

DeepFake Detection Using Various Deep Learning Techniques

A DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE
REQUIREMENT FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
INFORMATION SYSTEMS

Submitted by:
Pradhumn Singh Sisodia
2K20/ISY/15

Under the supervision of

Dr. PRIYANKA MEEL
ASSISTANT PROFESSOR
DEPARTMENT OF INFORMATION TECHNOLOGY



DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Bawana Road, Delhi-110042

MAY, 2022

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, Pradhumn Singh Sisodia, Roll No. 2K20/ISY/15 student of M. Tech., Information Systems, hereby declare that the major project titled "DeepFake Detection Using Various Deep Learning Techniques" which is submitted by me to the Department of Information Technology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi

Date: May 26, 2022



**Pradhumn
Singh Sisodia**

(2K20/ISY/15)

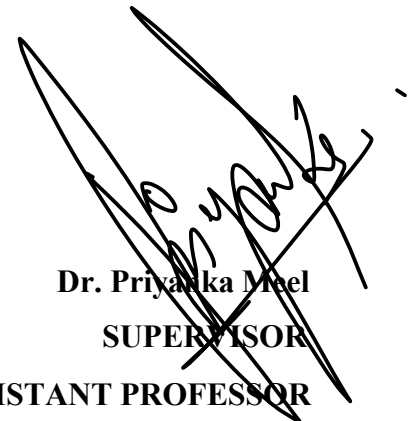
DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL
UNIVERSITY
(Formerly Delhi college of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the major project titled “DeepFake Detection Using Various Deep Learning Techniques” which is submitted by Pradhumn Singh Sisodia, Roll No. 2K20/ISY/15 Information Technology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date: May 26, 2022



Dr. Priyanka Meel
SUPERVISOR
ASSISTANT PROFESSOR

DEPARTMENT OF INFORMATION TECHNOLOGY

ACKNOWLEDGEMENT

I express my gratitude to my major project guide Dr. Priyanka Meel, Assistant Professor, IT Dept., Delhi Technological University, for the valuable support and guidance she provided in making this major project. It is my pleasure to record my sincere thanks to my respected guide for his constructive criticism and insight without which the project would not have shaped as it has.

I humbly extend my words of gratitude to other faculty members of this department for providing their valuable help and time whenever it was required.

Pradhumn Singh Sisodia

Roll No. 2K20/ISY/15

Mtech (Information Systems)

Abstract

Deepfakes is a face swapping technique that allows anyone to change faces in a video with incredibly realistic results. But when used nefariously, this strategy can have a substantial influence on society, for example, by distributing bogus news or encouraging cyberbullying. As a result, the capacity to detect deepfakes is a critical concern. We address the subject of deepfakes detection in this research by detecting deepfakes in video frames. Existing research in the field of deepfake detection reveals that the increased obstacles given by new deepfake movies make detection approaches more difficult to detect. In this study we performed experiments using various SOTA architectures on DFDC dataset and then after comparing the performance of those both on accuracy and time to train ,found architecture which shows a perfect balance of accuracy and low computation time needed to train. The final proposed solution uses Efficient Net V2 as backbone in network to obtain competitive results.

Keywords: Deepfake, Deepfake detection, SOTA architectures , binary classification, EfficientNets.

Table of Contents

CANDIDATE’S DECLARATION	ii
CERTIFICATE	iii
ACKNOWLEDGEMENT	iv
Abstract.....	v
Table of Figures.....	viii
List of Tables	ix
Chapter 1	10
INTRODUCTION.....	10
1.1 Deepfakes- Study.....	12
1.1.1 Types of Deepfakes.....	12
1.1.2 Deepfake Generation	15
1.1.3 Deepfake Detection.....	19
Chapter 2.....	22
Background and Related Work.....	22
2.1 Background.....	22
2.2 Related Work.....	31
2.2.1 VGG16.....	32
2.2.2 MesoNet.....	33
2.2.3 Xception Net.....	34
2.2.4 Vision Transformer.....	35
2.2.5 WS-DAN.....	36
2.2.6 MobileV2Net.....	37
2.2.7 Efficient Net	38

Chapter 3	39
Proposed Model.....	39
3.1 EfficientNet V2.....	39
3.2 Proposed model Architecture.....	41
Chapter 4	42
Implementation	42
4.1 Environmental Setup.....	42
4.2 Dataset Used.....	43
4.3 Data Preprocessing.....	43
4.4 Training & Validation.....	45
Chapter 5	46
Results And Evaluation	46
Chapter 6	50
Conclusion.....	50
References	51

Table of Figures

Figure 1 Original Vs Deepfake Source: Google images	10
Figure 2:Papers published over deepfake detection source:app.dimensions.ai.... Error! Bookmark not defined.	
Figure 3:New Face Synthesis	12
Figure 4:Face Swapping of different actors	13
Figure 5:Face and Voice forgery of Putin.....	14
Figure 6:Attribute Manipulation.....	14
Figure 7:Deepfake Generator (GAN)	15
Figure 8:Basic Working of Autoencoder	17
Figure 9:Auto encoder over Multiple Images	17
Figure 10:Swapping Faces using Autoencoder.....	18
Figure 11: Classification Of Deepfake Detection Techniques	20
Figure 12:Basic Working of Convolution operation	22
Figure 13:Convolutional Networks(CNN)	23
Figure 14:Max Pooling operation.....	24
Figure 15:Average Pooling Operation	24
Figure 16:ReLU and Leaky ReLU.....	25
Figure 17:Comparison of Sigmoid and Softmax	26
Figure 18:Working of Depthwise Convolutions	28
Figure 19:Working of Depthwise Separable Convolution.....	29
Figure 20: VGG16 Architecture	32
Figure 21:MesoNet Architecture.....	33
Figure 22:Xception Net architecture	34
Figure 23:Deepfake Detection using Vision Transformer	35
Figure 24: WSDAN Architecture.....	36
Figure 25: MobileV2 Net architecture	37
Figure 26: ReLU6 Activation Function used in MobileV2Net	37
Figure 27: Efficient Net Architecture.....	38
Figure 28: EfficientNetV2 Architecture.....	40
Figure 29:Performance Comparison Of EfficientNetV2 against other Architectures	40
Figure 30: Proposed Model	41
Figure 31: Frame Captured in one of videos	43
Figure 32:Face Detected and Cropped in data Preprocessing.....	44
Figure 33:Model Architecture.....	45
Figure 34: Training and validation Loss till 20 epochs	47
Figure 35:Training Accuracy and Validation accuracy of Proposed Model.....	47
Figure 36:Confuion matrix in percentage terms	48

List of Tables

Table 1:Deepfake Datasets.....	19
Table 2:Related Works	31
Table 3:Confusion Matrix	48
Table 4:Comparison of Various Models	49

Chapter 1

INTRODUCTION

Deceptive media has been a serious concern in recent years, especially after the development of Deepfakes, which are photographs and videos that have been altered using either generative adversarial networks (GANs) [2]. (GAN) or autoencoders (AE) [1] which are able to generate deepfake faster than manual manipulation. If you have access to massive volumes of data, making realistic modified media assets might be relatively simple with this technique. Some of the uses include cinematography, videography, electronic games, and augmented worlds. This very technology, on the other hand, may be used for nefarious reasons, such as blackmailing individuals with false pornographic films or launching fake-news campaigns to alter public opinion. It may, in the long run, damage trust in the media, particularly serious and reliable sources. Some fakes are easy to spot since they were manufactured for fun and involve well-known celebrities and politicians in unusual situations. Furthermore, on the web, both the original and the altered version are frequently available, reducing any dispute about legitimacy. Whenever a video represents a slightly lesser popular person and only the altered image is generally accessible, ensuring digital integrity becomes much more difficult. Today, high-quality deepfakes are excellent enough that most people can't detect the difference, especially when the films are veiled by the poor resolution of social media video sharing. Even professionals have difficulty visually distinguishing the finest deepfakes from the actual video. This necessitates the development of new detection technologies. Deepfakes allow people's voices and appearances to be duplicated. Just about anything can be made that replica say or do by a deepfake designer. Below is an example of how good the deepfakes have become.



Figure 1: Original Vs Deepfake Source: Google images

Countermeasures against face forgeries in digital media must be developed to combat this problem. As a result, the research community has been working hard to create methods for identifying facial modification in photos and movies. The number of seminars and conferences on media forensics is steadily expanding. The number of papers written on topic of deepfake detection has expanded quite significantly in last 2- 3 years, as illustrated in Figure 2.

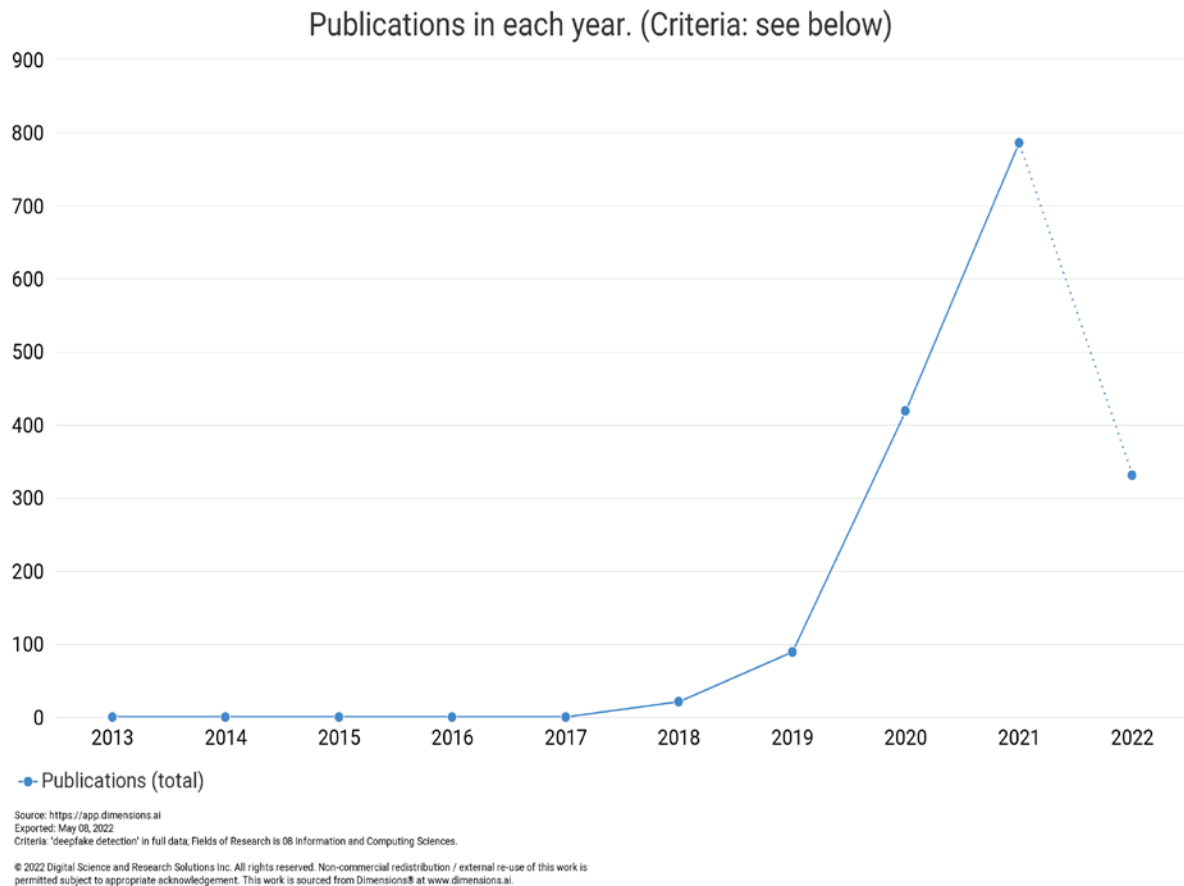


Figure 2: Papers published over deepfake detection source: app.dimensions.ai

Furthermore, challenges are conducted on a regular basis in both industry and academia to combat the threat of deepfakes. From the perspective of computer vision, this chapter presents an in-depth examination of deepfake production and detection methods. The remainder of the chapter is organized as follows; the first portion of this chapter will look at types of deepfakes and several deepfake generation strategies employed by various researchers. The next section will go over several deep learning algorithms for detecting these deepfakes.

1.1 Deepfakes- Study

This section aims to provide a brief overview of the types of deepfakes and manipulations possible. Also, it relates to the field of deep learning by explaining architecture responsible for the synthesis of deepfakes, GANs and Autoencoders. Lastly to get overview of some of the methods , datasets used to detect deepfakes and some important concepts used in proposed model.

1.1.1 Types of Deepfakes

New Face Synthesis

This manipulation generates entire photographs of non-existent faces. Many different industries, such as the video game and 3D modeling industries, may benefit from this manipulation, but it could also be used to spread disinformation for negative purposes, such as the producing quite realistic fake profiles on social media. Every time we visit a site like this, it creates new random faces. Thispersondoesnotexist.com .StyleGAN [3] is one such GAN used for generating new faces.



Figure 3:New Face Synthesis

Face Swapping

This essentially entails replacing one person's face with that of another person, usually with people who have similar facial traits, to get a better image. This may be done in two ways: one utilizing computer graphics and the other using a revolutionary deep learning approach.



Figure 4: Face Swapping of different actors

Forgery of a person's face and voice

The artificial facial motions of the desired individual are created using face reenactment. It manipulates the aspects of the target person's facial pictures, such as the movement of their lips, eyebrows, eyes, and head tilting, in order to deform their facial emotions. As a result, speech forgeries are followed by speech synthesis, which creates a model of the target person's voice. Speech forging then synchronizes the altered facial expressions, text, and voice. In the generated phony film, the targeted entity seems to say something he or she never uttered. Furthermore, current enhanced speech synthesis algorithms provide greater viability in speech modulation. For example, consumers may choose a voice of any age and gender.



Figure 5: Face and Voice forgery of Putin

Attribute Manipulation

Face manipulation, also known as facial editing or face retouching, is altering some elements of the face, such as hair or skin color, gender, age, and the addition of spectacles. Customers might utilize this technology to try on a range of things in a virtual environment, such as cosmetics and lipstick, spectacles, and haircuts.



Figure 6: Attribute Manipulation

Hybrid Applications

Existing deep learning approaches may be used to develop hybrid applications by combining multiple deepfakes. Nirkin et al. [4] developed a GAN model for real-time face swapping and reenactment. Users may swap faces and modify their age, gender, smile, and haircut with commercial FaceApp [5] smartphone applications.

1.1.2 Deepfake Generation

The majority of online deepfakes are created using deep learning techniques such as Auto-Encoder or Generative adversarial Networks (GANs), which are made possible by access to massive amounts of data.

Generative Adversarial Networks

Generative adversarial networks (GANs) allow for the learning of deep representations without the need for significant tagging of training material. They do this by using a robust technique utilizing two networks to derive backpropagation signals. GANs may learn representations that can be utilized in a variety of applications, such as style transfer, semantic picture editing, image synthesis, image super-resolution, and classification.

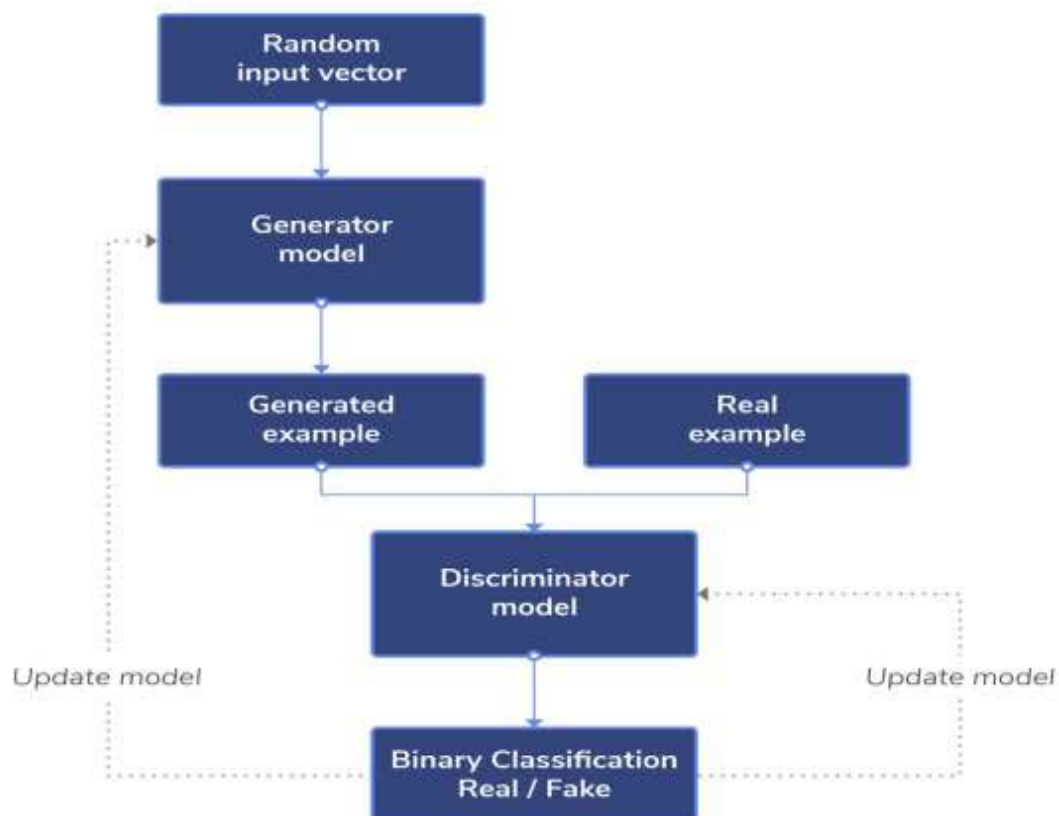


Figure 7: Deepfake Generator (GAN)

GAN is basically two higher-level neural networks that feed into one other. One creates increasingly comprehensive data, whereas the other steadily increases its classification abilities. GANs are an innovative method to train a generative model by framing the problem with two supervised learning sub-models: the generator model, which we train to produce new instances, and the discriminator model, which attempts to categorize examples as true or false. Generator: A model that produces fresh possible instances from issue area. Discriminator: A model for classifying occurrences as true or false. The two models basically play a zero-sum game until the discriminator model is fooled for about half the time. This shows that the generator model will generate a trusted instance. Generators are typically built using deconvolutional neural networks, which enable the ability to generate data. (For example, upsample the feature map to create a new image). Discriminators are built using common CNNs for the ability to break down data (such as images) into feature maps and classify the data.

Some commonly used types of GAN are as follows

- 1) Deep convolutional GAN (DC GAN) [6] - This is a deep convolutional GAN. It is one of the most widely used, powerful, and effective GAN architecture kinds. ConvNets are used instead of a Multi-layered perceptron to implement it. ConvNets are formed using a convolutional stride and no max pooling, and the layers in this network are not entirely linked.
- 2) Conditional GAN (CGAN) and Unconditional GAN (CGAN) [7]- Conditional GAN is a deep learning neural network with certain extra parameters. Labels are also added to Discriminator inputs to assist the discriminator inappropriately classifying the input and preventing the generator from filling it.
- 3) Least Square GAN (LSGAN) [8] — This is a GAN that uses the least-square loss function as the discriminator. The Pearson divergence is minimized when the LSGAN objective function is minimized.
- 4) Auxiliary Classifier GAN(ACGAN) - ACGAN is a more sophisticated form of CGAN. It specifies that the discriminator should not only categorize the picture as real or false but also offer the input image's source or class label.

Auto Encoders

In basic words auto - encoder is a neural net that learns to replicate input as output. It has an internal latent layer that specifies the pattern used to interpret the input and is made up of two basic components: an encoder that maps the input into the pattern and a decoder that maps the code back to the original input reconstruction.

To execute the copying operation flawlessly, it would just replicate the signal, which is why autoencoders are generally constrained in ways that compel them to approximately recreate the input, keeping only the most relevant features of the copy data.

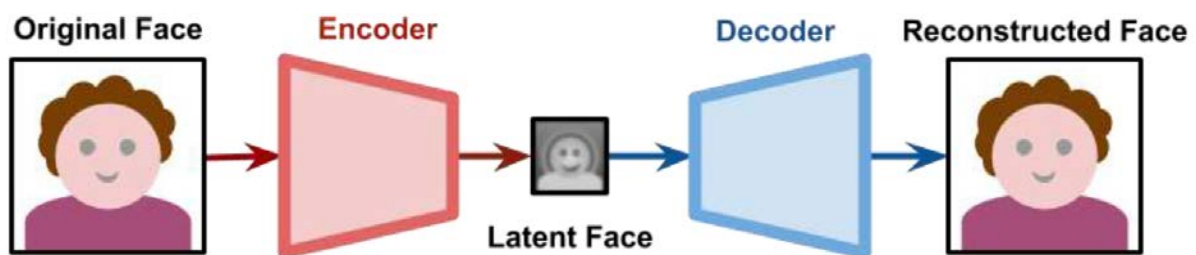


Figure 8: Basic Working of Autoencoder

An encoder is seen in the graphic above (in this particular case, a face). The effect is a lower-dimensional representation of the same face, which is also known as the base vector or latent face. Depending on the network design, the latent face may not appear to be a face.

When the latent face is sent through a decoder, it is rebuilt. Because the autoencoders are lossy, it's unlikely that the rebuilt face is as detailed as the original.

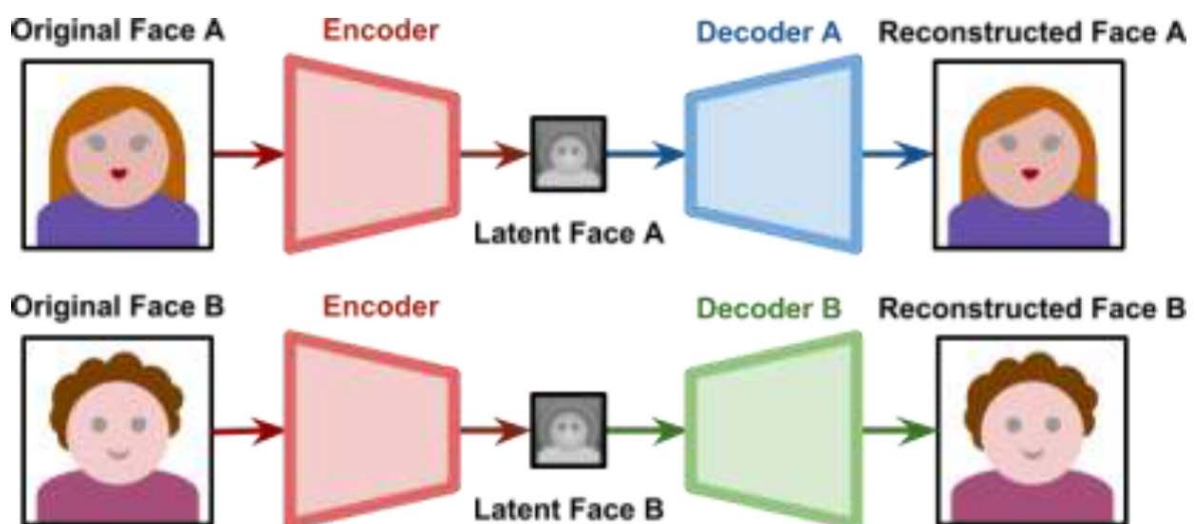


Figure 9: Auto encoder over Multiple Images

It's important to remember that training two autoencoders individually is incompatible. The latent faces are focused on particular aspects that each network has judged significant throughout its training procedures. However, the latent space of two auto-encoders trained on distinct faces is different. The swapping technique is made feasible by requiring all latent faces to be programmed on the same features. Both networks shared the same encoder, which was addressed by using two distinct decoders. During the training phase, those two networks are handled individually. Only A-sides are used to train Decoder A, while B-sides are used to train Decoder B. Nonetheless, all latent sides are generated by the same encoder. This necessitates the encoder's recognition of shared properties on both sides. As every face has a similar structure, it is not unrealistic to assume that the encoder would learn the "face" definition itself.

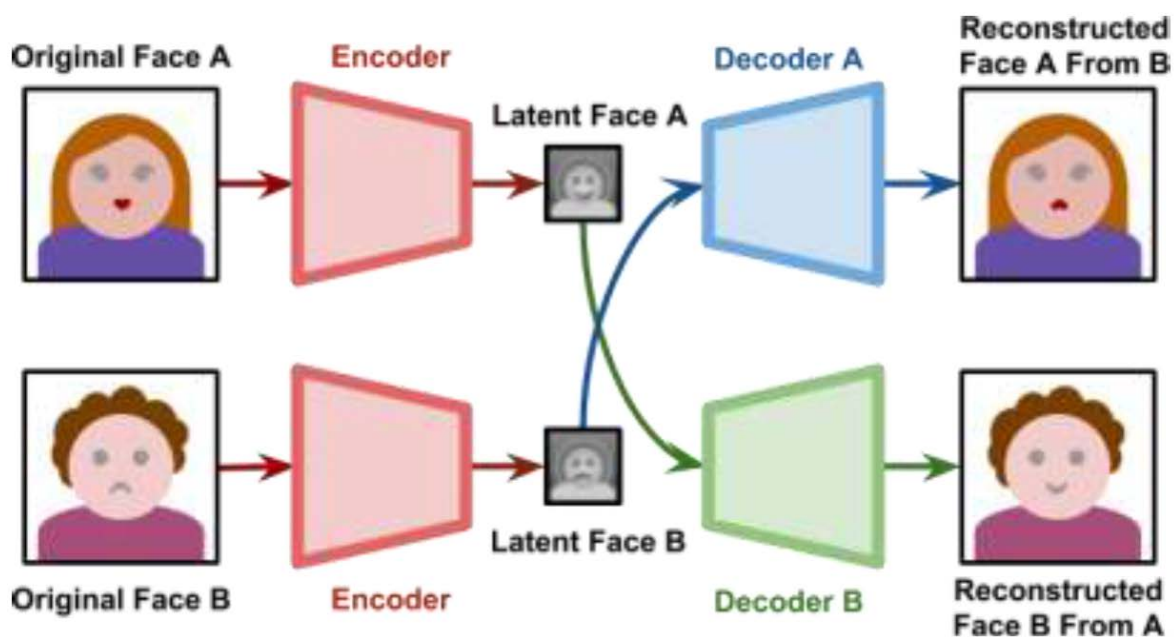


Figure 10: Swapping Faces using Autoencoder

A latent face formed by subject A is passed from subject B to decoder B when the training procedure is completed. The decoder B will try to reconstruct topic B from the information on subject A, as shown in the diagram. If the network is sufficiently generalized, the latent space would be facial expressions and orientations. This means that for object B, a face with the same emotion and position as subject A is created.

1.1.3 Deepfake Detection

Over the recent years several approaches have been proposed by the researchers to detect deepfakes also many competitions are held to counter the problem of deepfakes which gives researchers good amount of data to train their deep learning models on. Some of the data sets used and available to researchers are given in the table below.

Table 1: Deepfake Datasets

Datasets	No. of Real Videos	No. of Fake Videos	Total Number of Videos	Year Published
UADFV	49	49	98	2018.11
DF-TIMIT-LQ DF-TIMIT-HQ [9]	320	320	640	2018.12
FaceForensics++ [10]- DF	1000	1000	2000	2019.01
DFD[10]	363	3068	3431	2019.09
DFDC-preview [11]	1131	4113	5244	2019.10
Celeb-DF [12]	590	5639	6229	2019.11
Wild Deepfakes [13]	3805	3509	7314	2021.01

Deepfake detection is frequently conceived binary classification issue, with classifiers used to differentiate between authentic and manipulated movies. In order to improve classification algorithms, these approaches require a huge library of actual and fraudulent videos. Although some deepfake detection techniques can also be applied on images but most of them have been applied to videos.

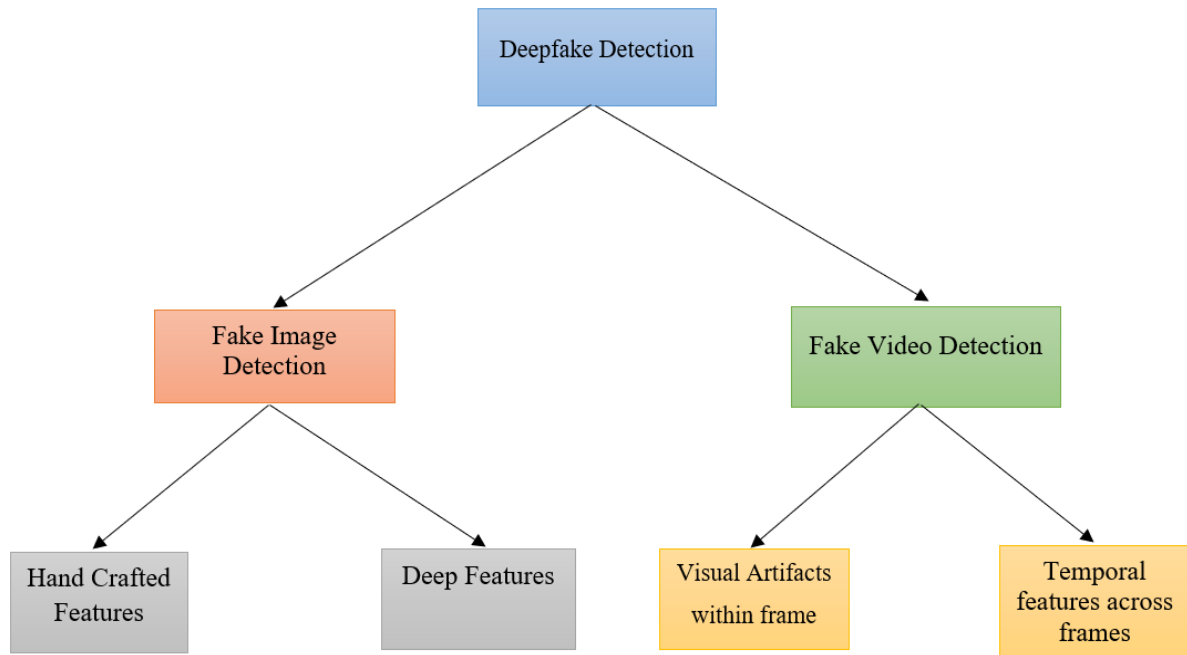


Figure 11: Classification Of Deepfake Detection Techniques

Detection of image Deepfakes

Methods for detecting deepfakes were presented immediately after it was used for nefarious purposes. Early attempts focused on artefact-derived handcrafted qualities and defects in the fake picture generation process. In the recent years deep learning has been used to detect deepfakes by automatically extracting notable and dissimilar properties. Despite the fact that GAN research is ongoing and numerous new extensions are regularly offered, most studies on the detection of GAN-produced images ignore the detection models' generalization capability. To eliminate low level information from GAN pictures, [14] employed data augmentation methods, such as Gaussian noise and Gaussian blur. This enhances the statistical similarity between real and fake photographs at the pixel level, allowing the forensic classifier to learn more basic and significant characteristics and generalize better than previous image forensics methods. Deepfake detection using GAN is modelled as a hypothesis testing issue.

Video Deepfake Detection

Most image detection algorithms cannot be employed for videos because of the significant loss of frame data following video compression. Moreover, videos include temporal properties that change between frames, making it difficult for techniques built to identify solely still fraudulent pictures to detect them. This section examines deepfake video detection methods and divides them into two categories: methods that use temporal information and methods that look at visual artefacts inside frames.

Detection Methods based on Temporal Features across frames

Based on the discovery that temporal coherence is not adequately preserved in the synthesis process of deepfakes, Sabir et al. [15] exploited these features of video streams to detect deepfakes. Because video editing is done frame by frame, low-level aberrations generated by face adjustments are called temporal artefacts with inconsistencies across frames. A recurrent convolutional model (RCN) based on the neural network DenseNet [16] was created to use temporal differences between frames. Deepfake movies also feature intra-frame errors and temporal anomalies across frames, according to [17]. They then suggest an approach that uses Convolutional Neural Network and long-term memory to detect deepfake videos. The accumulative neural network extracts frame-level characteristics, which are subsequently integrated into the LSTM to provide a time sequence descriptor. Finally, utilizing sequence descriptors, a dense network is employed to identify ethical films from authentic films.

Detection Method based on visual artifacts within a frame

This section looks at the other method of obtaining discriminant characteristics by decomposing films into frames and looking at visual artefacts inside single frames. To distinguish between fraudulent and legitimate films, these characteristics are divided into either a deep classifier or a shallow classifier. Deepfake movies are typically made with low resolutions, necessitating a face warping strategy having operations like rotation and shearing to match the actual configuration. This procedure leaves artefacts that CNN models like VGG16 [18], ResNet50, ResNet101, and ResNet152 [19] may identify due to the resolution discrepancy between the warped face region and the surrounding environment.

Chapter 2

Background and Related Work

2.1 Background

Some of important concepts used in our proposed model are explained below :

Convolutional Nets

The convolutional neural network (CNN) is a type of neural network model that was designed to operate with two-dimensional image data, although it may also be used with one-dimensional and three-dimensional data. The convolutional neural network's core is the convolutional layer, which gives the network its name. This layer performs the "convolution" operation. In a convolutional neural network, a convolution is a linear process that involves product of set of weights with the input which is identical to a regular neural network. Because the technique was designed for two-dimensional input, the multiplication is done between an array of input data and a two-dimensional array of weights termed a filter or a kernel.

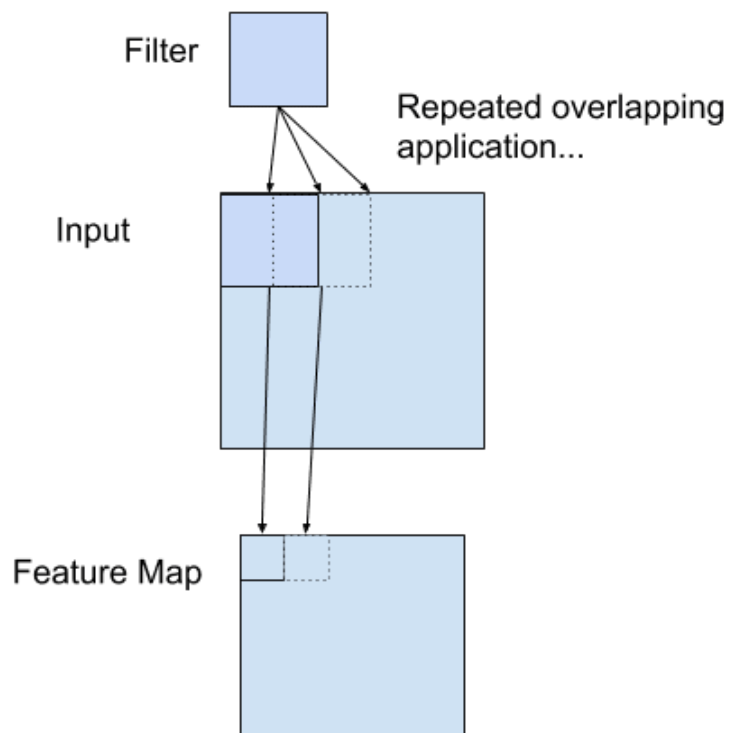


Figure 12: Basic Working of Convolution operation

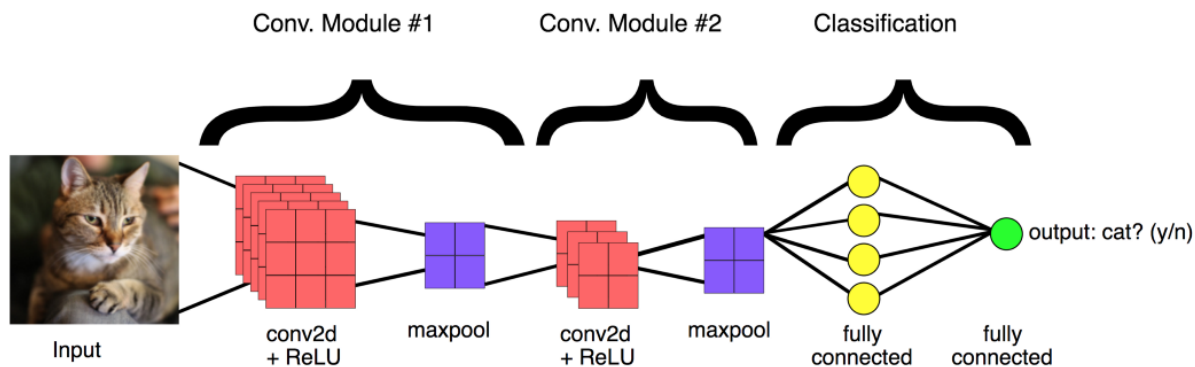


Figure 13: Convolutional Networks (CNN)

In general, every Neural Network used for image processing has the following layers:

Convolutional Layer, Pooling Layer, and Dense Layer are the layers that make up the input layer.

Convolution is a filter that is applied to a picture in order to extract features from it. Several convolutions are utilized to extract distinct aspects from the picture, such as edges and highlighted patterns. This convolution generates a filter of a certain size which is three cross three. Then filter is generated it begins element-by-element product from the image's top left corner. Multiplying items with the same index is known as element-wise multiplication. These calculated values are added together to produce a pixel value, which is then saved in the new matrix. This freshly created matrix will be used for further processing. As we add filters to the produced matrix, the size of the matrix shrinks.

Filter size of previous matrix + 1 = new matrix size

These feature matrices are supplied as input to the following layer.

Following the use of convolutions, there is a notion known as pooling. Pooling is a technique for reducing picture size. Pooling may be divided into two categories:

MAX POOLING -

Max Pooling is the process of picking the maximum value from a matrix of a given size. This approach is useful for extracting important characteristics or features that are highlighted in the image.

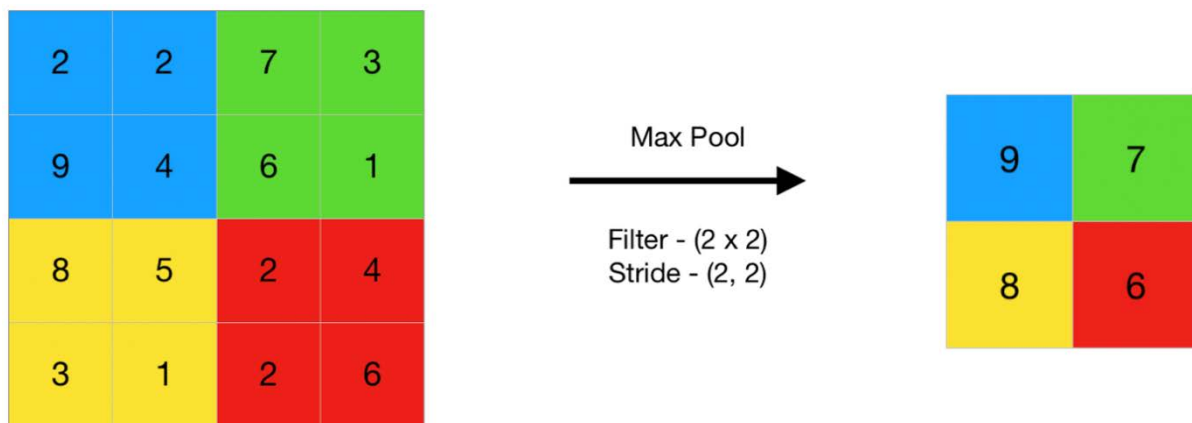


Figure 14: Max Pooling operation

Average Pooling –

Average Pooling: Average pooling is done by averaging all the pixel values in the pooling layer's matrix.

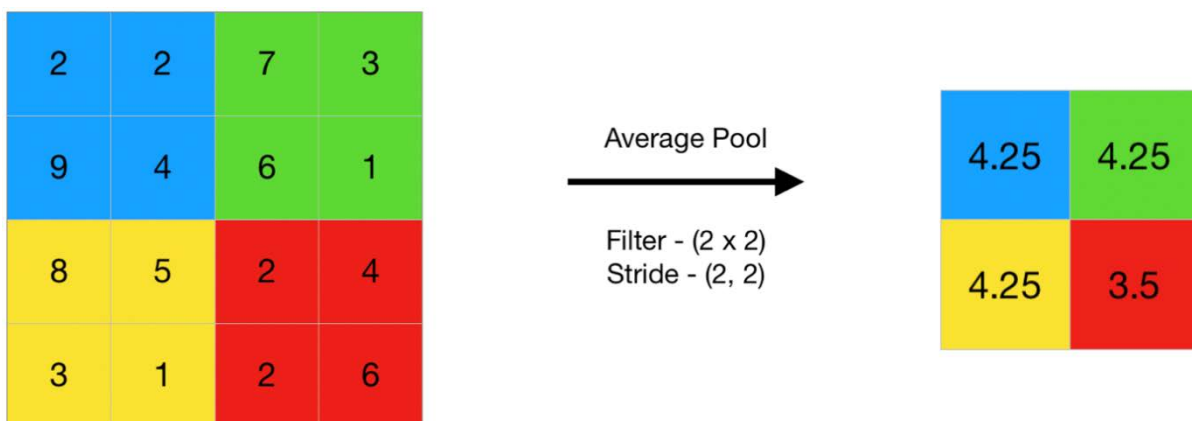


Figure 15: Average Pooling Operation

Max pooling is employed in the majority of circumstances because it performs substantially better than average pooling.

The input will be transferred to the dense layer after passing through all convolutional and pooling layers. Because the output of the convolutional layer is multi-dimensional, we can't transfer it directly to the dense layer because the dense layer requires input in a single-dimensional structure, such as a 1-D array which is done using `flatten()` function.

Dense Layer

The dense layer is a basic layer of neurons in which each neuron receives input from all neurons in the previous layer, thus the name. Dense Layers are used to recognize pictures based on the output of convolutional layers.

Activation function

1.ReLU & Leaky ReLU

The Rectified linear unit (ReLU) is now the often used activation function, extending from 0 to infinity. All negative values are converted to zero, and the conversion rate is so fast that it cannot appropriately map or fit into data, providing a difficulty. Instead of ReLU, the Leaky ReLU function is used to avoid this unfitting. The range of the Leaky ReLU is enlarged, which improves performance.

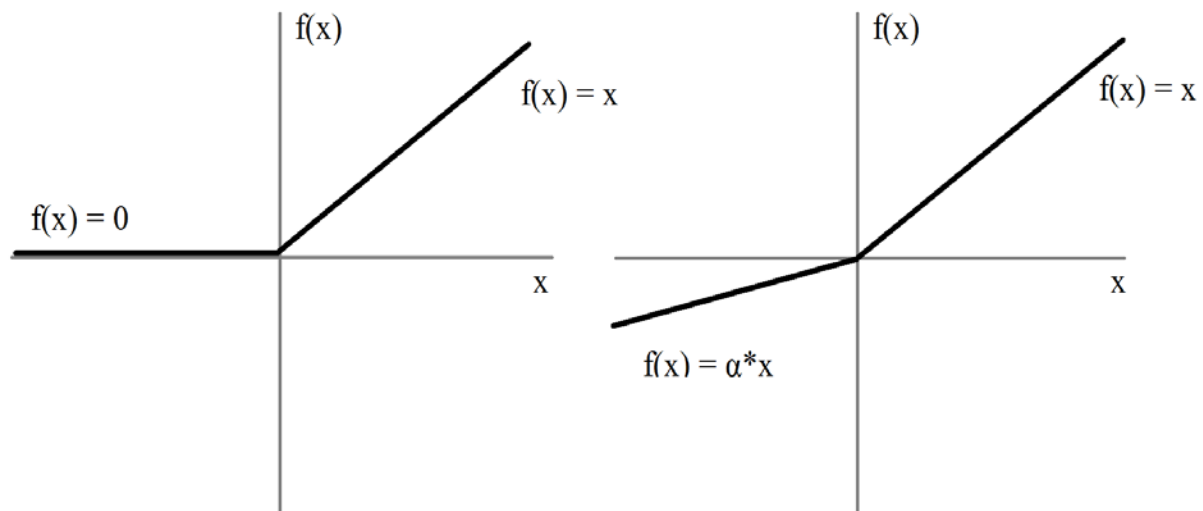


Figure 16:ReLU and Leaky ReLU

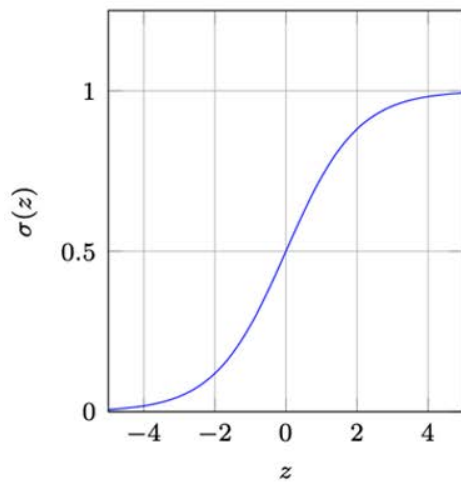
2)Sigmoid Activation Function

The sigmoid activation function is commonly used because it performs well. It is a probabilistic approach to decision making that lies between 0 to 1, so when there is need to make a decision or predict an output, this activation function is used because the range is the smallest, so prediction is much more precise.

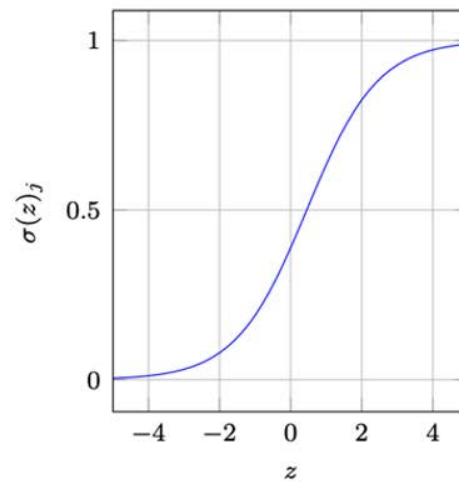
The sigmoid function has the following equation:

$$Y = \frac{1}{(1+e^{(-x)})}$$

The sigmoid function produces difficulty of its own which is known as the vanishing gradient problem, which arises when big inputs are converted between 0 and 1 and their derivatives become significantly smaller, resulting in unsatisfactory output. ReLU is used instead to tackle such problem.



(a) Sigmoid activation function.



(b) Softmax activation function.

Figure 17: Comparison of Sigmoid and Softmax

3) Softmax Activation Function

Softmax is mostly utilized in the final layer for decision making. Similar to sigmoid activation, softmax assigns values to input variables based on their weight, and the total of these weights slowly equals one. Both sigmoid and softmax are equally accessible for binary classification, however often softmax and cross-entropy together are utilized for multi-class classification problems.

Depth wise Convolutions

In convolutional neural networks, 2D convolutions are the most often used convolutional layer (CNN). The primary distinction between 2D convolutions and Depth wise Convolution is that 2D convolutions are applied to all/multiple input channels, but Depth wise Convolution is applied to each channel individually.

Methodology —

- 1) Three-dimensional input tensor is separated into different channels.
- 2) The input is convolved using a filter for each channel (2D)
- 3) Each channel's output is then layered to obtain the output for the complete 3D tensor.

Graphical representation of the same is given in fig no:2.7

Depth wise Separable Convolutions

Depth wise separable convolutions operate with kernels that cannot be "factored" into two smaller kernels, unlike spatial separable convolutions. As a result, it is more widely utilized.

The phrase "depth wise separable convolution" refers to the fact that it operates with both the spatial and depth dimensions — the number of channels. Three channels can be found in an RGB image. After a few convolutions, an image may have several channels. Each channel, for example, evaluates the intensity and other properties of each pixel for each color. A 128-channel image can be understood in 128 various ways.

Depth wise convolutions are frequently used in conjunction with a second step called Depth wise Separable Convolution. Filtering (all preceding processes) and Combining (combining the three colour channels to produce 'n' number of channels, as desired — in the example below, the three channels are combined to form a single channel output).

The Diagram for working of Depthwise separable Convolutions is illustrated in fig no: 2.8

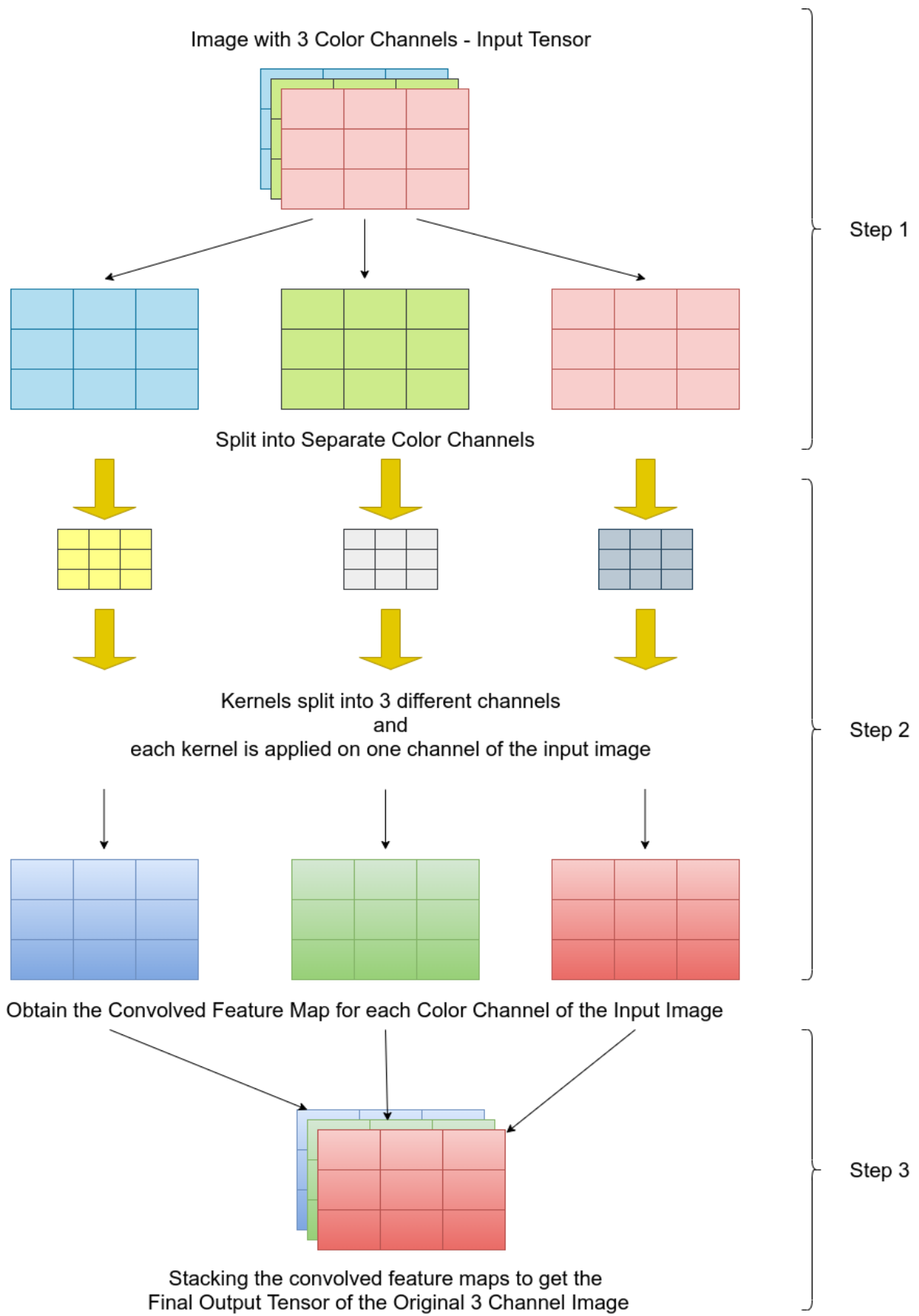


Figure 18: Working of Depthwise Convolutions

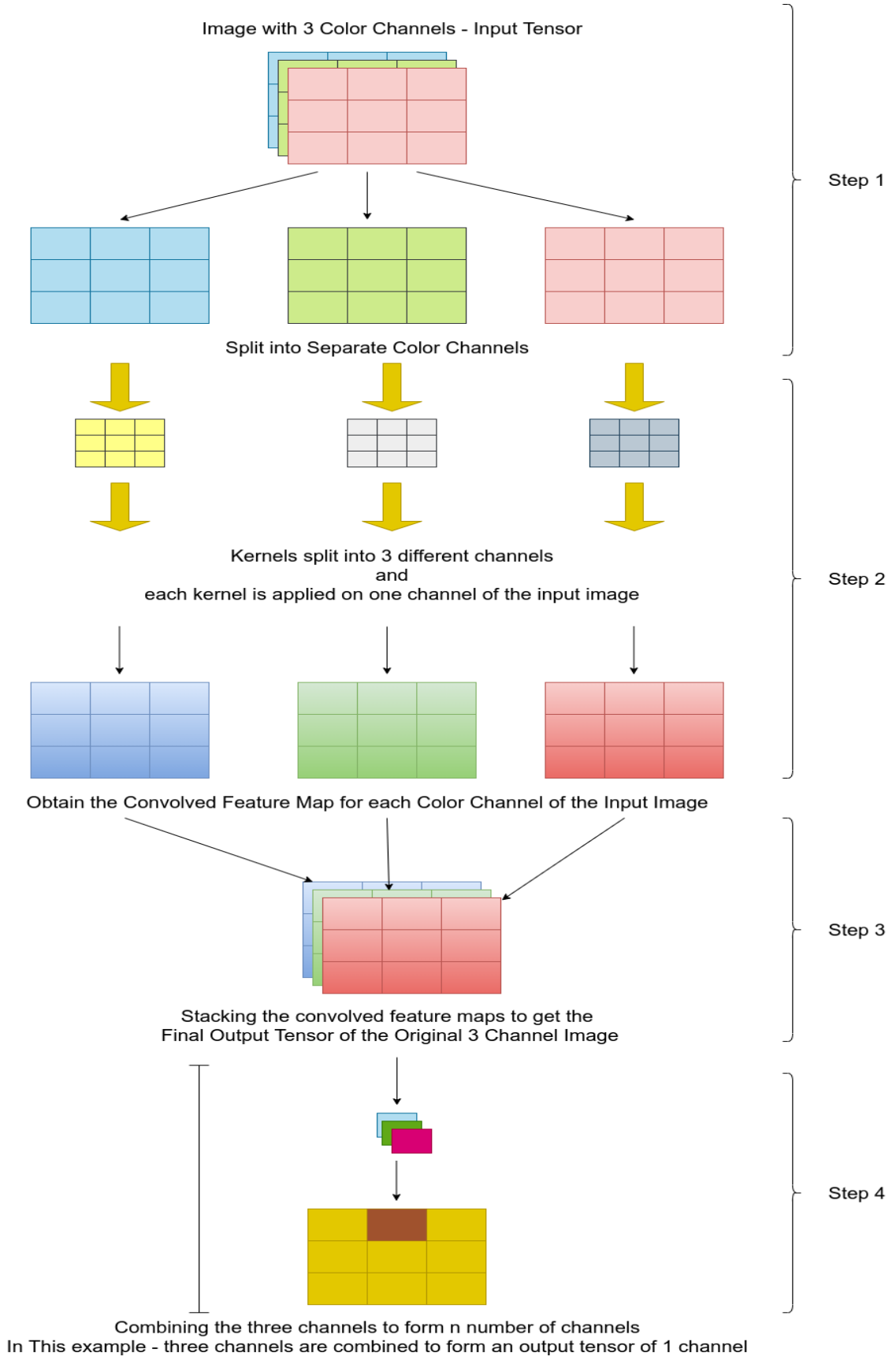


Figure 19: Working of Depthwise Separable Convolution

Comparison of 2D convolutions with Depth wise convolutions

Reason for Depth wise Separable convolutions perform better than simple 2D Convolutions:

Assume we have an input tensor with the dimensions $8 \times 8 \times 3$. The required output tensor is $8 \times 8 \times 256$ in size.

Convolutions in 2D —

$$(8 \times 8) \times (5 \times 5 \times 3) \times (256) = 1,228,800 \text{ multiplications}$$

Depth wise Separable convolutions

a. Filtering — Because the signal is split into single channels, a $5 \times 5 \times 1$ filter is required instead of a $5 \times 5 \times 3$, and because there are three channels, a total of three $5 \times 5 \times 1$ filters are required. $(8 \times 8) \times (5 \times 5 \times 1) \times (3) = 3,800$

b. Combining — Because the total number of channels needed is 256,

$$(8 \times 8) \times (1 \times 1 \times 3) \times (256) = 49,152$$

Total multiplications: $3,800 + 49,152 = 53,952$.

To achieve the identical result, a 2D convolution will take 1,228,800 multiplications, but a Depth wise Separable convolution will only require 53,952 multiplications.

This helps neural nets using depth wise separable convolutions achieve better results and also much faster as it requires less computations as compared to 2D convolutions. The primary distinction is that we change the picture 256 times in standard convolution. Every transformation requires a total of $5 \times 5 \times 3 \times 8 \times 8 = 4800$ multiplications. We only truly alter the picture once in the separable convolution — in the depth wise convolution. After that, we simply lengthen the converted picture to 256 channels. We can save processing resources by not having to convert the image many times.

2.2 Related Work

This section contains an overview of prior work in the topic of deepfakes detection, with ML models as potential solutions. Because of the quick pace of progress in the subject, a comprehensive review of all published approaches would be impossible. Instead, the focus of this part will be on a few of the models employed in the experiment.

As far as feasible, materials for this study's literature review were chosen from research articles published in academic journals and major conferences in the domains computer vision, and image processing.

Table 2: Related Works

S.no	Architecture	Authors	Year Published
1	VGG16[18]	Simonyan, Karen, and Andrew Zisserman.	2014
2	MesoNET [20]	Afchar, Darius, Vincent Nozick, Junichi Yamagishi, and Isao Echizen	2018
3	Xception [21]	Chollet, Francois	2017
4	Vision Transformer [22]	Khormali, Aminollah, and Jiann-Shiun Yuan	2022
5	WS-DAN [23]	Hu, Tao, Honggang Qi, Qingming Huang, and Yan Lu.	2019
6	MobileV2Net [24]	Sandler et.al	2018
7	Efficient Net [25]	Tan, Mingxing, and Quoc Le.	2019

2.2.1 VGG16

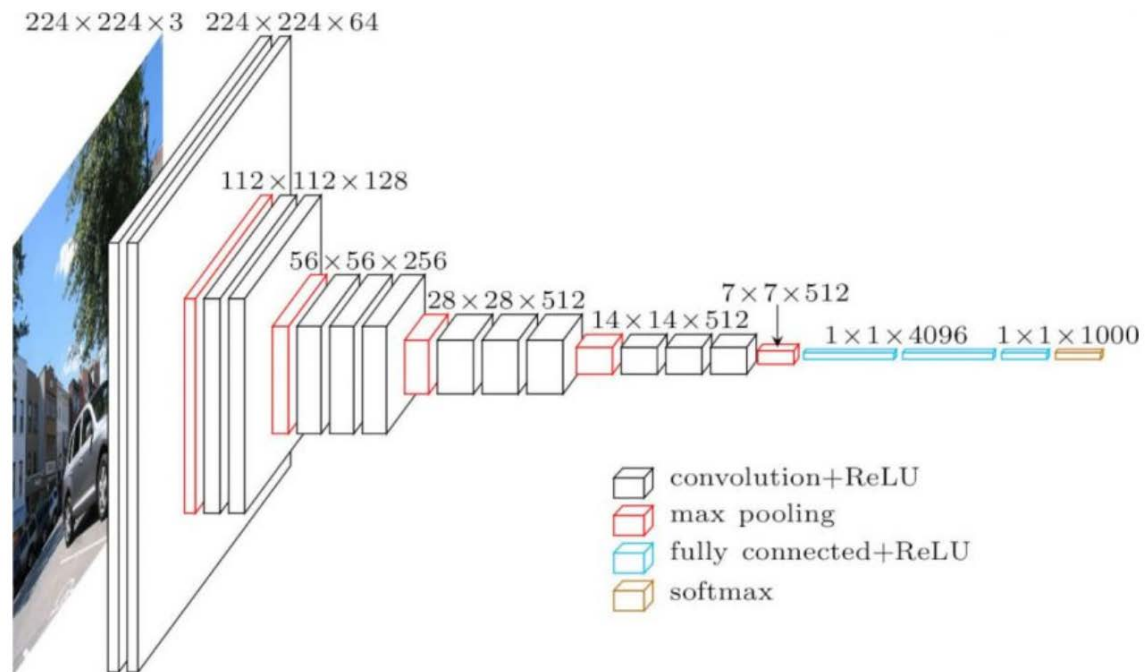


Figure 20: VGG16 Architecture

Karen Simonyan and Andrew Zisserman [18] of Oxford University's Visual Geometry Group Lab suggested VGG 16 in their work in 2014. In the 2014 ILSVRC competition, this model took first and second place in both categories. Object localisation is the first step in detecting objects in a picture from 200 different classifications. The second step is picture classification, which involves labelling each image with one of 1000 categories.

VGG 16's Challenges:

- Training is quite sluggish it took 3 weeks to train even on Nvidia Titan GPU
- VGG-16 trained imageNet weights are 528 MB in size. Resulting in substantial amount of storage usage rendering it inefficient.

2.2.2 MesoNet

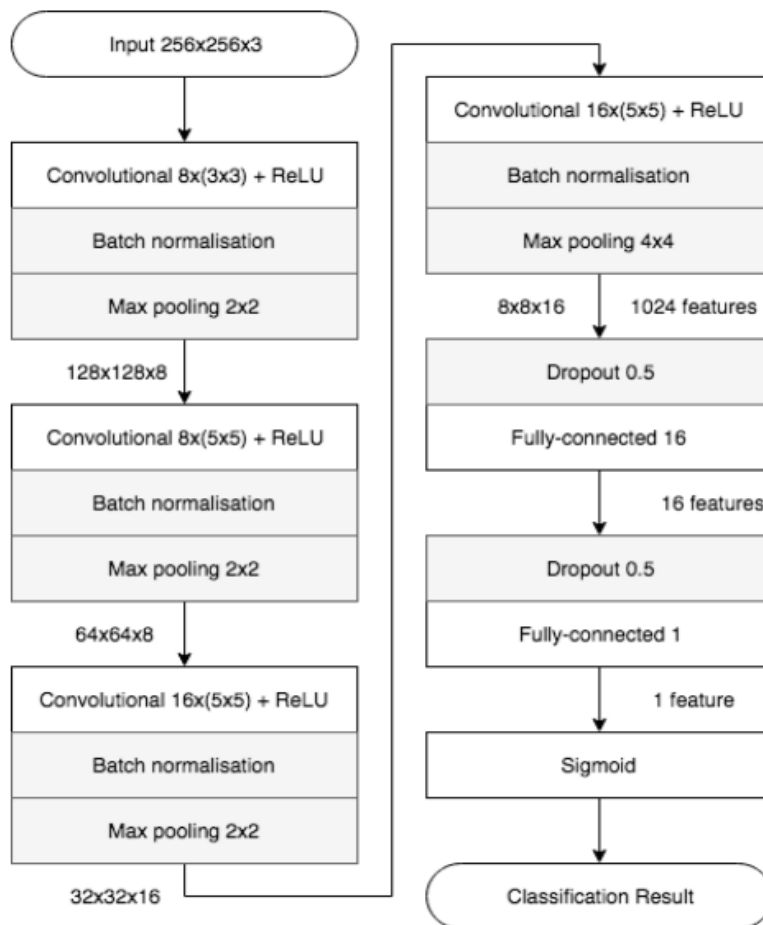


Figure 21: MesoNet Architecture

MesoNet [20] was proposed by Darius Afchar, Vincent Nozick, Junichi Yamagishi, Isao Echizen in 2018 to detect forgery in videos. The MesoNet uses eyeballs as a significant feature because the pixels in the eye area provide a very strong signal in actual pictures. Because produced images are hazy, while eyes are the most detailed element of genuine images, this is to be anticipated. The predictions are produced on a frame-by-frame basis for each facial picture taken from videos using the Viola-Jones detector, just like Deepfake's video generation method.

2.2.3 Xception Net

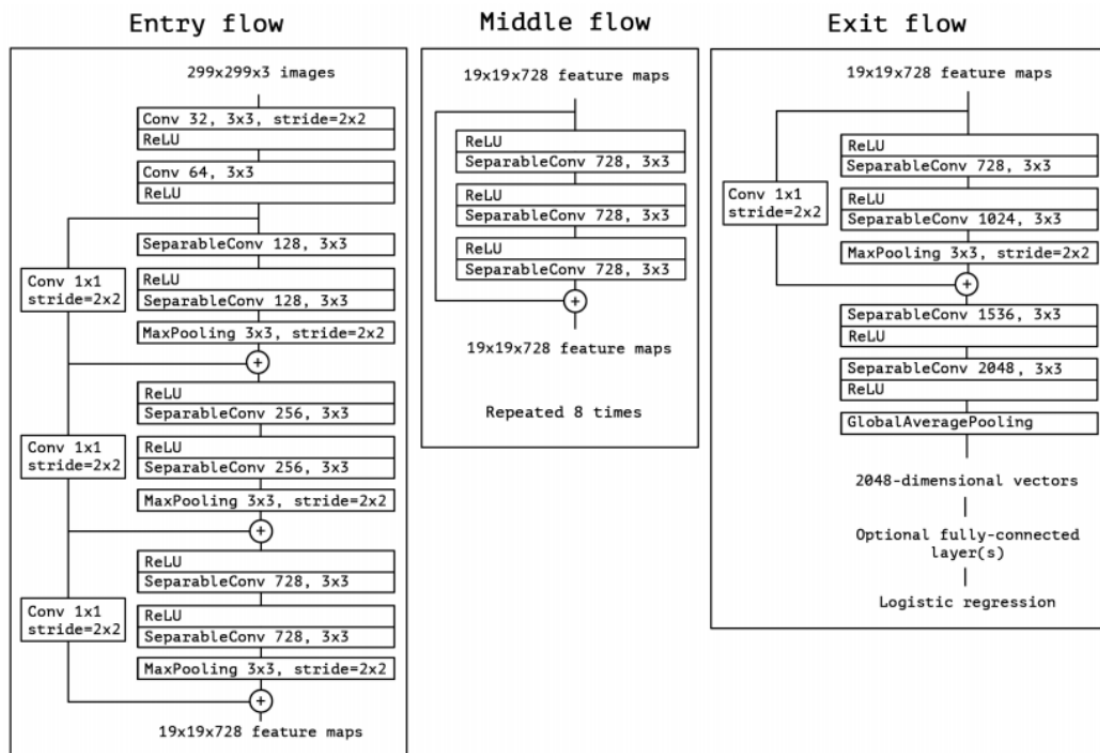


Figure 22: Xception Net architecture

Xception [21] Proposed by Google stands for extreme Inception Model which is also published by google .Leaving out the first and last modules of the Xception architecture, it has thirty six convolutional layers arranged into fourteen modules, all of which have linear residual connections surrounding them. The concept is based on the use of depth wise separable convolutions. First, 3x3 convolution is done to each of the input channels, extracting spatial information from each channel separately. After that, 1x1 convolution is used to extract data from a cross-channel dimension.

2.2.4 Vision Transformer

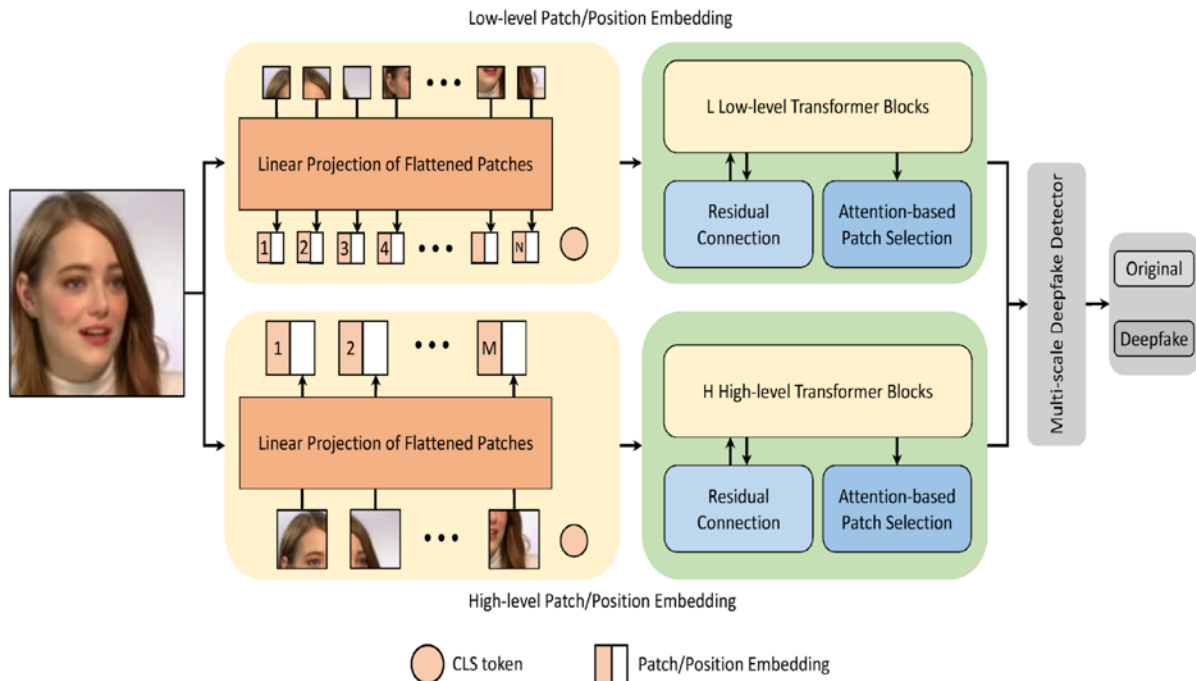


Figure 23: Deepfake Detection using Vision Transformer

The suggested deepfake detection system based fully on transformers has been developed which is described in brief. Distinctive qualities of transformer models on detecting concealed evidence of distortions from local picture attributes and the global interaction of pixels at distinct forgery scales are leveraged in the development of DFDT [22]. “The DFDT’s primary components are patch extraction and embedding, a multi-stream transformer block, attention-based patch selection, and a multi-scale classifier.”[22] Heo et al. [28] have suggested a strategy using distillation on vision transformer and which is built on EfficientNet [25] characteristics. Khan and Dai [26] introduced a video transformer for deepfake with twelve blocks of transformer and XceptionNet for image extraction. Wodajo and Atnafu [27] proposed a convolutional vision transformer that use CNNs as a feature extractor and a transformer block as a classifier.

2.2.5 WS-DAN

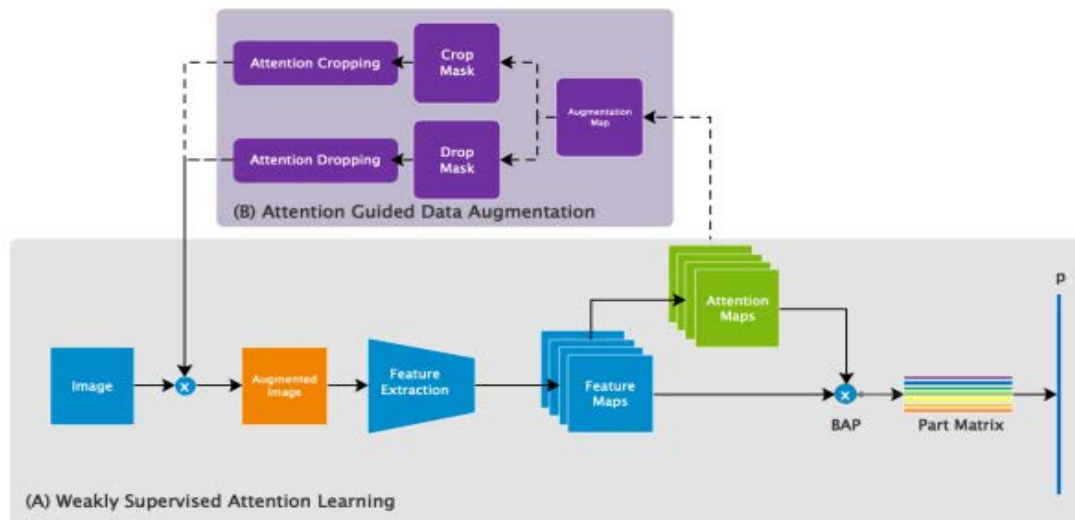


Figure 24: WSDAN Architecture

Weakly Supervised WS-DAN [23] learns to construct attention maps to extract sequential local features and reflect the spatial distribution of discriminative object components to address the fine-grained visual classification issue. Also being utilized to increase the efficiency of data augmentation is attention-guided data augmentation, which comprises attention cropping and attention dropping. Cropping and resizing one of the attention portions at random improves the portrayal of local features. Attention dropping removes one of the attention areas from the picture at random to encourage the model to extract the feature from several discriminative sections. Furthermore, attention maps are employed to properly locate and magnify the entire object, thus improving categorization accuracy.

2.2.6 MobileV2Net

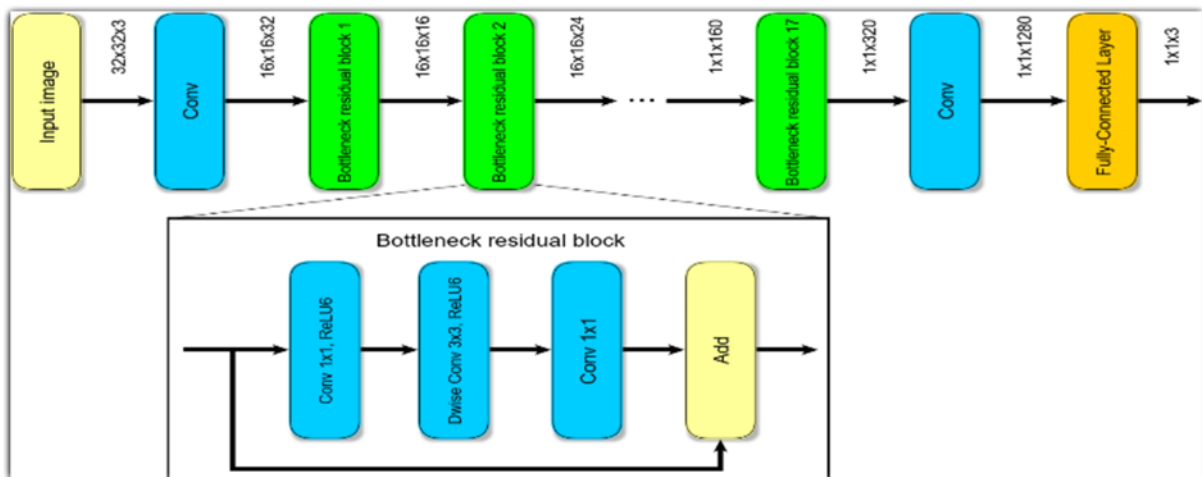


Figure 25: MobileV2 Net architecture

There are two sorts of blocks in MobileNetV2 [24]. One is a one-stride residual block. Another block for shrinking with a 2 stride. Both sorts of blocks have three levels. The first layer is 1*1 convolution using ReLU6. The depth wise convolution is the second layer. Another 1*1 convolution is used in the third layer, but this time there is no non-linearity. If ReLU is applied again, deep networks will behave like linear classifier on non-zero volume portion of result domain, according to the assertion. There is also a p expansion factor. For all main experiments, p is kept as six. The internal output would have $128 * p = 128 * 6 = 1068$ channels if the input had 128 channels. ReLU6 activation function gives 6 for input greater than 6 other than that it behaves just like RELU.

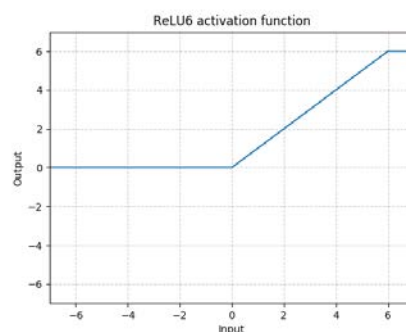


Figure 26: ReLU6 Activation Function used in MobileV2Net

2.2.7 Efficient Net

Stage	Operator	Resolution	#channels	#layers
1	Conv3x3	224x224	32	1
2	MBConv1,k3x3	112x112	16	1
3	MBConv6,k3x3	112x112	24	2
4	MBConv6,k5x5	56x56	40	2
5	MBConv6,k3x3	28x28	80	3
6	MBConv6,k5x5	14x14	112	3
7	MBConv6,k5x5	14x14	192	4
8	MBConv6,k3x3	7x7	320	1
9	Conv1x1/Pooling/FC	7x7	1, 280	1

Figure 27: Efficient Net Architecture

Any network dimension, such as width, depth, or resolution, may be scaled up to improve accuracy, but the advantage diminishes as the model gets bigger. The inventors of efficient net [25] provided a simple but effective scaling strategy that use a compound coefficient to consistently scale network width, depth, and resolution:

$$\text{Depth} = \alpha^\phi \quad \text{width} = \beta^\phi \quad \text{resolution} = \gamma^\phi$$

$$\text{such that } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$

$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

The MBConv block is simply an Inverted Residual Block with a Squeeze and Excite block thrown in for good measure (as seen in MobileNetV2).

Chapter 3

Proposed Model

After going through various research papers on deepfake detection which included some state of the art architectures which were used for feature extraction and classification purposes. Some models like Xception, Vision transformers turned out to be quite good on performance but some models used were both time and resource consuming. After comparing the accuracy, computation time needed on DFDC sample dataset efficient net comes out on top due to its low computational and highly effective detection architecture inspired by scaling mobile nets so, in our proposed model we have decided to use better version of efficient net released in 2021 which improves upon building still smaller model without compromising on accuracy. This model has been used as backbone for our model yielding promising results. This section aims to explain efficient net v2 along with architecture used in our proposed model.

3.1 EfficientNet V2

For high-quality, rapid image categorization, EfficientNets has been the state of art. They were introduced roughly two years ago and quickly gained popularity because to the way they scaled, which allowed them to learn considerably quicker than previous networks. Google launched EfficientNetV2 [29], which is a significant increase over EfficientNet in terms of training time and efficiency. In this section, we'll look at how EfficientNetV2 improves upon the prior version. Higher performing networks, EfficientNets, are built on the principle of attaining better performance with fewer parameters. When the number of parameters is reduced, several advantages emerge, such as smaller model sizes that are easier to remember. However, this frequently results in a decrease in performance. As a result, the key difficulty is to reduce the amount of parameters while maintaining performance. EfficientNetV2 employs progressive learning, which implies that, while the picture sizes are initially tiny when the training begins, they gradually expand in size. This method arises from the fact that EfficientNets' training speeds degrade as picture sizes get larger.

Progressive learning, on the other hand, is not a novel notion; it has been utilized previously. The problem is that the same regularization effect was utilized for different picture sizes when it was previously employed. According to the authors of EfficientNetV2, this reduces network capacity and performance. To address this issue, they dynamically increase the regularization along with the image sizes.

Stage	Operator	Stride	#Channels	#Layers
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3	1	24	2
2	Fused-MBConv4, k3x3	2	48	4
3	Fused-MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25	1	160	9
6	MBConv6, k3x3, SE0.25	2	256	15
7	Conv1x1 & Pooling & FC	-	1280	1

Figure 28: EfficientNetV2 Architecture

EfficientNets use a "depth wise convolution layer," which has fewer parameters and FLOPS but cannot fully exploit newer accelerators (GPU/CPU) . To address this issue, a recent research titled "MobileDets: Searching for Object Detection Architectures for Mobile Accelerators" [30] proposes a new layer called "Fused-MB Conv layer" to overcome the problem. In this case, EfficientNetV2 employs this additional layer. But, because the fused layers have a larger number of parameters, they have used NAS search to search for best combination of old MB Conv layers and the fused ones. Below graph shows the performance of efficient netv2.

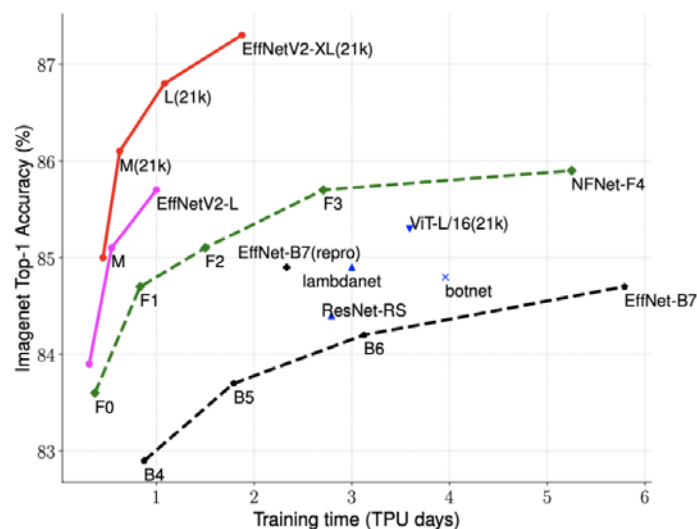


Figure 29: Performance Comparison Of EfficientNetV2 against other Architectures

3.2 Proposed model Architecture

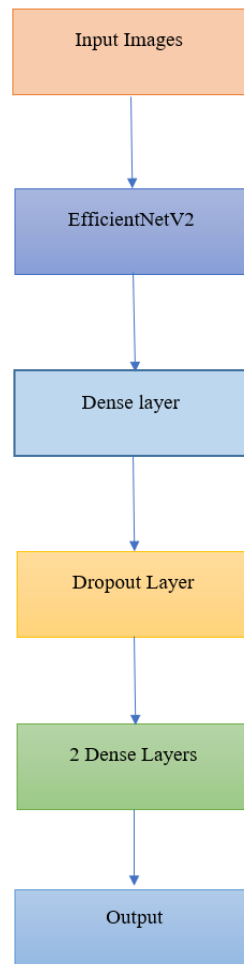


Figure 30: Proposed Model

Proposed uses facial images generated by data preprocessing of the videos as input into the EfficientNetV2 and then the output from EfficientNetV2 is passed into dense layer which is then passed into a dropout layer and then into 2 dense layers successively ReLU is used as activation function for all the layers except the last layer producing the output which uses sigmoid activation function

Chapter 4

Implementation

This Section discusses the detailed implementation of the proposed model and experiment. Starting with section 4.1 which discusses the environmental setup used to implement the project. Section 4.2 details about the dataset used for training and testing section 4.3 discusses the data preprocessing steps used to prepare the data for training and then section 4.4 finally discusses the training and validation done.

4.1 Environmental Setup

The environment used for the implementation of the experimentation and evaluation is as follows

Machine used :Laptop Acer Nitro 5

GPU/CPU used : GPU- Nvidia RTX3060 CPU – Ryzen 7 5800 series

VRAM/RAM: VRAM- 6GB RAM:16 GB

Secondary Memory:1TB

Language Used: Python 3.9

Tools Used: Visual Code

Frameworks & Libraries used: TensorFlow, Keras, MTCNN, NumPy, Matplotlib

4.2 Dataset Used

The Deepfake Detection Challenge includes the Deepfake Detection Challenge Training Set [11]The whole training set and a brief sample training set are available for download in this challenge. These videos present a true challenge with visual variety, such as gender, skin tone, age, head postures, and video background. With well over 470 GB of video, the complete dataset contains a vast collection of actual and deepfake samples. Because of the time restrictions of this investigation and the restricted computational resources available, I will only use a sample of the training set in this experiment. There are 400 videos in the limited sample training set: 323 deepfakes and 77 actual ones.

4.3 Data Preprocessing

Data preprocessing involves following steps

- 1) Converting videos to image (frames) and saving frames from each video separately in a folder with same name as the video. From each video 10 frames are stored as the video length is very short. All the frames extracted are scaled based on their size. The following picture scaling algorithms were used to account for various video quality and to maximize image processing performance:



Figure 31: Frame Captured in one of videos

Image Size(pixels)	Resize factor
<300	2x
300-1000	1x
1000-1900	0.5x
>1900	0.33x

2) Cropping Faces using face detection library MTCNN and storing them in subfolder where frames are saved.

Crop off the facial components from the frame photos so that the model can focus on attempting to capture the manipulation of the face artefacts. In circumstances multiple people appear in a particular video frame every prediction result is saved independently to provide the training dataset more variation. This GitHub repository provided the pre-trained MTCNN model: <https://github.com/ipazc/mtcnn>

30 percent margins were added to both sides of the identified face bounding box.

To capture the facial photos, the confidence level was set at 95%.

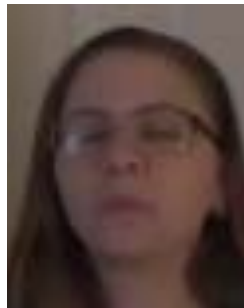


Figure 32: Face Detected and Cropped in data Preprocessing

3) Since the number of fakes is significantly greater than the number of actual faces, down sampling was done on the fake dataset depending on the number of real crops since real videos are used to create fake ones. This assisted to avoiding class imbalance problems throughout the training period. Also data has been partitioned into training, validation, and testing sets as the final stage in the data preparation procedure in the ratio of 80:10:10.

Total Number of Real faces: 978

Total Number of Fake faces: 3627

After Splitting the dataset into train ,validation, test we have 1578 images in train 198 in validation set and 204 in test set.

4.4 Training & Validation

Layer (type)	Output Shape	Param #
efficientnetv2-s	(None, 1280)	20331360
dense (Dense)	(None, 512)	655872
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 128)	65664
dense_2 (Dense)	(None, 1)	129
Total params: 21,053,025		
Trainable params: 20,899,153		
Non-trainable params: 153,872		

Figure 33: Model Architecture

The training is carried with above architecture and the backbone for the development effort is EfficientNetV2. As most deepfake films are constructed frame by frame, the deepfake detection is designed to be a binary classification problem the EfficientNet V2 model has been updated in several ways: The top input layer has been replaced by a 128x128 input with a depth of 3 and V2 final result is sent to a global max pooling layer. To flatten the output two dense layers have been added which have ReLU activation function then it is succeeded by final output layer having sigmoid as activation function .So if input is passed as fairly cropped facial image the model returns a value lying in range of 0 to 1 indicating the likelihood that image is fake(0) or real(1).

Chapter 5

Results And Evaluation

This Section gives a brief overview of the evaluation metrics used and results obtained

Although the model is trained by varying hyper parameters like total number of epochs trained different learning rate for optimizers, dropout rate, batch size but the best results are achieved using the following hyperparameters.

Total Number of epochs trained -20

Input image size – 128*128

Optimizer used – Adam

Learning rate – 0.0001

Dropout rate – 0.5

Batch size- 32

Regularization -L2 with 0.001 as rate

Since the raw data was highly imbalanced we have done down sampling to reduce the imbalance which helped us achieve higher precision and recall along with high accuracy.

Binary cross entropy loss is used to monitor validation loss. The following graph gives the comparison of training loss versus validation loss over number of epochs . Also the training accuracy and validation accuracy are given in graph below.

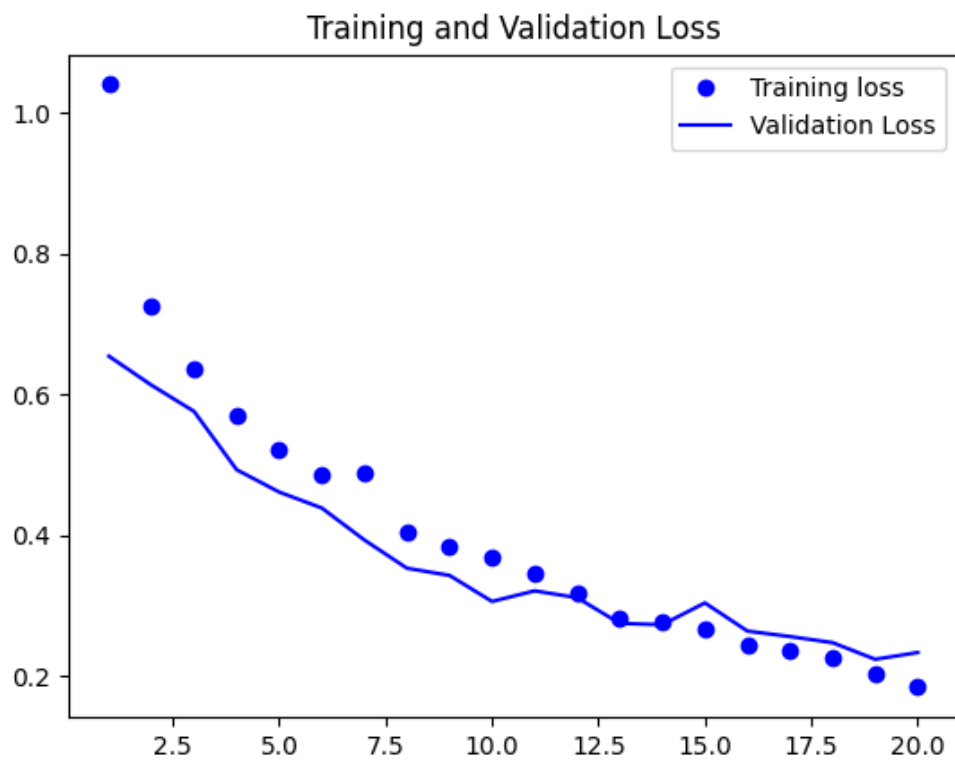


Figure 34: Training and validation Loss till 20 epochs

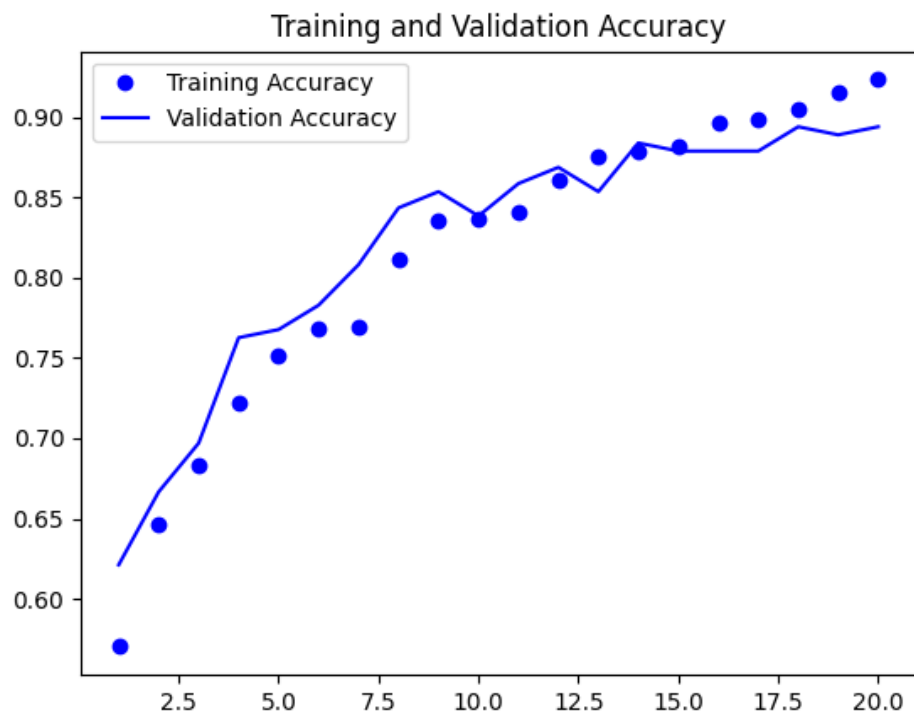


Figure 35: Training Accuracy and Validation accuracy of Proposed Model

After a machine learning model has been trained, it may be used to forecast new data by categorizing each sample into one of the predetermined classes. One technique to ensure that these predictions are credible is to divide the dataset into three subgroups for training, validation, and testing. Moreover, several assessment criteria are employed during testing to assess a model's performance quality. Accuracy, confusion matrix, precision, recall, log loss, and area under the ROC curve are some of the most frequent ones for classification tasks.

Following are some results from aforementioned evaluation metrics

Confusion matrix is taken out for the test set which consists of 204 images comes out to be

Table 3: Confusion Matrix

Confusion Matrix in Number of Images terms		Predicted Values	
		Real	Fake
True Values	Real	94	6
	Fake	14	90

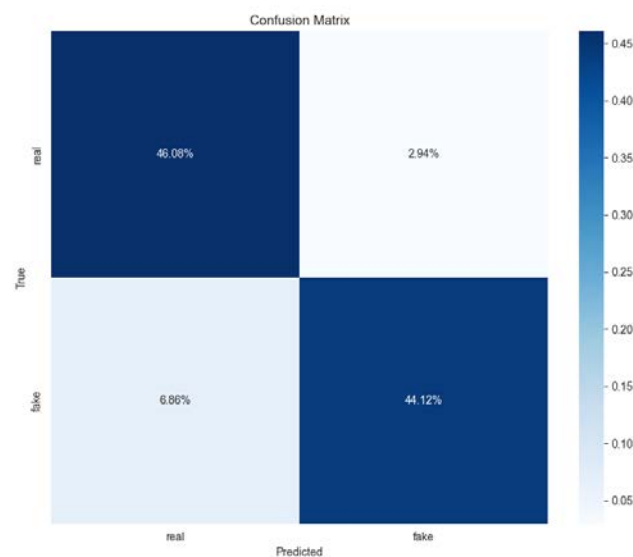


Figure 36: Confuion matrix in percentage terms

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} = \frac{94+90}{94+6+90+14} = 90.2\%$$

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{94}{94+6} = 94\%$$

$$\text{Recall} = \frac{TP}{TP+FN} = \frac{94}{94+14} = 87.03\%$$

$$\text{F1 Score} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} = 2 * \frac{0.94 * 0.87}{0.94 + 0.87} = 0.90$$

Where TP= Number of True Positives (image predicted as real is actually real)

TN=Number of True Negatives(Image predicted as deepfake is actually a deepfake)

FP=Number of False Positives (Images predicted as real but are actually fake)

FN=Number of false negative (images predicted as fake but are actually real)

Comparison of Accuracy of Various Models run on local machine

Table 4: Comparison of Various Models

Model Name	Model Test Accuracy	Dataset Used
VGG-16	77%	DFDC sample
WS-DAN + Xception	85%	DFDC sample
Efficient Net	87%	DFDC sample
Proposed model	90%	DFDC sample

Chapter 6

Conclusion

Given the tremendous significance of multimedia in daily life and online communication, being able to identify whether a media contains modified media is extremely crucial. As a result, the focus of this research was on detecting face modifications in video sequences, specifically false films created using deepfake technology. This study aims to compare various models applied for image detection and use the best performance model in terms of accuracy and minimum computational time and use it for deepfake detection. Since the model uses less computational time it can be further improved and used for mobile devices in future without compromising on accuracy.

References

- [1] Makhzani, Alireza, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey, “ "Adversarial autoencoders." ,” *arXiv preprint arXiv:1511.05644*, (2015).
- [2] Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, “ "Generative adversarial nets." ,” *Advances in neural information processing systems 27* , (2014).
- [3] Karras, Tero, Samuli Laine, and Timo Aila, “"A style-based generator architecture for generative adversarial networks." ,” *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. pp. 4401-4410. , 2019..
- [4] Nirkin, Yuval, Yosi Keller, and Tal Hassner, “ "Fsgan: Subject agnostic face swapping and reenactment." ,” *In Proceedings of the IEEE/CVF international conference on computer vision*, pp. pp. 7184-7193, 2019..
- [5] “FaceApp,” [Online]. Available: <https://medium.com/@12harsharyan/faceapp-how-neural-networks-can-do-wonders-c2b83b00da7b>.
- [6] Radford, Alec, Luke Metz, and Soumith Chintala, “ Unsupervised representation learning with deep convolutional generative adversarial networks.” *arXiv preprint arXiv:1511.06434*, (2015)..
- [7] Mirza, Mehdi, and Simon Osindero, “"Conditional generative adversarial nets." ,” *arXiv preprint arXiv:1411.1784*, (2014)..
- [8] Mao, Xudong, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley., “Least squares generative adversarial networks.” *In Proceedings of the IEEE international conference on computer vision*, pp. pp. 2794-2802, 2017..
- [9] P. Korshunov and S. Marcel, “DeepFakes: a New Threat to Face Recognition? Assessment and Detection.” *arXiv and Idiap Research Report*, 2018.
- [10] Rössler, Andreas, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner., “"Faceforensics++: Learning to detect manipulated facial images." ,” *In Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. pp. 1-11, 2019.

- [11] Dolhansky, Brian, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer, “The deepfake detection challenge (dfdc) preview dataset,” *arXiv preprint arXiv:1910.08854*, 2019.
- [12] Li, Yuezun, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu, “Celeb-df: A large-scale challenging dataset for deepfake forensics,” *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. pp. 3207-3216., 2020.
- [13] Zi, Bojia, Minghao Chang, Jingjing Chen, Xingjun Ma, and Yu-Gang Jiang, “Wilddeepfake: A challenging real-world dataset for deepfake detection.,” *In Proceedings of the 28th ACM International Conference on Multimedia*, pp. pp. 2382-2390, 2020..
- [14] Xuan, Xinsheng, Bo Peng, Wei Wang, and Jing Dong, “On the generalization of GAN image forensics.,” *In Chinese conference on biometric recognition*, , pp. pp. 134-141, 2019.
- [15] Sabir, Ekraam, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natarajan, “Recurrent convolutional strategies for face manipulation detection in videos,” *Interfaces (GUI)* 3, pp. 80-87, 2019.
- [16] Iandola, Forrest, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer “Densenet: Implementing efficient convnet descriptor pyramids,” *arXiv preprint arXiv:1404.1869*, 2014.
- [17] Güera, David, and Edward J. Delp., “Deepfake video detection using recurrent neural networks,” *In 2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS)*,, pp. 1-6. , 2018.
- [18] Simonyan, Karen, and Andrew Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [19] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Identity mappings in deep residual networks,” *In European conference on computer vision*, pp. pp. 630-645, 2016..

- [20] Afchar, Darius, Vincent Nozick, Junichi Yamagishi, and Isao Echizen., “Mesonet: a compact facial video forgery detection network,” *In 2018 IEEE international workshop on information forensics and security (WIFS)*, pp. pp. 1-7, 2018.
- [21] Chollet, Francois, “Xception: Deep Learning with Depthwise Separable Convolutions,” *In IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [22] Khormali, Aminollah, and Jiann-Shiun Yuan, “DFDT: An End-to-End DeepFake Detection Framework Using Vision Transformer.,” *Applied Sciences 12*, no. 6, 2022.
- [23] Hu, Tao, Honggang Qi, Qingming Huang, and Yan Lu, “See better before looking closer: Weakly supervised data augmentation network for fine-grained visual classification.,” *arXiv preprint arXiv:1901.09891* , 2019.
- [24] Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen, “Mobilenetv2: Inverted residuals and linear bottleneck,” *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. pp. 4510-4520, 2018.
- [25] Tan, Mingxing, and Quoc Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” *In International conference on machine learning*, pp. pp. 6105-6114, 2019.
- [26] Khan, Sohail Ahmed, and Hang Da, “Video Transformer for Deepfake Detection with Incremental Learning.,” *In Proceedings of the 29th ACM International Conference on Multimedia* , pp. pp. 1821-1828, 2021..
- [27] Wodajo, Deressa, and Solomon Atnafu, “Deepfake video detection using convolutional vision transformer,” *arXiv preprint arXiv:2102.11126*, 2021.
- [28] Heo, Young-Jin, Young-Ju Choi, Young-Woon Lee, and Byung-Gyu Kim, ““Deepfake detection scheme based on vision transformer and distillation.,” *arXiv preprint arXiv:2104.01353* , 2021.
- [29] Tan, Mingxing, and Quoc Le, “Efficientnetv2: Smaller models and faster training,” *In International Conference on Machine Learning, PMLR* , pp. pp. 10096-10106., 2021..
- [30] Xiong, Yunyang, Hanxiao Liu, Suyog Gupta, Berkin Akin, Gabriel Bender, Yongzhe Wang, Pieter-Jan Kindermans, Mingxing Tan, Vikas Singh, and Bo Chen. “Mobiledets:

Searching for object detection architectures for mobile accelerators.,” *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*,, pp. pp. 3825-3834., 2021.

PAPER NAME

2K20_ISY_15_thesis(revised).pdf

WORD COUNT

6766 Words

CHARACTER COUNT

36632 Characters

PAGE COUNT

43 Pages

FILE SIZE

1.7MB

SUBMISSION DATE

May 24, 2022 8:18 PM GMT+5:30

REPORT DATE

May 24, 2022 8:19 PM GMT+5:30


● 16% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 6% Internet database
- 1% Publications database
- Crossref Posted Content database
- 14% Submitted Works database

● Excluded from Similarity Report

- Crossref database
- Bibliographic material