

# Deep Fake Detection Using Hybrid Deep Learning Architecture

A Project Report Submitted in Partial Fulfillment of the Requirements for Award  
of  
the Degree of Bachelor of Technology in Information and Communication Technology

Submitted by  
**Ishan Mistry** 18BIT033  
**Alister Rodrigues** 18BIT041  
**Khush Joshi** 18BIT056  
**Manali Shah** 18BIT060  
**Mansi Raveshia** 18BIT063

Under the Supervision and Guidance of  
**Dr. Mohendra Roy**  
Assistant Professor  
Pandit Deendayal Energy University

Submitted to  
Department of Information and Communication Technology  
School of Technology  
Pandit Deendayal Energy University (PDEU)  
Gandhinagar, INDIA, 382007

# Declaration

---

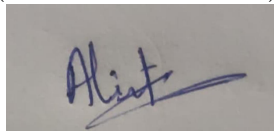
I hereby declare that the project work entitled “**Deep Fake Detection Using Hybrid Deep Learning Architecture**” is an authentic record of my own work carried out in Pandit Deendayal Energy University research institute as requirement of B. Tech dissertation for the award of **Bachelor of Technology in Information and Communication Technology**. I have duly acknowledged all the sources from which the ideas and extracts have been taken. The project is free from any plagiarism and has not been submitted elsewhere for any degree, diploma and certificate.

Signature: .....



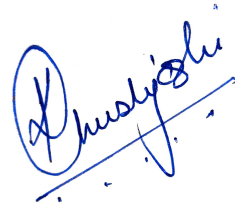
Ishan Mistry  
(Roll No. 18BIT033)

Signature: .....



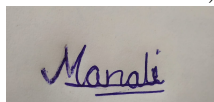
Alister Rodrigues  
(Roll No. 18BIT041)

Signature: .....



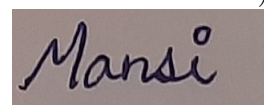
Khush Joshi  
(Roll No. 18BIT056)

Signature: .....



Manali Shah  
(Roll No. 18BIT060)

Signature: .....



Mansi Raveshia  
(Roll No. 18BIT063)

Department of Information and Communication Technology  
**PDEU**

Dr. Ganga Prasad Pandey  
H. O. D.

Phone: +91-79-2327 5397  
e-mail: mehul.raval@sot.pdpu.ac.in

---

## Certificate

This is to certify that the project entitled “**Deep Fake Detection Using Hybrid Deep Learning Architecture**” submitted by **Ishan Mistry**(18BIT033) , **Mansi Raveshia**(18BIT063) , **Manali Shah**(18BIT060) , **Khush Joshi**(18BIT056), **Alistar Rodrigues**(18BIT041) to the Department of Information and Communication Technology under School of Technology, PDEU in partial fulfillment of the requirements for award of the degree of **Bachelor of Technology in Information and Communication Technology** embodies work carried out under the guidance and supervision of Dr.Mohendra Roy, Assistant Professor, Pandit Deendayal Energy University.

.....  
Dr. Ganga Prasad Pandey  
(HOD, I.C.T. Department, PDEU.)

## Pandit Deendayal Energy University

Raisan, Gandhinagar

DR.Mohendra Roy  
Assistant Professor

Phone: 9435980357  
e-mail:  
mohendra.Roy@sot.pdpu.ac.in

---

### Certificate

This is to certify that the project entitled “**Deep Fake Detection Using Hybrid Deep Learning Architecture**” submitted to the Department of Information and Communication Technology under School of Technology, Pandit Deendayal Petroleum University in partial fulfillment of the requirements for award of the degree of **Bachelor of Technology in Information and Communication Technology** is a record of work carried out by **Is-han Mistry**(18BIT033) , **Mansi Raveshia**(18BIT063) , **Manali Shah**(18BIT060) , **Khush Joshi**(18BIT056), **Alister Rodrigues**(18BIT041) under my supervision and guidance in the ”Pandit Deendayal Energy University”, ”Gandhinagar”.

Signature .....

Dr. Mohendra Roy  
(Assistant Professor)  
9435980357  
mohendra.Roy@sot.pdpu.ac.in

# Certificate of Approval

---

The forgoing project entitled “**Deep Fake Detection Using Hybrid Deep Learning Architecture**” submitted by **Ishan Mistry**(18BIT033) , **Mansi Raveshia**(18BIT063) , **Manali Shah**(18BIT060) , **Khush Joshi**(18BIT056), **Alistar Rodrigues**(18BIT041) to the Information and Communication Technology under School of Technology, PDPU is hereby approved as project work carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite of **Bachelor of Technology in Information and Communication Technology** degree for which it has been submitted. It has been understood that by this approval of the undersigned do not necessarily endorse or approve every statement made, opinion expressed or conclusion drawn therein but approve only for the purpose for which it is being submitted.

.....

Examiner

## Acknowledgement

The success and final outcome of this project required a lot of guidance and assistance from many people. It is our privilege to pledge the following few lines of dedication to those who helped us directly or indirectly in completing my project.

This work would not have been possible without the constant support, guidance, and assistance of our advisor Dr. Mohendra Roy. His levels of punctuality, knowledge, and ingenuity is something I will always keep aspiring to.

Furthermore, I am thankful to and fortunate enough to the faculties of ICT Department for all the knowledge they have imparted, which I could put to use in this project in numerous contexts, not only limited to the concepts of Artificial Intelligence and Machine Learning but also including the not so forgotten concepts ranging from Calculus and Linear Algebra to the Signal Processing Concepts

Most importantly, last but not the least, I appreciate the co-operation and benevolence of my friends for helping me with the data recording and gathering all the insights from the 30 papers that we have done so far, it would have been a tedious journey to begin with, if it had not been the team members working together.



# Contents

<b>Contents</b>	<b>vi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem statement . . . . .	1
1.2 Motivation . . . . .	1
1.3 Literature review . . . . .	2
1.3.1 DeepFake Generation Techniques . . . . .	2
1.3.2 DeepFake Detection Techniques - Deep Learning Based . . . . .	5
1.3.3 DeepFake Detection Techniques - Conventional Method . . . . .	7
1.3.4 DeepFake Evasion Techniques . . . . .	10
1.4 Datasets . . . . .	10
1.5 Metrics and Functions Encountered . . . . .	10
1.5.1 Metrics . . . . .	11
1.5.2 Loss Functions . . . . .	12
1.6 Exploratory Data Analysis . . . . .	13
1.7 Image Processing Techniques . . . . .	17
1.8 Models Implemented . . . . .	19
1.8.1 MesoNet . . . . .	19
1.8.2 CNN-1 (Trial) . . . . .	20
1.8.3 DenseNet and DenseNet with GrayScale . . . . .	21
1.8.4 Siamese Network . . . . .	23
1.8.5 Deep Convolutional Neural Network Model . . . . .	26
<b>2 Timeline - Workflow</b>	<b>31</b>
2.1 Research Phase - Accomplished . . . . .	31
2.1.1 Week 1 - Logistics . . . . .	31
2.1.2 Week 2 - Survey Papers . . . . .	31
2.1.3 Week 3 - A Deeper Dive . . . . .	31
2.1.4 Week 4 - Conventional : Non DL based Methods . . . . .	31
2.1.5 Week 5 - Recapitulation . . . . .	31
2.1.6 Week 6 - Data Gathering . . . . .	32
2.2 Execution Phase . . . . .	35
2.2.1 Week 7 - Dataset Exploration and Preparation . . . . .	35
2.2.2 Week 8 - Preprocessing of Data . . . . .	35
2.2.3 Week 9 - Designing Model Architecture . . . . .	35
2.2.4 Week 10 - Training Workflow . . . . .	35
2.2.5 Week 11 - Prediction Workflow . . . . .	35
2.2.6 Week 12,13 - Buffer Weeks . . . . .	35



**3 Conclusion and Observations** **37**

3.1 Observations from papers . . . . . 37

3.2 Experimental Results . . . . . 38

3.3 Future Aspects . . . . . 38

# Chapter 1

## Introduction

### 1.1 Problem statement

Detection of Deep Fakes using a hybrid neural network architecture : The Deep Fakes out there need to be kept in check and for that very reason we need to elevate our detection techniques too, in order to tackle them. This project requires studying the various implemented detection schemes and use the insights to devise a new and possibly better hybrid architecture for the detection of Deep Fakes.

### 1.2 Motivation

People have a tendency to readily believe what they see, quite often overlooking the credibility of the source of the video/image/text. Although people are getting more and more vigilant these days, becoming skeptical towards blindly trusting the media, but there is a caveat to this self established trust. The situation above is nothing but a subtle case of misinformation.

Humans are believed to have six degrees of separation between you and your farthest friend. "Six degrees of separation" is the idea that all people on average are six, or fewer, social connections away from each other. As a result, a chain of "friend of a friend" statements can be made to connect any two people in a maximum of six steps. Moreover the situation has been exacerbated by the various channels of media/communication that are consumed these days - News groups/conspiracy groups/Social Media. There is no one to substantiate the validity of these seemingly true sources, as it is a result of the mere misinformation that has been made really easy by the current forms of media.

At first glance, these matters might seem trivial and even not alarming at all, because in most cases the impact of these is not directly visible. But there are times when things could go down-hill and have serious repercussions. For example, at times such information could be used to belittle someone or could be a philippic, aimed to damage the reputation of someone in power. This when combined with the current power of the spreading information can be treacherous.

These subtle but misinformed texts or images are not easy to track and the process could be cumbersome as this spurious information is gleaned into everyday lives to a point that they come out as obvious and evidently become the truth. The facts and figures presented/spread could still be validated with the human expertise in the specific domains - Politics/Sports/Daily Sciences etc.[2] But with advances in Deep Learning, deep fakes have become so realistic that it is really tough for us to visibly differentiate a true video from a deep fake video[7]. This gives rise to the need of having accurate detection techniques for their detection and hence their removal from the web. The bad people keep getting better and better and so shall we, in order

to counter them.

## 1.3 Literature review

### 1.3.1 DeepFake Generation Techniques

Methods based on the neural image style transfer becomes the tool for creating the deepfakes videos. Addition to this there are now several open source softwares like FakeApp [8], DFaker[8], Faceswap-GAN[8], faceswap[8] and DeepFaceLab[8] can generate the deepfakes. Some DeepFakes pose more serious threats like Identity Swap, whereas Expression swap and attribute manipulation are on the milder side of the spectrum[6]. Entire face synthesis is a low risk strategy because no soft or hard bio metrics are changed in the process. The DeepFake Generation has been classified into 4 major categories :

1. Entire Face Syntheses
2. Attribute Manipulation
3. Identity Swap
4. Expression Swap
5. Other Methods

### Entire Face Syntheses

These types of generation techniques take Input as Random Vectors and produce as Output - High Quality non-existent faces. Target images are not needed (latent vector is manipulated for results). The following architectures have been used typically for the Entire face synthesis.[6][10]

#### 1. DCGAN

First work that does CNN+GAN[14][4]. Focuses on unsupervised learning - had problems like balancing discriminator and generator

#### 2. WGAN

This took care of the balance and provided reasonable and efficient approximation of the EM distance. WGAN uses weight clipping to enforce a Lipschitz constraint. To improve the weight clipping operation, they have proposed to penalize the norm of the gradient of the discriminator with respect to its input fake image. The new designs train stably when generating high-quality home images.

#### 3. CrammerGAN

Simply using Wasserstein probability can not simultaneously satisfy **sum invariance**, **scale sensitivity**, and **unbiased sample gradients**. This is what brought the Crammer-GANS into the limelight. It combined the best of the Wasserstein and Kullback-Leibler divergences to propose the Cramér distance.

#### 4. PGGGAN

The focus is on high resolution images. The images initially start from a low resolution and are being detailed in a step by step fashion with the new layers being added to the model. This method is very reasonable in that it can **speed up the training** as well as greatly **stabilize the GAN**.

## 5. BigGAN

The main focus is to generate high resolution diverse images. They have applied orthogonal regularization to enforce the generator to be satisfied with a simple “truncation trick”. Thus, the user can control the trade-off between image fidelity<sup>1</sup> and variety by reducing the variance of the generator’s input.

## 6. StyleGAN

Automatically learns the unsupervised separation of high-level attributes such as pose and human identity. The architecture also leads to stochastic variation in the generated images (e.g., freckles, hair). Furthermore, it enables intuitive, scale-specific control of the synthesis. They have encouraged good conditioning in the mapping from latent codes to images by the new design of generator normalization, progressive growing, and generator regularization.

## 7. Glow

A flow-based generative model that uses an invertible  $1 \times 1$  convolution. The method is based on the theory that a generative model optimized towards the plain log-likelihood objective has the ability to generate efficient realistic-looking synthesis and manipulate large images.

# Attribute Manipulation

Similar to the previous techniques, these also don’t necessarily require Target images(latent vector manipulated for results). It is known as face editing, which cannot only modify simple face attributes such as hair color, bald,smile, but also retouch complex attributes like gender, age, etc. Examples of such architectures used are as follows [13] [6] :

## 1. IcGAN

This can be seen as an extension of cGANs(Conditional GANs) . They have evaluated encoders to map a real image into a latent spaceand a conditional representation, which allows the reconstruction and modification of arbitrary attributes of real human face images.

## 2. StarGAN

As previous studies only did image to image translation only for 2 domains - which is cumbersome and time consuming, StarGAN was devised to perform **translation to multiple domains**. It allows simultaneous training of multiple different-domain datasets within a single network.

## 3. StarGAN2

Maintains diversity of generated images as well as scalability over multiple domains as obtained in StarGAN. They replaced StarGAN’s domain label with their domain-specific style code. To adapt the style code, they have proposed two modules: a **mapping network**(transforms random noise into style codes) and a **style encoder**.

## 4. GANimation

---

<sup>1</sup>The ability to discriminate between two images or how closely the image represents the real source distribution

StarGAN had the limitation of the content of the datasets, it can only generate a discrete number of expressions. To address this we have a novel GAN conditioning method based on **action units (AU) annotations**. It defines the human expression with a continuous manifold of the anatomical facial movements. The magnitude of activation of each AU can be controlled independently. Different AUs can also be combined with each other with this method.

## 5. AttGAN

Previous methods have attempted to establish an attribute independent latent representation for further attribute editing. However, since the facial attributes are relevant, requesting for the invariance of the latent representation to the attributes is excessive. Therefore, simply forcing the attribute-independent constraint on the latent representation not only restricts its representation ability but also may result in information loss, which is harmful to the attribute editing. To solve this problem, **AttGAN** has **removed the strict attribute independent constraint from the latent representation**. It just applies the attribute classification constraint to the generated image to guarantee the correctness of attribute manipulation. Meanwhile, it groups attribute classification constraint, reconstruction learning, and adversarial learning together for high-quality facial attribute editing.

## 6. STGAN

Improvement of AttGAN has selectively taken the difference between target and source attribute vectors as the input of the model. They have enhanced attribute editing by adding a **selective transfer unit that can adaptively select and modifying the encoder feature to the encoder-decoder**.

## Identity Swap

This function is able to replace the face in the target image with the face in the source image. The task is to replace the face of one person in a video with the face of another person. Two approaches have been extensively considered in the domain. First being the classical computer graphics-based techniques such as FaceSwap and secondly novel deep learning techniques known as DeepFake.

## Expression Swap

Expression swap is similar to identity swap. It is able to replace the facial expression in the target image with the facial expression in the source image. It is also known as face reenactment. In our investigation, only Face2Face and A2V were attempted by the surveyed DeepFake detection methods.

## Other Methods

Some of the other methods for generating Deep Fakes are as follows :

### Style Transfer

1. **GatedGAN** - GatedGAN uses gated networks to transfer multiple styles in a single model. They have added a gated transformer into the encoder-decoder

## 2. AAMS -

AAMS has developed an attention-aware multistroke style transfer model. They have enabled using different brush strokes to render the diverse levels of detail. They also have coordinated spatial distribution of visual attention between the content image and stylized image.

### In-painting

**1. ContextAtten** The new architecture proposed by them can synthesize novel image structures as well as explicitly utilize surrounding image features as references to make better predictions.

**2. SC-FEGAN** A novel image editing system that generates images with the free-form masks, sketches, and color provided by the users.

### Rendering

**1. CRN** CRN is basically a single feed-forward network, trained end-to-end with a direct regression objective. It has proposed a rendering network to produce a photographic image with a two-dimensional semantic specification of the scene.

**2. GauGAN** GauGAN has proposed a simple yet effective layer for synthesizing photo-realistic images with an input semantic layout. They have proposed to use a spatially-adaptive, learned transformation to modulate the activation in normalization layers with the input layout.

### Detection evasive

**1. SDGAN** - SDGAN has proposed using a spectral discriminator to simulate the frequency distribution of the real data when generating images.

**2. WUCGAN** - WUCGAN has shown common **up-sampling methods** are causing the **inability** of GANs to reproduce spectral distributions of real images correctly. To overcome this drawback, they have proposed to add a spectral regularization term to the training optimization objective.

## 1.3.2 DeepFake Detection Techniques - Deep Learning Based

1. The first category of DeepFake detection methods are data-driven, which directly employ various types of DNNs trained on real and DeepFake videos, not relying on any specific artifact.
2. The second category of DeepFake detection algorithms use signal level artifacts introduced during the synthesis process such as those described in the Introduction.
3. Third category is based on inconsistencies exhibited by the physical/physiological aspects in the DeepFake videos

## Spatial Based

### 1. Image Forensics based detection

Differences between synthesised faces and real faces are revealed in the chrominance components, especially in the residual domain. The main idea was proposing to train a one-class classifier on real faces by leveraging the differences in the chrominance components for tackling the unseen GANs. However, performance against perturbation attacks like image transformations is unknown. Similarly in tackling fake videos, researchers borrowed ideas from traditional video forensic by leveraging the local motion features captured from real videos to spot the abnormality of manipulated videos.

### 2. DNN-based detection

Completely data driven by utilising DNN models by extracting spatial features to improve effectiveness and generalization detection. It was found to be severely weak against adversarial attacks with additive noises. Existing studies to leverage DNN to identify deep fakes are categorised by

- (a) Improving generalisation abilities
- (b) Investigating artifact clues
- (c) Empowering CNN models

### 3. Obvious artifact clues

Generated deepfakes exhibit obvious artifacts due to limitations in AI and can be leveraged. A full convolutional approach is applied for training classifiers. Two vectors from the aforementioned two networks are compared for detecting the identity-to-identity discrepancies. This approach also has a good generalization ability across GANs.

### 4. Detection and localisation

By locating manipulated regions that provide evidence for forensics, it was found that the imperfection of upsampling methods exhibits obvious clues for detection and forgery localization where the manipulated area could be precisely marked.[10]

### 5. Facial image preprocessing

Some studies propose preprocessing the facial images before sending them to binary classifiers for discrimination. Layer-by-layer neuron behaviors provide more subtle features for capturing the differences between real and fake faces. This provides a new insight for spotting fake faces by monitoring third-party DNN-based neuron behaviors, which could be extended to other fields like fake speech detection.

## Frequency based

### 1. GAN-based artifacts

Investigating imperfect designs of existing GANs which provides obvious signals, it was observed that the internal value of the generator is normalized which limits the frequency of saturated pixels. Then, a simple SVM-based classifier is trained to measure the frequency of saturated and under-exposed pixels in each facial image for discriminating fake faces.[9]

## 2. Frequency Domain

Severe artifacts introduced due to the upsampling techniques in GANs, to which a classifier with a simple linear model and a CNN based model can achieve promising results on the entire frequency spectrum.

## Biological Signal based

[11]

### 1. Visual-Audio inconsistency

Specific words given that involve lip touching is found inconsistent in fake videos. Lip sync inconsistencies are strong but not solid evidence towards deep fakes.

### 2. Visual Inconsistency

Indicates that synthesized faces are inconsistent and unnatural. In noticing general behavioural patterns in humans and deepfakes, a lot of artifacts can be found.

### 3. Biological Signals in video

Biological signals like heartbeat rhythms and monitoring blood flow to observe subtle color changes in the skin.

## Other Detectors

**Distributed ledger technologies (DLT)** - Leveraging distributed ledger technologies (DLT) to combat digital deception, user behavior clues like the eye-gaze for DeepFake detection. Finding the artifacts which exist in the specific facial region could improve the detection performance by a large margin than the entire face.

### 1.3.3 DeepFake Detection Techniques - Conventional Method

#### 1. Steganalysis feature

The steganalysis feature (low level features) is a local descriptor based on cooccurrence statistics of nearby pixel noise residuals obtained from multiple linear and non-linear filters. First attempt was made for the above technique by modelling it as a **Gaussian Model**, [15] which was later improved by treating the task as **anomaly detection** which used a **discriminative learning autoencoder outlier removing method based on steganalysis features**

Noteworthy for CNNs : **The performance degrades significantly when multiple post processing techniques are applied to tampered regions.**

For steganalysis they have a Deep CNN that is based on **triple loss** - in order to ensure that two parts of the image from the same image are closer in the learned embedding whereas the difference between those from different images are large.

This is implemented by simply constraining the distances between the features of the patches by some margin  $m$ .



Formally, given Image patch  $x_a$  (anchor patch), a patch  $x_p$  (positive patch) from the same image, and  $x_n$  (negative patch) from a different image.

$$d(f(r(x_a)), f(r(x_p))) + m < d(f(r(x_a)), f(r(x_n)))$$

where  $r(x)$  is the steganalysis features of patch  $x$ ,  $f(r(x))$  - is the embedding we want to learn of  $x$ , and  $d()$  is the sum of squares distance measure.

Loss function is hence :

$$L(f) = \sum_{a,p,n} \max(0, m + d(f_a, f_p) - d(f_a, f_n))$$

In laymen terms, the triplet network tells if the two patches under study are from the same image or not .This means if the patches are from the same image then they are closer in distance based on the learned embedding and if they are farther then they come from a different image - hence tampered.

Combined metric : Steganalysis + SVM

$$F(q) + \lambda \frac{1}{N_q} \sum_{x \in q} S(x)$$

where  $N_q$  is the number of patches inside face  $q$ .

## 2. CFA pattern

This method estimates the CFA pattern and uses a GMM algorithm to classify the variance of prediction error using the estimated CFA pattern. The output of this method is a local level tampering probability map. For the face region, an average probability is calculated as the final score.

## 3. Improved DCT Coefficient

This method estimates the DCT coefficients for all the regions in the given image to find the singly JPEG compressed regions and classifies them as tampered regions. The output of this method is a probability map indicating tampering.

4. **Spatial-Temporal Features** We extract spatial-temporal features by detecting space-time interest points in videos. we calculate the response function by application of separable linear filters.[11] Assuming a stationary camera or a process that can account for camera motion, the response function has the form:

$$R = (I \cdot g \cdot h_{ev})^2 + (I \cdot g \cdot h_{od})^2$$

where  $I(x, y, t)$  denotes images in the video,  $g(x, y; \sigma)$  is the 2D Gaussian smoothing kernel, applied only along the spatial dimensions  $(x, y)$ , and  $h_{ev}$  and  $h_{od}$  are a quadrature pair of 1D Gabor filters applied temporally, which are defined as

$$h_{ev}(t; \tau, \omega) = -\cos(2\pi t\omega) e^{\frac{-t^2}{\tau^2}}$$

$$h_{od}(t; \tau, \omega) = -\sin(2\pi t\omega) e^{\frac{-t^2}{\tau^2}}$$

In all cases we use  $\omega = \frac{4}{\tau}$ . The two parameters  $\sigma$  and  $\tau$  correspond roughly to the spatial and temporal scales of the detector. Each interest point is extracted as a local

maxima of the response function. Any region with spatially distinguishing characteristics undergoing a complex motion can induce a strong response, while region undergoing pure translational motion, or areas without spatially distinguishing features, will not induce a strong response.

- At each detected interest point, a cuboid is extracted which contains the spatio-temporally windowed pixel values. The side length of cuboids is set as approximately six times the scales along each dimension, so containing most of the volume of data that contribute to the response function at each interest point. After extracting the cuboids, the original video is discarded, which is represented as a collection of the cuboids. To compare two cuboids, different descriptors for cuboids have been evaluated, including normalized pixel values, brightness gradient and windowed optical flow, followed by a conversion into a vector by flattening, global histogramming, and local histogramming. As suggested, we adopt the flattened brightness gradient as the cuboid descriptor. To reduce the dimensionality, the descriptor is projected to a lower dimensional PCA space. By clustering a large number of cuboids extracted from the training data using the K-Means algorithm, we derive a library of cuboid prototypes. So each cuboid is assigned a type by mapping it to the closest prototype vector. we use the histogram of the cuboid types to describe the video.

## 5. Recognition : SVM -

The Support Vector Machine (SVM) classifier to recognize affective body gestures. For the cases where it is difficult to estimate the density model in high-dimensional space, the discriminant approach is preferable to the generative approach.

Given a training set of labeled examples  $(x_i, y_i), i = 1, \dots, l$  where  $x_i \in \mathbb{R}^n$  and  $y_i \in \{1, -1\}$ , a new test example

$$f(x) = \text{sgn}(\sum_{i=1}^l y_i K(x_i, x) + b)$$

where  $\alpha_i$  are Lagrange multipliers of a dual optimization problem that describe the separating hyperplane,  $K(\cdot, \cdot)$  is a kernel function, and  $b$  is the threshold parameter of the hyperplane. The training sample  $x_i$  with  $\alpha_i > 0$  is called the support vector, and SVM finds the hyperplane that maximizes the distance between the support vectors and the hyperplane. Given a non-linear mapping  $\phi$  that embeds the input data into the high dimensional space, kernels have the form of

$$K(x_i, x_j) = h \cdot \phi(x_i) \cdot \phi(x_j) \cdot i.$$

Advantage of using SVM here is that it allows domain-specific selection of the kernel function, and the most commonly used kernel functions are the linear, polynomial, and Radial Basis Function (RBF) kernels.

## 6. Canonical Correlation Analysis for Fusing Facial and Body Gestures :

CCA is a statistical technique developed for measuring linear relationships between two multidimensional variables. CCA to find corresponding points in stereo images was applied. CCA to model the relation between an object's poses with raw brightness images for appearance-based 3D pose estimation was suggested. A method using CCA to learn a semantic representation to web images and their associated text was proposed. Given two zero-mean random variables  $x \in \mathbb{R}^m$  and  $y \in \mathbb{R}^n$ , CCA finds pairs of directions  $w_x$  and  $w_y$  that maximize the correlation between the projections  $x$  and  $y$  are called canonical variates.

$$\rho = \frac{E[xy]}{\sqrt{E[x^2]E[y^2]}} = \frac{E[W_x^T xy^T W_y^T]}{\sqrt{E[W_x^T xy^T W_x^T]E[W_x^T yy^T W_y^T]}} = \frac{W_x^T C_{xy} W_y}{\sqrt{W_x^T C_{xx} W_x W_y^T C_{yy} W_y}}$$

where  $C_{xx} \in \mathbb{R}^{m \times m}$  and  $C_{yy} \in \mathbb{R}^{n \times n}$  are the within-set co-variance matrices of  $x$  and  $y$  respectively and  $C_{xy} \in \mathbb{R}^{m \times n}$  denotes their between-sets co-variance matrix.

### 1.3.4 DeepFake Evasion Techniques

With the fast improvement of DeepFake finders, specialists begin focusing on plan techniques to dodge the phony faces being distinguished.

In particular, given a genuine or phony face, avoidance strategies map it to another one that can't be accurately arranged by the cutting edge DeepFake identifiers, stowing away the phony appearances from being found.[7]

We can generally separate all techniques into three kinds:-

1. The first type is based on the adversarial attack.
2. The second type of methods focus on removing the fake traces in the frequency domain. These methods mainly focus on the mismatching between real and fake faces in the frequency domain while neglecting other potential factors that may make fake faces be identified easily.
3. The third kind of methods regard evasion as a general image generation process and use advanced image filtering or generative models to mislead Deep-Fake detectors.

## 1.4 Datasets

Datasets Encountered	
Real Datasets	Fake Datasets
CASIA-WebFace	UADFV
CelebA	DeepFake-TIMIT
VGGFace2	DFDCPreview
FFHQ	FaceForensics++(FF++)
VGGFace	MFCCeleb-DF
Ms-Celeb-1M	FakeCatcher
MegaFace	DFFD
LSUN	GoogleDFD
	DeeperForensics

## 1.5 Metrics and Functions Encountered

We have compiled a list of all the unfamiliar Loss Functions and evaluation metrics from all the papers that we have read so far. The list is as follows :

### 1.5.1 Metrics

1. **ACC** - Non conditional Accuracy. ACC defines overall accuracy as the probability of correspondence between a positive decision and true condition (i.e., the proportion of correct classification decisions or of decor cases).

2. **EER(Equal Error Rate)** -

When the rates are equal, the common value is referred to as the equal error rate. The value indicates that the proportion of false acceptances is equal to the proportion of false rejections. The lower the equal error rate value, the higher the accuracy of the biometric system.

- (a) Equal Error Rate - Implementation tips
- (b) Usage in facial recognition
- (c) Intuition

3. **DCT Coefficient(Discrete Cosine Transform)** -

DCT coefficient (0,0) is the DC coefficient, or average sample value. Since natural images tend to vary only slightly from sample to sample, low frequency coefficients are typically larger values and high frequency coefficients are typically smaller values.

- (a) DCT - coding and other techniques
- (b) Math behind the Transform

4. **CAM(Class Activation Maps)** -

Class Activation Maps (CAM) is a powerful technique used in Computer Vision for classification tasks. It allows the scientist to inspect the image to be categorized and understand which parts/pixels of that image have contributed more to the final output of the model.

- (a) Good Introduction
- (b) Pytorch implementation
- (c) Intro + Pytorch based

5. **CFA(The Bayer Filter)**

A Bayer filter mosaic is a color filter array (CFA) for arranging RGB color filters on a square grid of photosensors. Its particular arrangement of color filters is used in most single-chip digital image sensors used in digital cameras, camcorders, and scanners to create a color image. - Good Intuition

6. **ACPR** - APCER measures the proportion of attacks, deep fakes in this case, that are incorrectly classified as bonafide or original videos. The counterpart metric is bona fide presentation classification error rate (BPCER), which measures the proportion of incorrectly classified original videos.

## 1.5.2 Loss Functions

### 1. Triplet loss :

Triplet loss is a loss function for machine learning algorithms where a baseline (anchor) input is compared to a positive (truthy) input and a negative (falsy) input. ... This can be avoided by posing the problem as a similarity learning problem instead of a classification problem.

- (a) Intuition
- (b) Pytorch implementation hints

### 2. Geodesic distances :

A simple measure of the distance between two vertices in a graph is the shortest path between the vertices. Formally, the geodesic distance between two vertices is the length in terms of the number of edges of the shortest path between the vertices.

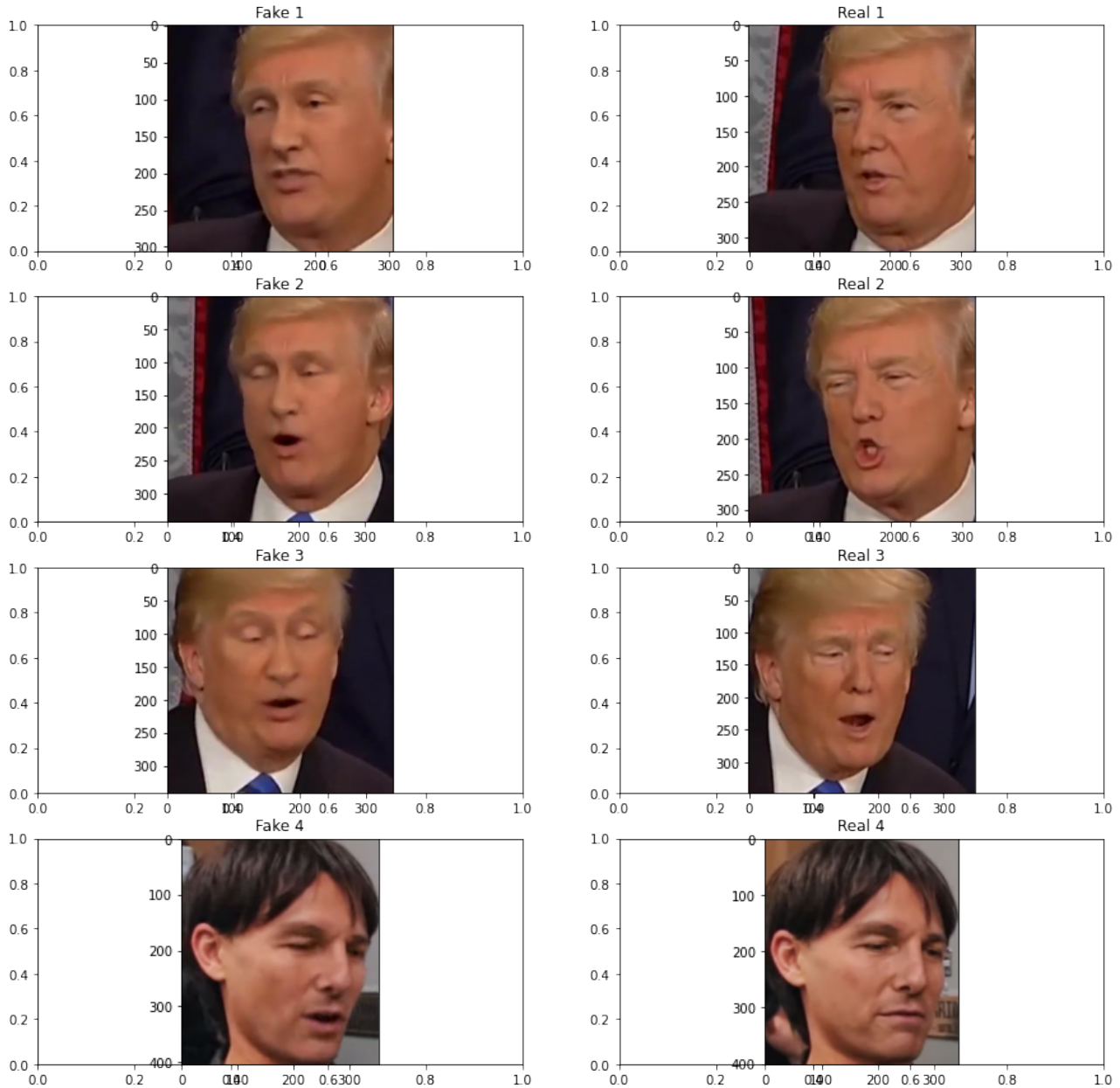
- (a) Margin Loss: insert a geodesic distance margin between the sample and centers.
- (b) Intra-Loss: decrease the geodesic distance between the sample and the corresponding center.
- (c) Inter-Loss: increase the geodesic distance between different centers.
- (d) Triplet-Loss: Insert a geodesic distance margin between triplet samples

### 3. Commonly used Loss Functions : Brief Article

## 1.6 Exploratory Data Analysis

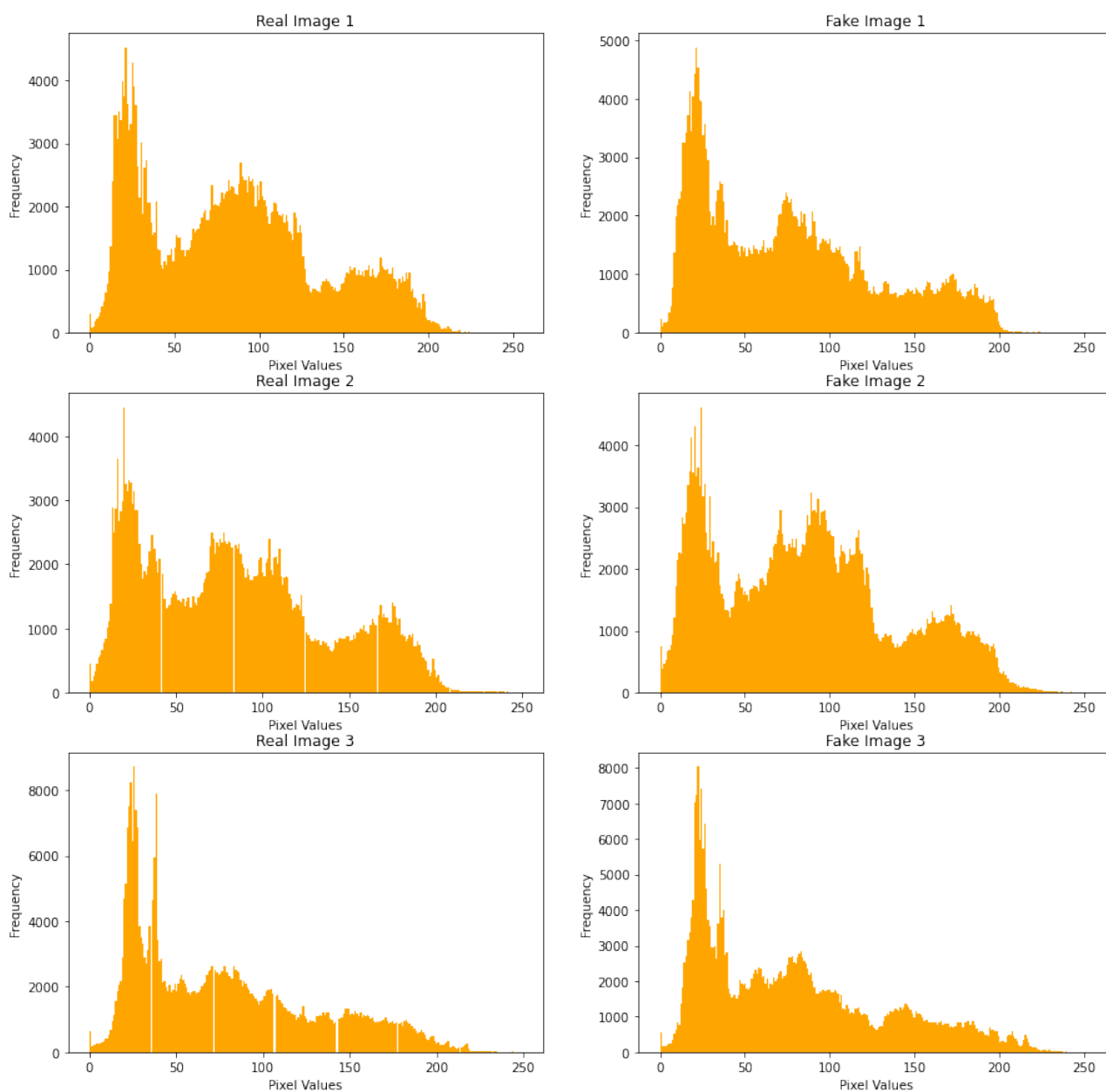
Here instead of jumping straight into the analysis of thousands of images, we first decided to explore and hunt for evidences that could test any speculations about the image distributions.

Figure 1.1: The set of images that were used for the initial analysis



The images chosen were not a paired set in the dataset, in fact the dataset doesn't even have classes or faces for which their real/fake counterparts exists. Hence it was through visual discernment of the images that these sets of images were chosen. The original dataset that we are working with has a random set of real faces and a random set of deep fake faces.

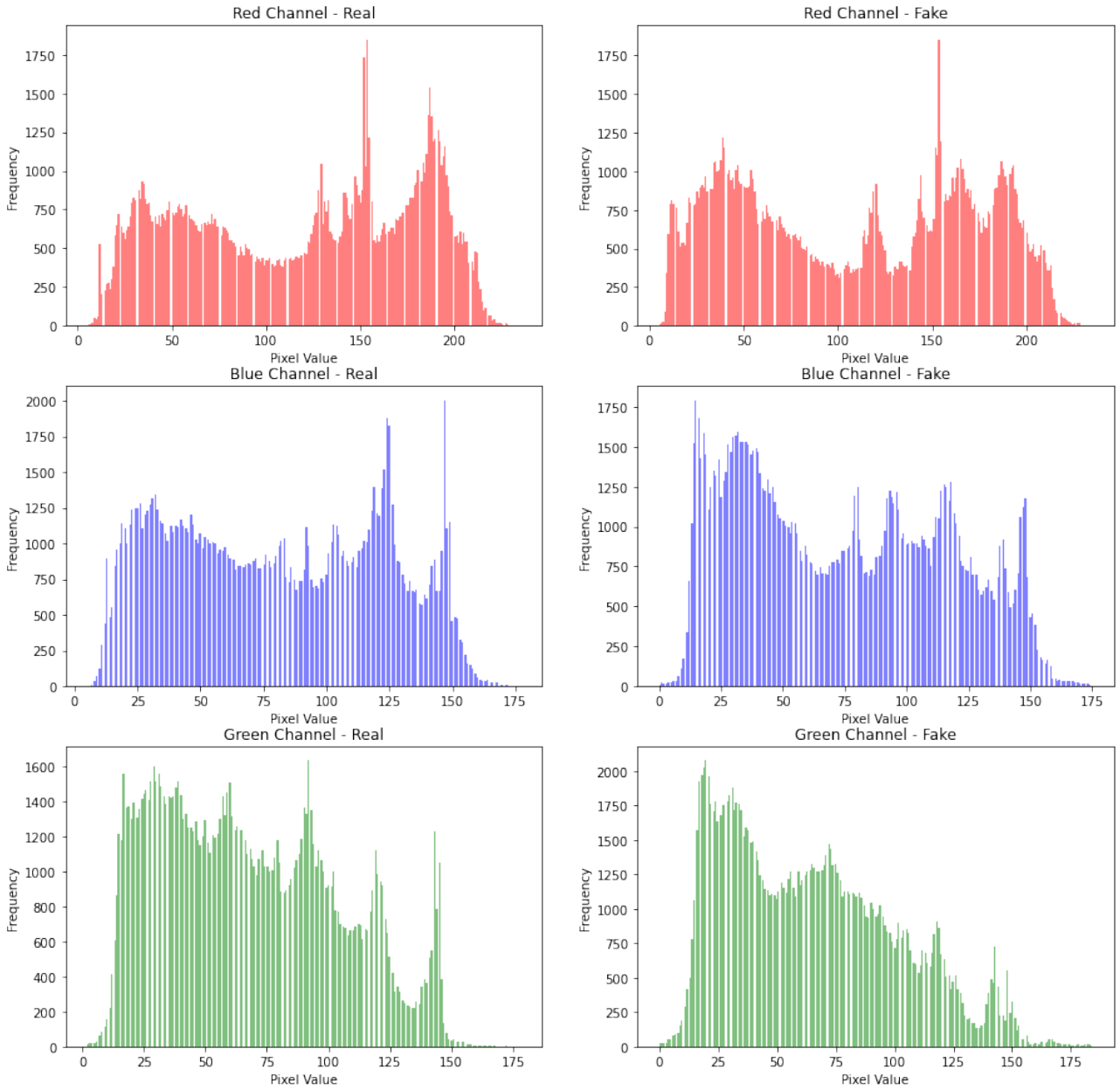
## Bin counts of the pixel values



These were the initial plots which was used to have a feel of what the pixel distribution looks like - made a histogram plot of the image using 256 bins and plotting these against the frequency of their occurrence.

It is difficult to make acute observations from the overall/combined values as the trend is quite similar (nothing unexpected as they are "Deep Fakes"). Although we can observe that there are at times there are discontinuities present in the real images but the texture of the fake images interpolates the same.

## Channel wise distn of pixels for Real\_3 and Fake\_3



We now go for a more fine grained approach and try and plot the channel wise distribution of pixels. Here we will only plot the distribution for one set of images and as we will see the differences in the patterns become more prominent.

One observation that could be made is that the frequency of some pixel values is conditionally higher than that of those compared with the real counterpart(All the plots can attest to this).One reason could be the presence of other factors like changes in brightness or contrast values.

Another observation is simply the unexpected rises and dips in the frequency of certain pixel values (green channel plots support this assertion)  
The histogram however only looks at the pixel value and not their spatial arrangement and hence we tend to use fourier transform to look at both at the same time



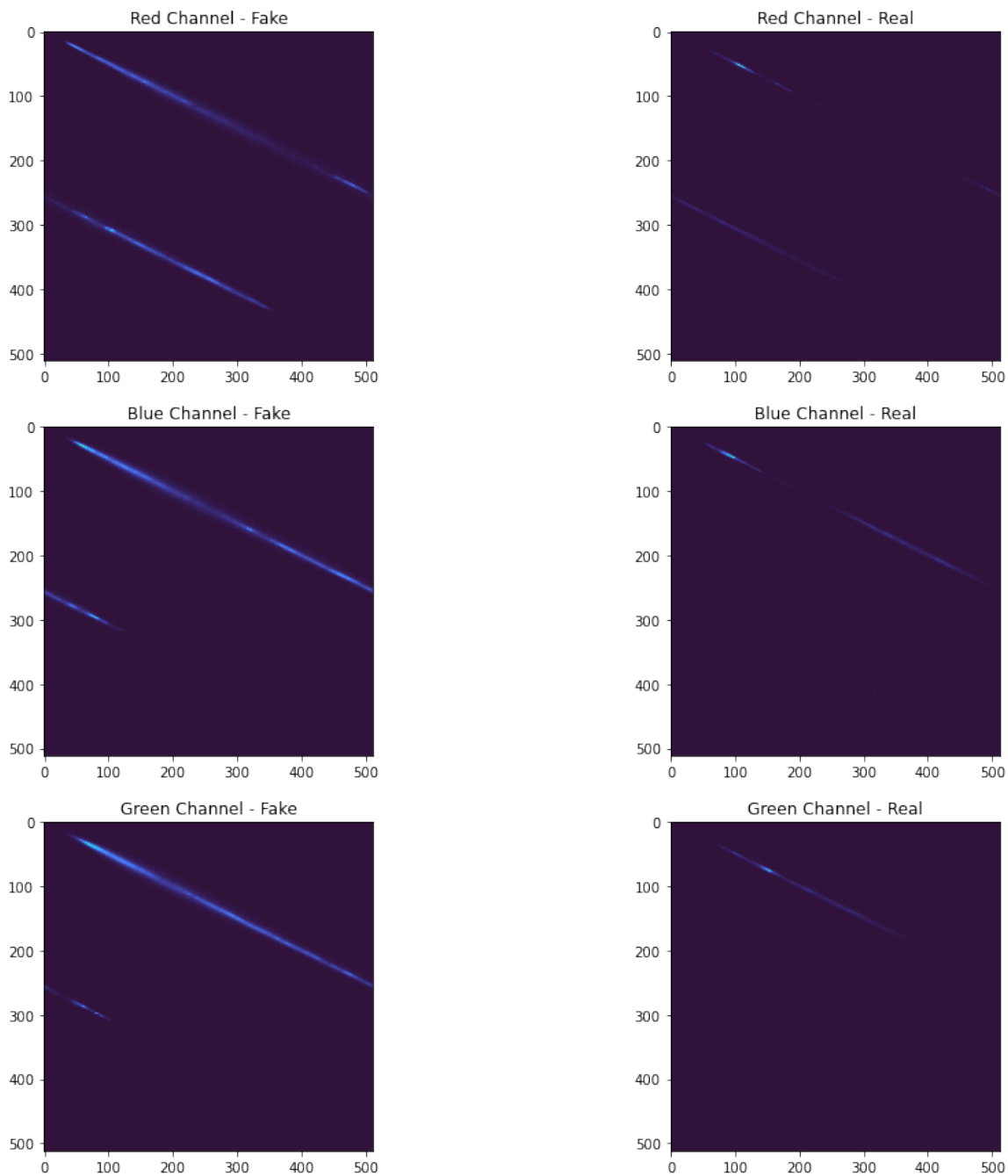


Figure 1.2: GLCM Plots for the respective channels

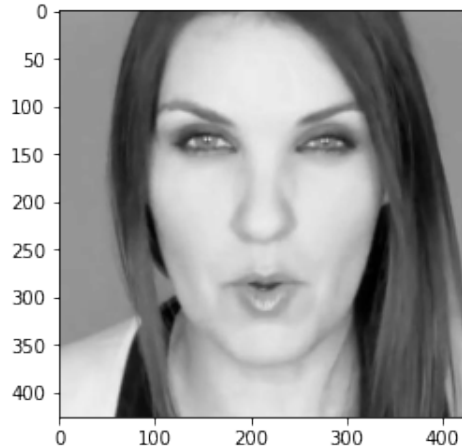
To overcome this limitation we have started experimenting with GLCM matrices - Gray Level Co-occurrence matrices. We will try to train a model on these characteristics as the new features that haven extracted from the images.

Link: <https://colab.research.google.com/eda-fake-data>

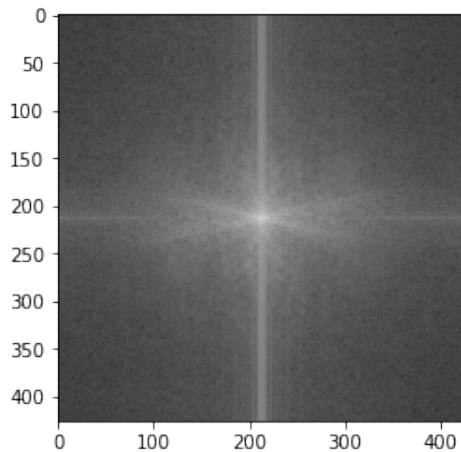
## 1.7 Image Processing Techniques

1. **Fourier Transform:** Fourier Transform breaks down an image into sine and cosine components. For this technique, Fast Fourier Transform (FFT) was implemented over DFT due to its fast computation. A signal is sampled over a period of time and divided into its frequency components. When we apply FFT to the input image, the center regions are represented by low-frequency components and outer regions are represented by high-frequency components. By applying a mask (low pass filter) in the center region, we get better edge detection.

Image used:



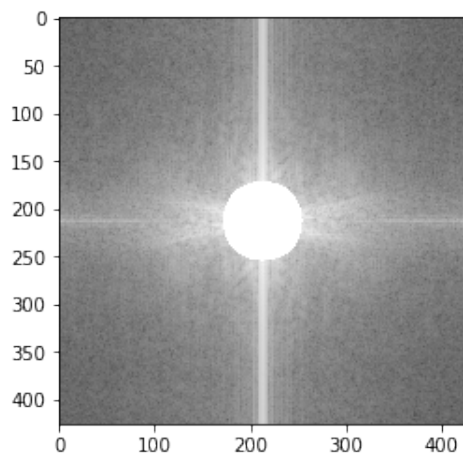
- (a) Image is converted to float32 and output is set to complex output, so as to extract the magnitude of this complex number. Fourier transform is applied by shifting the zero-frequency component to the center of the array.



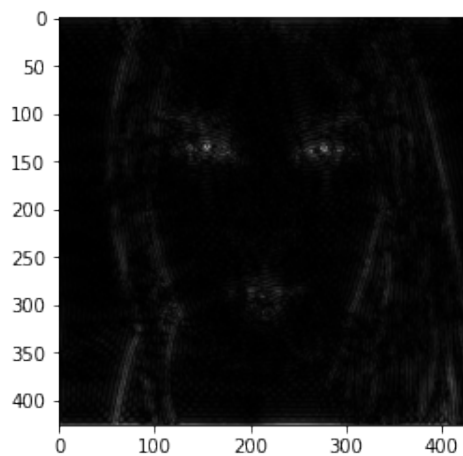
The white vertical and horizontal lines refer to the sharp horizontal and vertical elements of the image. The central speck is the DC component of the image, which gives the information of the color intensity of the image. If this central speck is altered, it can drastically change the image altogether. These specks pertain to the noise in the image.

- (b) To block off the central regions or pixels, A circular high pass filter is applied as a mask to the central region. The remaining specks are used for edge detection because low frequencies at the center are blocked and

only high frequencies are allowed. Since the edges are high-frequency components, they amplify noise.



- (c) Now that the central speck, the inverse FFT is applied and lastly, inverse DFT to convert back to image domain from the frequency domain



- (d) Link for the code - <https://colab.research.google.com/fourier-transform>

## 1.8 Models Implemented

This section comprises of Deep Learning Models which were performed before and after the preprocessing and the exploratory analysis of the data. Here are some implemented models:

### 1.8.1 MesoNet

**MesoNet:** MesoNet classifier is the pre-trained model for the detection of deepfakes and facial video forgery, as it has pre-trained weights. We initialised the model with following parameters:

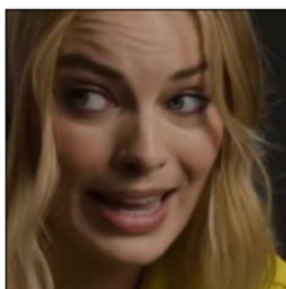
1. Learning Rate:- 0.001
2. Optimizer -: Adam
3. Loss:- Mean Square Error
4. Metrics:- Accuracy

#### Model Architecture:

1. Input: The input in the model is given as a single image of the dimensions and channels (256,256,3)
2. Hidden Layers: There were 4 hidden layers of 2 dimensional Convolution and MaxPooling ( pool-size = 2X2) . The first 2 layers has 8 filters, with kernel size in 1st hidden layer is (3,3) and in 2nd layer it is (5,5). In the last 2 layers i.e 3rd and 4th, there were 16 filters in each layer and the kernel size was same in both the layers (5,5). Padding was same across the whole network and the activation function used was ReLU.
3. Output Layer: Output layer returns the single dimensional array of features.

After initialising the model we used Image Data Generator to classify the images into 2 parts: Real and DeepFaked. With the help of pretrained weights and model we made the predictions on the data and labelled it correctly. We made an array of the images which are real and predicted as real and real but predicted as fake and vice-versa.

1. Real and Predicted as Real: Here the model confidence shows the accuracy how much a prediction of real is correct.



Model Confidence ::  
0.845813



Model Confidence ::  
0.987334

2. Real but predicted as Fake:



Model Confidence ::  
0.260479



Model Confidence ::  
0.345430

3. DeepFake and predicted as deepfake:



Model Confidence ::  
0.987530

4. DeepFake and predicted as Real:



Model Confidence ::  
0.554716

### 1.8.2 CNN-1 (Trial)

In this model we initialise 2 classes and entered the parameters and dimensions of the image. Preprocessing was done using Tensorflow operations such as decoding and resizing. We initialised labels for images as 0 for DeepFake and 1 for Real. **Model Summary :**

1. Input: Image data of predefined dimensions and channels.
2. Loss: Sparse Categorical Cross Entropy
3. Optimizer: rmsprop
4. Metrics: Accuracy
5. Hidden Layers: 3
6. Model Type: Sequential

7. 3 Convolutional layers of (32X32X64) filters and kernel size of (3X3).

8. Hidden Layer Activation Function: ReLu

9. MaxPooling Layer of (2X2) poolsize.

Batch Sized was initialised as 32 and training data was split into 80-20 ratio i.e. (80 for training and 20 for testing). Training epoch was set as 5.

### Output:

```
Epoch 1/5
50/50 [=====] - 1024s 15s/step - loss: 206.2384 - accuracy: 0.5019 - val_loss: 0.6930 - val_accuracy: 0.5096

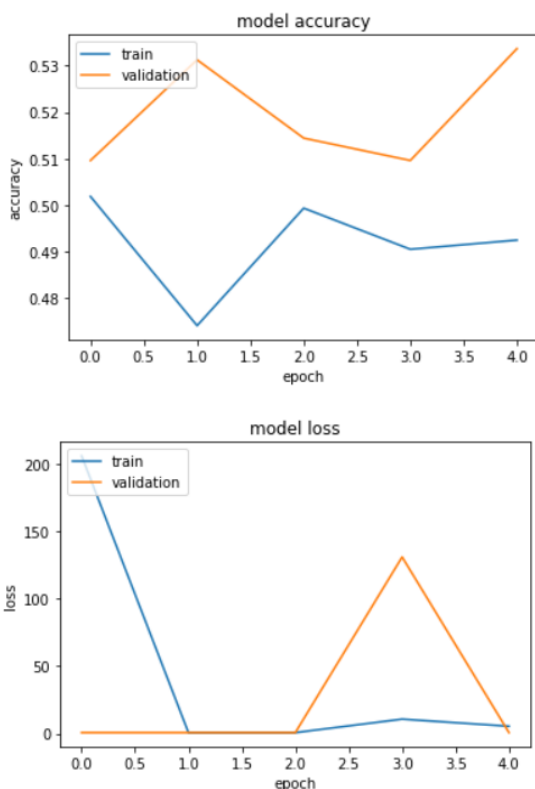
Epoch 00001: val_accuracy improved from -inf to 0.50962, saving model to model.h5
Epoch 2/5
50/50 [=====] - 735s 15s/step - loss: 0.6925 - accuracy: 0.4741 - val_loss: 0.6931 - val_accuracy: 0.5312

Epoch 00002: val_accuracy improved from 0.50962 to 0.53125, saving model to model.h5
Epoch 3/5
50/50 [=====] - 739s 15s/step - loss: 0.6928 - accuracy: 0.4994 - val_loss: 0.6931 - val_accuracy: 0.5144

Epoch 00003: val_accuracy did not improve from 0.53125
Epoch 4/5
50/50 [=====] - 737s 15s/step - loss: 10.7189 - accuracy: 0.4905 - val_loss: 131.1313 - val_accuracy: 0.5096

Epoch 00004: val_accuracy did not improve from 0.53125
Epoch 5/5
50/50 [=====] - 745s 15s/step - loss: 5.4543 - accuracy: 0.4925 - val_loss: 0.6928 - val_accuracy: 0.5337

Epoch 00005: val_accuracy improved from 0.53125 to 0.53365, saving model to model.h5
50/50 [=====] - 121s 2s/step - loss: 0.6932 - accuracy: 0.4944
13/13 [=====] - 274s 3s/step - loss: 0.6931 - accuracy: 0.5024
train_loss = 0.6932117938995361, train_acc = 49.437499046325684%
val_loss = 0.6931356191635132, val_acc = 50.24038553237915%
```



### 1.8.3 DenseNet and DenseNet with GrayScale

The main premise of this model was to gauge how much importance the color has on the classification of fake vs real images. Color images and gray images differ in pixel and resolution, hence this can impact in classification of fake images. For this model, we used the prebuilt DenseNet

but changed all the images to grayScale using colormode parameter. A label of 1 was assigned to real images and 0 to fake images.

### Model Summary :

1. Input: Image data of predefined dimensions and channels.
2. Loss: binary crossentropy
3. Optimizer: Adam
4. Metrics: Accuracy

```
Model: "sequential"
-----
Layer (type)                 Output Shape              Param #
-----
densenet121 (Model)          (None, 7, 7, 1024)       7031232
-----
global_average_pooling2d (G1 (None, 1024)          0
-----
dense (Dense)                (None, 1)                1025
=====
Total params: 7,032,257
Trainable params: 6,948,609
Non-trainable params: 83,648
-----
```

### Output:

1. Train: Training Epoch was set as 3.

#### For DenseNet:

```
Epoch 1/3
2021-12-10 04:48:39.664868: I tensorflow/stream_executor/cuda/cuda_dnn.cc:369] Loaded cuDNN version 8005

1562/1562 [=====] - 971s 611ms/step - loss: 0.5638 - accuracy: 0.6980 - val_loss:
0.7386 - val_accuracy: 0.6585
Epoch 2/3
1562/1562 [=====] - 628s 402ms/step - loss: 0.3070 - accuracy: 0.8686 - val_loss:
0.5491 - val_accuracy: 0.7782
Epoch 3/3
1562/1562 [=====] - 624s 399ms/step - loss: 0.1752 - accuracy: 0.9300 - val_loss:
0.1710 - val_accuracy: 0.9321
```

#### For DenseNet with GrayScale:

```
Epoch 1/3
2021-12-10 04:00:53.586680: I tensorflow/stream_executor/cuda/cuda_dnn.cc:369] Loaded cuDNN version 8005

1562/1562 [=====] - 1010s 633ms/step - loss: 0.5298 - accuracy: 0.7279 - val_loss: 0.5536 - val_accuracy:
0.7514
Epoch 2/3
1562/1562 [=====] - 600s 384ms/step - loss: 0.2859 - accuracy: 0.8768 - val_loss: 0.5320 - val_accuracy:
0.7845
Epoch 3/3
1562/1562 [=====] - 621s 397ms/step - loss: 0.1562 - accuracy: 0.9376 - val_loss: 0.2205 - val_accuracy:
0.9106
```

## 2. Metrics:

For DenseNet:

```
ROC AUC Score: 0.9825735900000001
AP Score: 0.9810687171798793

      precision    recall  f1-score   support

0         0.93        0.94        0.93        10000
1         0.94        0.93        0.93        10000

 accuracy          0.93          0.93          0.93        20000
 macro avg         0.93          0.93          0.93        20000
weighted avg         0.93          0.93          0.93        20000
```

For DenseNet with GrayScale:

```
ROC-AUC Score: 0.97812024
AP Score: 0.9779177141650164

      precision    recall  f1-score   support

0         0.88        0.96        0.92        10000
1         0.95        0.87        0.91        10000

 accuracy          0.91          0.91          0.91        20000
 macro avg         0.92          0.91          0.91        20000
weighted avg         0.92          0.91          0.91        20000
```

### 1.8.4 Siamese Network

1. **Description** : Siamese Network Architecture using two Sister Network and implemented using one shot learning. The main objective of the network was not to classify but rather learn the differences between the two networks. Hence, **contrastive loss** has been used to train a model to distinguish between two faces under different conditions - angles, illumination, glasses etc

$$\text{Loss} = (1-Y) \frac{1}{2}(D_W)^2 + (Y) \frac{1}{2} \max(0, m - D_W)$$

where,

Y = If the inputs are from the same class then Y = 0 , otherwise Y = 1.

$D_W$  = Euclidean Distance between the outputs of the two siamese networks

m = Margin Value



The network class:

```
class SN(nn.Module):
    def __init__(self):
        super(SN,self).__init__()
        self.cnn1 = nn.Sequential(

            nn.ReflectionPad2d(1),
            #Padding the image with a reflection across all the dimensions
            nn.Conv2d(in_channels = 1, out_channels = 4, kernel_size = 3),
            nn.ReLU(inplace = True),

            nn.ReflectionPad2d(1),
            nn.Conv2d(in_channels = 4, out_channels = 8, kernel_size = 3),
            nn.ReLU(inplace = True),

            nn.ReflectionPad2d(1),
            nn.Conv2d(in_channels = 8, out_channels = 8, kernel_size = 3),
            nn.ReLU(inplace = True)

        )

        self.fc1 = nn.Sequential(
            nn.Linear(in_features = 8*100*100, out_features = 500),
            nn.ReLU(inplace = True),
            nn.Linear(in_features = 500, out_features=500),
            nn.ReLU(inplace = True),
            nn.Linear(in_features= 500, out_features=5)
        )

    def forward_once(self,t):
        """
        Making a single pass to the whole network
        """
        output = self.cnn1(t)
        output = output.view(output.size()[0], -1)
        #Memory efficient reshaping of the tensor
        output = self.fc1(output)
        return output

    def forward(self, input1, input2):
        """
        The sister networks must have the same set of weights,
        hence instead of creating two different networks we
        make two passes through the network.
        """
        output1 = self.forward_once(input1)
        # Image 1 passes through the network
        output2 = self.forward_once(input2)
        # Image 2 passes through the network
        return output1,output2
```

## The Contrastive Loss Function

```
class ContrastiveLoss(torch.nn.Module):
    """
    Contrastive loss function implemetation

    """

    def __init__(self, margin=2.0):
        super(ContrastiveLoss, self).__init__()
        self.margin = margin

    def forward(self, output1, output2, label):
        euclidean_distance = f.pairwise_distance(output1, output2,
                                                  keepdim = True)
        loss_contrastive = torch.mean((1-label) * torch.pow(
            euclidean_distance, 2)+(
            label) * torch.pow(torch.clamp(self.margin -
            euclidean_distance, min=0
            .0), 2))

        return loss_contrastive
```



Figure 1.3: Comparison of different faces resulting into a high score for the dissimilarity



Figure 1.4: Comparison of similar faces resulting into a low score for the dissimilarity

2. **Output:** Dissimilarity Score where the lesser the score the more familiar the output.

3. **Link for the code**

<https://colab.research.google.com/siamese-net>

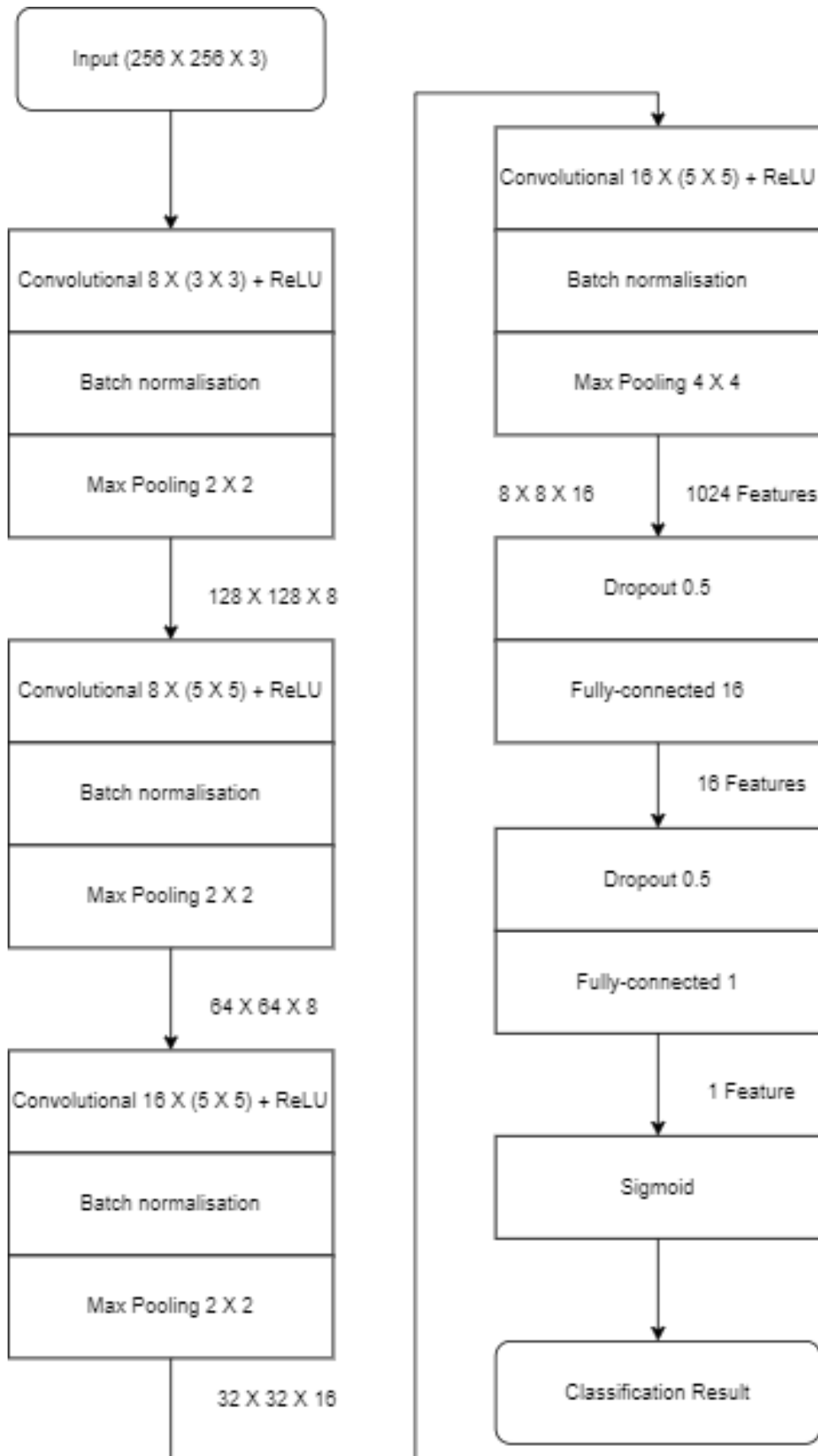
### 1.8.5 Deep Convolutional Neural Network Model

The model, as mentioned above is a binary classifier built as a Deep Convolutional Neural Network (D-CNN), trained to classify images into one of two classes. One class refers to "real" images (images of real people) and the other refers to "fake" images (images generated by DeepFake AI).

#### 1. Model Architecture:

- (a) 3 X 256 X 256 input layer, with the input being scaled by 255 and augmentations applied on it.
- (b) Convolutional layer with 8 filters, 3 x 3 in size and stride of 1, followed by a max pooling layer of size 2 x 2.
- (c) Convolutional layer with 8 filters, 5 x 5 in size and stride of 1, followed by a max pooling layer of size 2 x 2.
- (d) Two convolutional layers with 16 filters, 5 x 5 in size and stride of 1, followed by max pooling layers with pooling window of 2 x 2.
- (e) Fully-connected layer with 16 units.
- (f) Fully-connected output layer with 1 unit and sigmoid activation.

Figure 1.5: The Model Architecture



While this architecture is closely followed, experiments with various activation functions have been carried out and the code is designed such that it is extremely convenient to

switch the activation function for the entire model. Specifically, in addition to using the standard ReLU activation, experiments with ELU and LeakyReLU have also been carried out. ReLU is the activation function of choice since there is no apparent risk of dead neurons. Additionally, there exists a LeakyReLU activation after the fully-connected 16-unit layer.

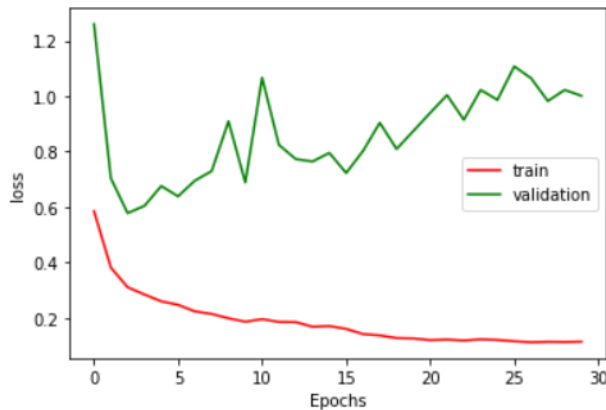
- (a) **Bath Normalization:** Batch Normalization is added after each convolutional layer to improve convergence speed and to combat overfitting.
- (b) **Dropout:** Dropout is added after the fully-connected 16-unit layer to combat overfitting.

Table 1.1: Dataset Information

Set	Sized of the forged images	Size of real images	Total
Training	5111	7250	12361
Test	2889	4259	7148
Total	8000	11509	19509

## 2. Results:

### (a) Loss Curve:



### (b) Model Accuracy:

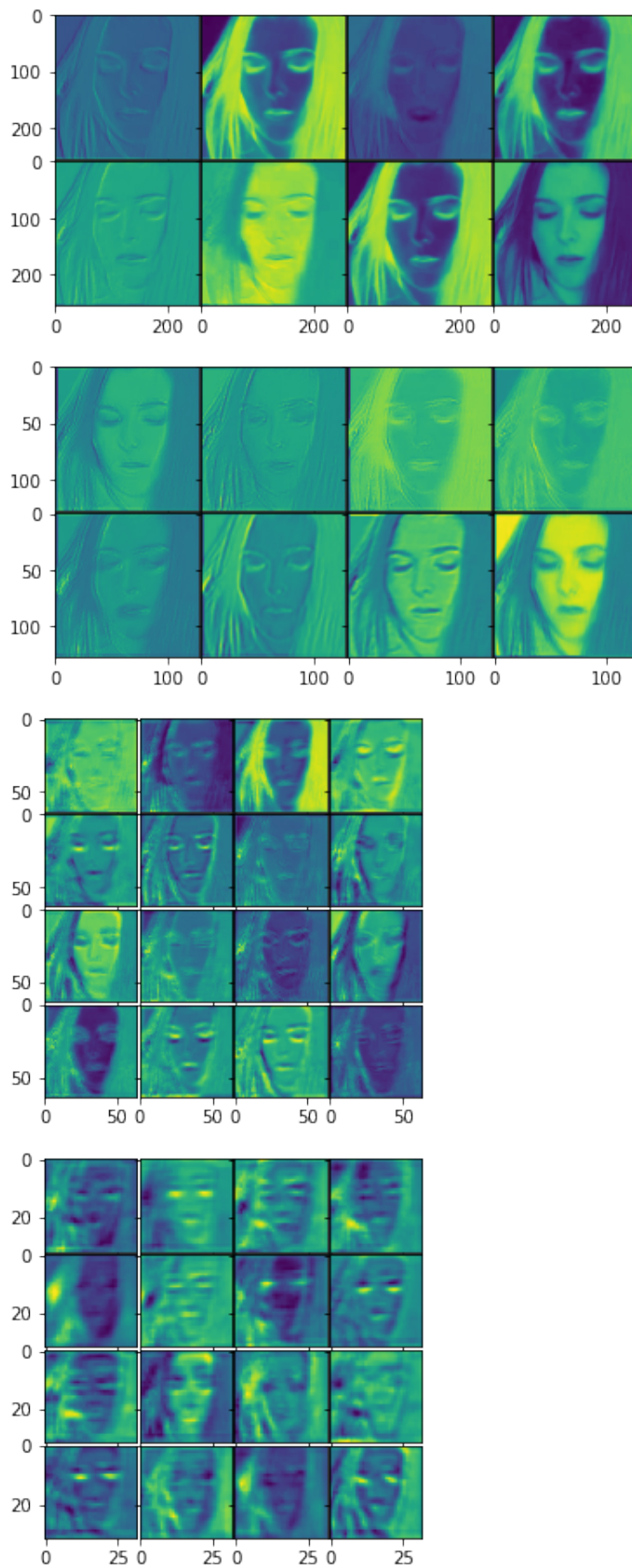
```
Found 12353 images belonging to 2 classes.
194/194 [=====] - 53s 268ms/step - loss: 0.2513 - accuracy: 0.8953
[0.25132259726524353, 0.8953290581703186]
```

### (c) Classification Report:

```
Found 12353 images belonging to 2 classes.
194/194 [=====] - 53s 273ms/step
```

	precision	recall	f1-score	support
0	0.86	0.89	0.88	5103
1	0.92	0.90	0.91	7250
accuracy			0.90	12353
macro avg	0.89	0.89	0.89	12353
weighted avg	0.90	0.90	0.90	12353

### (d) Hidden Layer Visualisations:



**Link to the code :** <https://colab.research.google.com/drive/DeepFake-1>



# Chapter 2

## Timeline - Workflow

The research project was primarily divided into two major sections/chunks. Firstly, the research Phase where we were supposed to get familiar with all the current roundabouts and happenings in the field of Deep Fakes.

Secondly, using the insights gained and formulate our own mathematical model/algorithm for Deep-Fake Detection using this proposed novel architecture.

### 2.1 Research Phase - Accomplished

#### 2.1.1 Week 1 - Logistics

The very first week of the project was spent in understanding the problem statement and also figuring out the logistics - plans, library management for the insights, tools etc.

#### 2.1.2 Week 2 - Survey Papers

We jumped right into the literature review by first looking at the various survey papers provided to us. Survey papers were crucial as they helped us gauge the current happenings in the domain.

#### 2.1.3 Week 3 - A Deeper Dive

Once we had an outlook on the many architectures and various methods, metrics and loss functions, we decided to narrow down our review to a few State of the Art Models that were extensively reviewed.

#### 2.1.4 Week 4 - Conventional : Non DL based Methods

Deep learning was used a lot in the Detection process but also there were a few conventional methods that proved to be really useful. So we hoped on to finding hints about these and reviewing the papers related to these non- Deep Learning based techniques.

#### 2.1.5 Week 5 - Recapitulation

We went through all the summaries and notes that we had prepared to perform discussions on what worked well and how the evolution of the various GAN architectures took place.



### 2.1.6 Week 6 - Data Gathering

For training our hybrid network we have thought of using an amalgamation of a few datasets. Hence we went through all the papers, yet again in order to point out all the datasets used and their implications and biases.[?]

Most of our insights have been summarized in a tabular form so that it is easy to understand and hence could be useful to refer in future. The table is as follows :

# Summary

Study	Methods	Classifiers	Process	Metrics	Assumptions	Loss Funcs
[12]	CNN Deep Learning	ResNet50, ResNet100, MXNet, SENet50, VGG2		Center-Loss, SphereFace, VGGFace2, MS1MV2,R100,ArcFace	ArcFace consistently outperforms the state of the art and can be easily implemented with negligible computational overhead.	SphereFace, Cos-Face, CM1, CM2, Norm-Softmax (NS), NS+Intra ,NS+Inter ,NS+Intra+Inter (0.35), Triplet ,ArcFace+Intra ,ArcFace+Inter ,ArcFace+Intra+Inter15 ,ArcFace+Triplet
[6]	CNN Deep Learning	GoogLeNet InceptionV3 MesoNet SVM ResNet-50 Designed CNN Logistic Regression Model XceptionNet Designed CapsuleNet SPPNet	Usage of Auto Encoders. Both Encoders and Decoders	Mask-SSIM score, ROC curve and AUC Score	a comprehensive evaluation of DeepFake detection methods and datasets to demonstrate the escalated level of challenges	
[8]	CNN+RNN Deep Learning	Multi Task CNN EfficientNet GRU	Automatic Weighting	Accuracy(Validation)		binary cross-entropy loss ArcFace loss
[9]	CNN Deep Learning	Generative adversarial networks variational autoencoders (VAE) KNN, MLP, Logistics regression		ROC curve AUC		
[10-26]	Deep Neural Networks	GANs VGGFace VGG-NET		AUC, ROC Values, True positive rate and False positive rate		
[10-29]	CNN + Steganalysis,CFA estimation using GMM (Guassian Mixture Models)	Deep CNN	Hidden tampered regions are detected using the difference between local noise residuals and the global noise residuals	ROC,AUC, Class Activation Maps, DTC Coefficient , CFA Pattern	Different imaging devices produce different CFA regions wrt tampered and authentic images.	Triplet loss + SVM Kernel loss

Summary						
Study	Methods	Classifiers	Process	Metrics	Assumptions	Loss Funcs
[19]	FGSM(Fast Gradient Sign Method), PGD (Multi step - Projected Gradient Descent), Traditional Adversarial Learning(Generator + Discriminator), Spatial Transformed Adersarial Learning(SAT), Blurring Adversarial Training, Noise Variance Analysis, Digital Shadow Writing Analysis	GANs Auto-Encoders Attention Mechanism	Adversarial training method that attempts to blur out the specific artifacts, by introducing pixel-wise Gaussian blurring models	AUC, ACC		Generator and Discriminator functions
[5]	Crowd - sourcing based eval - Quality-Crowd 2 framework	Xception and Efficient-Net (B4 variant)	Computer vs Human Accuracy tested	ANOVA, AUC,ACPER(Attack presentation error rate)	The accuracy of the algorithms have some correlation to the visual appearance of deepfakes, which is not the case.	
[15]	GAN Pipelining Steganalysis Deep Learning	SVM CNN RBM Attention Mechanism DRN Incremental Learning GAN Discriminator Score-level Fusion		EER AUC TCR(True Classification Rate)		Triplet Loss
[25]	Video/Audio modality embedding and emotions embedding	CNN for modality embedding and RNN for emotions embedding				Triplet Loss
[22]	EfficientNet family XceptionNet	CNN	The inclusion of attention mechanisms, and the other to include siamese training strategies in the learning process	AUC ,LogLoss		LogLoss, Triplet Margin Validation Loss
[29]	MesoInception-4, Cozzolino, Rahmouni ,XceptioniNet	CNN				
[30]	CNN+LSTM, DBiRNN, ShallowNet, Xception, MesoNet	CNN + RNN	Single Domain Learning. Merge Learning	CNN, RNN	Single Domain Learning. Merge Learning.	

## 2.2 Execution Phase

### 2.2.1 Week 7 - Dataset Exploration and Preparation

We are proposing an idea of a combined dataset which is prepared by exploring three different datasets such as : FaceForensics++, DeepFake Detection Challenge (DFDC) Dataset. These datasets have wide range of videos but we will filter the videos as per our requirements.

### 2.2.2 Week 8 - Preprocessing of Data

As we got our new hybrid dataset we are all set to perform preprocessing operations on it. Preprocessing of the dataset includes: Splitting the videos and images into the frames, Face Detection, Cropping the face, Creating new face cropped video and finally saving the video.

### 2.2.3 Week 9 - Designing Model Architecture

After preprocessing, model building operations will be started using different architectures such as ResNet, VGG-16, SVM, Deep CNN etc. Designing the model will also various pooling and sequential layers along with the activation functions.

### 2.2.4 Week 10 - Training Workflow

We will train our models of different architectures and evaluate them with several metrics and loss functions. The architectural model of best accuracy will be used for testing the dataset to make classifications and predictions.

### 2.2.5 Week 11 - Prediction Workflow

After testing of model is done we will use the model to make predictions whether the video is deep faked or real.

### 2.2.6 Week 12,13 - Buffer Weeks

These weeks are meant to be used in case of any discrepancy in the tentative timeline. And also once approved by our advisor, and if time permits, we would like to implement the proposed algorithm and also the many conventional architectures, into a Streamlit application. The application would be in form of a **live dashboard** where once can upload the video or image and detect if the image is fake or not, along with the **various statitics of the model**. They would also be able to **play around with the standard models** and their paramaters to some extent. The Streamlit application would be our way of contributing back to the domain.



# Chapter 3

## Conclusion and Observations

### 3.1 Observations from papers

Although CNN and many of the variations for the same such as CNN + RNN [5], CNN+LSTM [12], CNN along with another stream that uses an SVM classifier [15], have worked just fine, but they still suffer from the problem of poor generalization. Also the post processing techniques become less and less effective as the performance of the CNN based model degrades significantly. DeepCNNs that we came across we could conclude that the Adversarial training has been by far the most effective and popular training methodology with many variations [1]- Conventional Adversarial Learning , Blurred Adversarial Learning, wherein the images are blurred based on the gradients.

We also came across a few conventional methods - like analyzing steganalysis features [15] or observing CFA patterns from the learned embeddings of the images. People also used Improved DTC coefficient for the same. One can also use the spatial temporal features [11] to detect whether the image is a forged one or not.

Also, we stumbled upon many metrics and loss functions, the most popular among which is the Triplet Loss Function [15], followed by other loss functions that were enhanced versions of the Geodesic Distances for example - ArcFace, SphereFace, CosFace and various combinations of these with Intra, Inter and Norm-Softmax functions. [3]

Although we had been exposed to many architectures and techniques, but we were still able to grasp only a few of them and hence we look forward to implement the code of the described methods. We faced this issue of almost no paper having the code for their paper and this could act as resistance to the early researchers.

## 3.2 Experimental Results

Models				
Model	Accuracy	Loss Function	F-1 Score	ROC-AUC Score
DenseNet	0.93	binary Cross-Entropy	0.93	0.982
Dense-Net with GrayScale	0.92	binary Cross-Entropy	0.91	0.978
MesoNet	0.80	Mean Square Error	-	-
Deep Convolutional Neural Network	0.89	Binary Cross Entropy	0.88	-
CNN - Siamese Network	0.87	Contrastive Loss Function	-	-

## 3.3 Future Aspects

For future work, we might try to work on more explainable models by performing similar feature engineering that is reflected in the Exploratory Data Analysis. The major area of focus would be to convert the images into more comprehensible statistics - for example - A GLCM matrix has many properties associated with it -Dissimilarity, correlation,homogeneity,energy and contrast. All of the above measures are useful in identifying the texture of the image.Texture of the image is particularly important as we have encountered images that have texture as the major discerning factor that could lead us to identification of some low quality deep faces and once we achieve this milestone we could move on to tackle even higher quality Deep Fakes.The work on this approach has already begun and as we are learning how to interpret these measures, it might take us time to actually translate all the images into these measures and only then can we use explainable algorithms like Decision Trees and Random Forests.

# Bibliography

- [1] S. Agarwal, H. Farid, T. El-Gaaly, and S.-N. Lim. Detecting deep-fake videos from appearance and behavior. pages 1–6, 2020.
- [2] L. Bode, D. Lees, and D. Golding. The digital face and deepfakes on screen. *Convergence*, 27(4):849–854, 2021.
- [3] J. Deng, J. Guo, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *CoRR*, abs/1801.07698, 2018.
- [4] T. Do Nhu, I. Na, and S. Kim. Forensics face detection from gans using convolutional neural network, 10 2018.
- [5] D. Güera and E. J. Delp. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2018.
- [6] F. Juefei-Xu, R. Wang, Y. Huang, Q. Guo, L. Ma, and Y. Liu. Countering malicious deepfakes: Survey, battleground, and horizon. *arXiv preprint arXiv:2103.00218*, 2021.
- [7] P. Korshunov and S. Marcel. Subjective and objective evaluation of deepfake videos. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2510–2514. IEEE, 2021.
- [8] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu. Celeb-df: A new dataset for deepfake forensics. *CoRR*, abs/1909.12962, 2019.
- [9] F. Matern, C. Riess, and M. Stamminger. Exploiting visual artifacts to expose deepfakes and face manipulations. In *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pages 83–92, 2019.
- [10] D. M. Montserrat, H. Hao, S. K. Yarlagadda, S. Baireddy, R. Shao, J. Horváth, E. R. Bartusiak, J. Yang, D. Guera, F. Zhu, and E. J. Delp. Deepfakes detection with automatic face weighting. *CoRR*, abs/2004.12027, 2020.
- [11] C. Shan, S. Gong, and P. Mcowan. Beyond facial expressions: Learning human emotion from body gestures. 01 2007.
- [12] S. Tariq, S. Lee, and S. S. Woo. One detector to rule them all: Towards a general deepfake attack detection framework. *CoRR*, abs/2105.00187, 2021.
- [13] R. Tolosana, R. Vera-Rodríguez, J. Fierrez, A. Morales, and J. Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection. *CoRR*, abs/2001.00179, 2020.
- [14] X. Xuan, B. Peng, J. Dong, and W. Wang. On the generalization of GAN image forensics. *CoRR*, abs/1902.11153, 2019.



- [15] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis. Two-stream neural networks for tampered face detection. *CoRR*, abs/1803.11276, 2018.