A new approach based on TLBO for DC-DC converter in RFID tag,” *J. Intell. Fuzzy Syst.*, vol. 29, no. 5, pp. 1827–1833, 2015, doi: 10.3233/IFS-151660.

[43] R. W. Pettit, R. Fullem, C. Cheng, and C. I. Amos, “Artificial intelligence, machine learning, and deep learning for clinical outcome prediction,” *Emerg. Top. Life Sci.*, vol. 5, no. 6, pp. 729–745, 2021, doi: 10.1042/ETLS20210246.

[44] S. Sharma, S. Sharma, and A. Anidhya, “Understanding Activation Functions in Neural Networks,” *Int. J. Eng. Appl. Sci. Technol.*, vol. 4, no. 12, pp. 310–316, 2020.

[45] J. Feng and S. Lu, “Performance Analysis of Various Activation Functions in Artificial Neural Networks,” *J. Phys. Conf. Ser.*, vol. 1237, no. 2, 2019, doi: 10.1088/1742-6596/1237/2/022030.

[46] R. H. R. Hahnioser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, “Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit,” *Nature*, vol. 405, no. 6789, pp. 947–951, 2000, doi: 10.1038/35016072.

[47] A. F. M. Agarap, “Deep Learning using Rectified Linear Units (ReLU),” *arXiv*, no. 1, pp. 2–8, 2018.

[48] S.-H. Han, K. W. Kim, S. Kim, and Y. C. Youn, “Artificial Neural Network: Understanding the Basic Concepts without Mathematics,” *Dement. Neurocognitive Disord.*, vol. 17, no. 3, p. 83, 2018, doi: 10.12779/dnd.2018.17.3.83.

[49] S. I. Granshaw, *Neural networks and neurodiversity*, vol. 36, no. 175. 2021.

[50] S. Theodoridis, “Neural Networks and Deep Learning,” in *Machine Learning*, 2015.

[51] K. Janocha and W. M. Czarnecki, “On loss functions for deep neural networks in classification,” *Schedae Informaticae*, vol. 25, pp. 49–59, 2016, doi: 10.4467/20838476SI.16.004.6185.

[52] S. P. Siregar and A. Wanto, “Analysis of Artificial Neural Network Accuracy Using Backpropagation Algorithm In Predicting Process (Forecasting),” *IJISTECH (International J. Inf. Syst. Technol.*, vol. 1, no. 1, p. 34, 2017, doi: 10.30645/ijistech.v1i1.4.

[53] C. Fougstedt, M. Mazur, L. Svensson, H. Eliasson, M. Karlsson, and P. Larsson-Edefors, “Time-domain digital back propagation: Algorithm and finite-precision implementation aspects,” *Opt. InfoBase Conf. Pap.*, vol. Part F40-O, 2017, doi: 10.1364/OFC.2017.W1G.4.

[54] D. Rengasamy, M. Jafari, B. Rothwell, X. Chen, and G. P. Figueredo, “Deep learning with dynamically weighted loss function for sensor-based prognostics and health management,” *Sensors (Switzerland)*, vol. 20, no. 3, 2020, doi: 10.3390/s20030723.

[55] K. L. Du and M. N. S. Swamy, *Neural networks and statistical learning*, vol. 9781447155, no. October 2015. 2014.

[56] S. Ruder, “An overview of gradient descent optimization algorithms,” pp. 1–14, 2016, [Online]. Available: http://arxiv.org/abs/1609.04747.

[57] K. A. Powers, L. Candela, T. Anderson, and A. Einstein, “Chapter 2 of o nline l earning,” *Am. J. Crit. Care*, vol. 25, no. 4, pp. 803–832, 1992.

[58] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–15, 2015.

[59] S. Gollapudi, “Deep Learning for Computer Vision,” *Learn Comput. Vis. Using OpenCV*, pp. 51–69, 2019, doi: 10.1007/978-1-4842-4261-2_3.

[60] Dr. Juliansyah Noor, *Encyclopedia of Computational Biology Bioinformatics and*, vol. 53, no. 9. 2019.

[61] A. Yaguchi, T. Suzuki, S. Nitta, Y. Sakata, and A. Tanizawa, “S Calable D Eep N Eural N Etworks,” pp. 1–14, 2019.

[62] B. Hitaj, G. Ateniese, and F. Perez-Cruz, “Deep Models under the GAN: Information leakage from collaborative deep learning,” 2017, doi: 10.1145/3133956.3134012.

[63] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “NNs Archtectures review,” *Elsevier*, pp. 1–31, 2017.

[64] P. Wei, C. Liu, M. Liu, Y. Gao, and H. Liu, “CNN-based reference comparison method for classifying bare PCB defects,” vol. 2018, no. Acait, pp. 1528–1533, 2018, doi: 10.1049/joe.2018.8271.

[65] Dr. René de J. Romero Troncoso Titulo, “Electrónica Digital y Lógica Programable,” p. 2021, 2007.

[66] L. Fiack and N. Cuperlier, “Embedded and real-time architecture for bio-inspired vision-based robot navigation,” 2014, doi: 10.1007/s11554-013-0391-9.

[67] K. O’Shea and R. Nash, “An Introduction to Convolutional Neural Networks,” pp. 1–11, 2015.

[68] A. Patil and M. Rane, “Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition,” *Smart Innov. Syst. Technol.*, vol. 195, pp. 21–30, 2021, doi: 10.1007/978-981-15-7078-0_3.

[69] N. Milosevic, “Introduction to Convolutional Neural Networks,” *Introd. to*

Convolutional Neural Networks, no. April, 2020, doi: 10.1007/978-1-4842-5648-0.

[70] "Convolutional Neural Network." https://scientistcafe.com/ids/convolutional-neural-network.html.

[71] C. Google, "Xception : Deep Learning with Depthwise Separable Convolutions," pp. 1251–1258, 2014.

[72] G. Brain, A. N. Gomez, and G. Brain, "Depthwise Separable Convolutions for Neural Machine Translation," 2017.

[73] J. Yuan, W. Zhou, S. Lv, and Y. Chen, "Traffic scene depth analysis based on depthwise separable convolutional neural network," *J. Electr. Comput. Eng.*, vol. 2019, 2019, doi: 10.1155/2019/9340129.

[74] V. V. Doan, D. H. Nguyen, Q. L. Tran, D. Van Nguyen, and T. H. Le, "Real-time image semantic segmentation networks with residual depth-wise separable blocks," *Proc. - 2018 Jt. 10th Int. Conf. Soft Comput. Intell. Syst. 19th Int. Symp. Adv. Intell. Syst. SCIS-ISIS 2018*, pp. 174–179, 2018, doi: 10.1109/SCIS-ISIS.2018.00037.

[75] E. Alpaydin, "Neural Networks and Deep Learning," *Mach. Learn.*, 2021, doi: 10.7551/mitpress/13811.003.0007.

[76] D. Miao, W. Pedrycz, D. Ślezak, G. Peters, Q. Hu, and R. Wang, "Rough Sets and Knowledge Technology: 9th International Conference, RSKT 2014 Shanghai, China, October 24–26, 2014 Proceedings," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8818, no. October 2014, 2014, doi: 10.1007/978-3-319-11740-9.

[77] M. Jena, S. P. Mishra, and D. Mishra, "Empirical analysis of activation functions and pooling layers in CNN for classification of diabetic retinopathy," *Proc. - 2019 Int. Conf. Appl. Mach. Learn. ICAML 2019*, pp. 34–39, 2019, doi: 10.1109/ICAML48257.2019.00014.

[78] "what-are-max-pooling-average-pooling-global-max-pooling-and-global-average-pooling." https://github.com/christianversloot/machine-learning-articles/blob/main/what-are-max-pooling-average-pooling-global-max-pooling-and-global-average-pooling.md.

[79] B. Mele and G. Altarelli, "Lepton spectra as a measure of b quark polarization at LEP," *Phys. Lett. B*, vol. 299, no. 3–4, pp. 345–350, 1993, doi: 10.1016/0370-2693(93)90272-J.

[80] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, “How does batch normalization help optimization?,” 2018.

[81] A. Ucar, “Deep Convolutional Neural Networks for facial expression recognition,” *Proc. - 2017 IEEE Int. Conf. Innov. Intell. Syst. Appl. INISTA 2017*, no. April, pp. 371–375, 2017, doi: 10.1109/INISTA.2017.8001188.

[82] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” *Artif. Intell. Rev.*, vol. 53, no. 8, pp. 5455–5516, 2020, doi: 10.1007/s10462-020-09825-6.

[83] Y. Zhu and S. Newsam, “DenseNet for dense flow,” *Proc. - Int. Conf. Image Process. ICIP*, vol. 2017-Septe, pp. 790–794, 2018, doi: 10.1109/ICIP.2017.8296389.

[84] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, “DenseNet: Implementing Efficient ConvNet Descriptor Pyramids,” pp. 1–11, 2014.

[85] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2261–2269, 2017, doi: 10.1109/CVPR.2017.243.

[86] I. J. Goodfellow, J. Pouget-abadie, M. Mirza, B. Xu, and D. Warde-farley, “Generative Adversarial Nets,” pp. 1–9, 2014.

[87] X. Yi, E. Walia, and P. Babyn, “Generative adversarial network in medical imaging: A review,” *Med. Image Anal.*, vol. 58, 2019, doi: 10.1016/j.media.2019.101552.

[88] J. Feng *et al.*, “Generative adversarial networks based on collaborative learning and attention mechanism for hyperspectral image classification,” *Remote Sens.*, vol. 12, no. 7, 2020, doi: 10.3390/rs12071149.

[89] S. Belongie, C. Science, and C. Tech, “Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization,” 2017.

[90] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 4396–4405, 2019, doi: 10.1109/CVPR.2019.00453.

[91] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila,

"Analyzing and improving the image quality of stylegan," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 8107–8116, 2020, doi: 10.1109/CVPR42600.2020.00813.

[92] S. Young, T. Abdou, and A. Bener, "Deep super learner: A deep ensemble for classification problems," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10832 LNAI, pp. 84–95, 2018, doi: 10.1007/978-3-319-89656-4_7.

[93] Rismiyati, S. N. Endah, Khadijah, and I. N. Shiddiq, "Xception Architecture Transfer Learning for Garbage Classification," *ICICoS 2020 - Proceeding 4th Int. Conf. Informatics Comput. Sci.*, 2020, doi: 10.1109/ICICoS51170.2020.9299017.

[94] N. M. Zaitoun and M. J. Aqel, "Survey on Image Segmentation Techniques," *Procedia Comput. Sci.*, vol. 65, no. Iccmit, pp. 797–806, 2015, doi: 10.1016/j.procs.2015.09.027.

[95] B. Soni, P. K. Das, and D. M. Thounaojam, "CMFD: A detailed review of block based and key feature based techniques in image copymove forgery detection," vol. 12, no. 2, pp. 167–178, 2018, doi: 10.1049/iet-ipr.2017.0441.

[96] Barvinchenko V. I. "RGB color model" Vinnitsa national technical university pp. 2–4,2017.

[97] N. a Ibraheem, M. M. Hasan, R. Z. Khan, and P. K. Mishra, "Understanding Color Models : A Review," *ARPN J. Sci. Technol.*, vol. 2, no. 3, pp. 265–275, 2012.

[98] D. Cardani, "Adventures in hsv space," *... Robótica, Inst. Tecnológico Autónomo ...*, pp. 1–10, 2001.

[99] C. Chen and M. Ren, "The significance of license plate location based on Lab color space," *Proc. 2nd Int. Conf. Information, Electron. Comput.*, vol. 59, no. Icieac, pp. 78–81, 2014, doi: 10.2991/icieac-14.2014.18.

[100] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, 2009, doi: 10.1016/j.ipm.2009.03.002.

[101] B. Säfken and A. Silbersdorff, *Learning deep*. 2020.

[102] A. Rangnekar, N. Mokashi, E. J. Ientilucci, C. Kanan, and M. J. Hoffman, "AeroRIT: A New Scene for Hyperspectral Image Analysis," *IEEE Trans.*

Geosci. Remote Sens., vol. 58, no. 11, pp. 8116–8124, 2020, doi: 10.1109/TGRS.2020.2987199.

Appendix A

The Published Paper

Republic of Iraq
Ministry of Higher Education
And Scientific Research
University of Kerbala
Dean Office

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة كربلاء كلية العلوم
مكتب العميد

No : 73 / 5 / م
date : 2021 / 6 / 28

العدد : 73 / 5 / م
التاريخ : 2021 / 6 / 28

Subject: Paper Publishing Acceptance

Dear Ihsan Sahib, Tawfiq Alasadi

We are delighted to inform you that your manuscript entitled "**Deep Learning for Image Forgery classification based on Modified Xception Net and Dense Net**" has been accepted in the 9th International Conference of Applied Science and Technology (ICAST2021) Conference Proceeding.

The accepted manuscripts will be published in the AIP conference proceedings which is Scopus indexed and has an impact factor.

Once your article is uploaded online, a notification email will be sent to you.
If you have any enquiry, please, do not hesitate to contact us.

Yours Sincerely,

Karbala University
Dean Office
Science College

Jasim Alawadi

Assist. Prof. Dr. Jasim Hanoon Hashim Al-Awadi
Chairman of ICAST 2021
Dean of the Faculty of Science, University of Kerbala, Karbala, Iraq.
Phone: +9647831556219
E-mail: jasem.h@uokerbala.edu.iq

الخلاصة

يمكن تعريف تزييف الصور على أنه تلاعب أو تزوير أو تقليد لصورة ، و يعتبر تزوير الصور الرقمية في الوقت الحاضر من أكثر المشاكل التي تحتاج إلى حل ، خصوصا بعد استخدام الصور الرقمية في معظم المنصات الرقمية . وفي نفس الوقت ، فقد تطورت برامج معالجة الصور ، ومن ناحية أخرى ، فان التقنيات الجديدة للذكاء الاصطناعي والشبكات العصبية العميقة باستخدام شبكات الخصومة التوليدية **(GANs)** التي تنتج صور باستخدام الكمبيوتر، حيث يمكن للشبكات الحديثة من شبكة **GAN** أن تولد صور أو مقاطع فيديو مزيفة وغير حقيقية ولا يمكن اكتشافها بالعين المجردة . وللتغلب على هذه المشكلة فإن انظمة الكشف عن الصور المزيفة اصبحت مهمة. ولكنها تعاني من قلة الأداء العالي بسبب الدقة العالية والاتساق العالي في الصور المولدة من شبكات **GAN**، ولذلك فان الهدف من هذه المنهجية المقترحة هو الكشف عن الصور المزيفة، بمعنى آخر كيفية جعل الحاسوب يكتشف الصور المزيفة بشكل أفضل من الانسان.

تتكون المنهجية المقترحة من خمس مراحل رئيسية على النحو التالي: المعالجة المسبقة، وتهيئة البيانات، واستخراج خصائص الصورة، واتخاذ القرار، والمعالجة اللاحقة بتحديد التزييف. تتضمن المرحلة الأولى من المعالجة المسبقة قبل تدريب البيانات تغيير حجم الصور، وتحويل نظام ألوان الصورة من RGB إلى نظام ألوان YCbCr، وأنشاء مجموعة بيانات أخرى لأجزاء الصور. تقوم المرحلة الثانية بتهيئة البيانات التي تتضمن تقسيم مجموعة البيانات الصور ومجموعة بيانات اجزاء الصور إلى تدريب والتحقق من الصحة والاختبار. المرحلة الثالثة هي استخراج الخصائص، بناءً على شبكتي CNN عميقتين تعتمدان بشكل اساسي على التلافيف القابلة للفصل، لتمثيل الصورة كمتجه للميزات من خلال تعلم الشبكات على ميزات الصور لكل من الصور الحقيقية والمزيفة في مرحلة التدريب وعمل تنبؤات بناءً على الميزات الناتجة. ثم تدريب CNN خاصة لتحديد اماكن التزييف على معرفة خصائص أجزاء الصور بالشبكة واكتشاف الأجزاء المزيفة من الصور المزيفة. تتمثل المرحلة الرابعة باتخاذ قرار وحساب معدل لكل تنبؤ من شبكتي التصنيف في مرحلة الاختبار في الوقت الحقيقي وتحديد الصورة حقيقية أو مزيفة. المرحلة الخامسة هي خطوة المعالجة اللاحقة بعد اكتشاف أن الصورة مزيفة، وذلك بتأطير الأجزاء المزيفة من الصورة المزيفة باستخدام شبكة CNN الثالثة الخاصة بالتحديد.

اعتمدت هذه الأطروحة مجموعة بيانات "١٤٠ ألف وجه حقيقي ومزيف"، حيث تحتوي على ٧٠ ألف صورة حقيقية، و ٧٠ ألف صورة مزيفة تم إنشاؤها باستخدام شبكة StyleGan التي يصعب اكتشاف معظم الصور المزيفة الناتجة بنظام الرؤية البشري. أخيراً، حققت دقة التصنيف للنظام المقترح ٩٩,٩٩٪ بالإضافة إلى

٩٢,١٣٪ في عملية تحديد التزييف، وأخراً بعد تطبيق فكرة تحسين التعلم على شبكتي التصنيف حققت كل من شبكتي التصنيف على دقة ١٠٠٪ وشبكة التحديد ٩٢,٢٠٪.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعـــة بـابـــل
كلية تكنولوجيا المعلومات - قسم البرمجيات

نموذج التجميع للكشف عن تزوير الصور بالاعتماد على شبكات Xception

رسالة مقدمة إلى

مجلس كلية تكنولوجيا المعلومات - جامعة بابل كجزء من متطلبات

نيل درجة الماجستير في تكنولوجيا المعلومات / البرمجيات

من قبل

أحسان صاحب عبد الشهيد محمد

بإشـــراف

أ.د. توفيق عبد الخالق الاسدي

١٤٤٣ هـ ٢٠٢٢ م