

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ
РАБОТА GRADUATION THESIS

Обнаружение дипфейков с помощью глубокого обучения и CNN / Deepfake
detection using deep learning and CNN

Обучающийся / Student Сафдари Абизер

Факультет/институт/кластер/ Faculty/Institute/Cluster факультет
цифровых трансформаций

Группа/Group J42322с

Направление подготовки/ Subject area 01.04.02 Прикладная математика и информатика

Образовательная программа / Educational program Большие данные и машинное
обучение 2020

Язык реализации ОП / Language of the educational program Английский

Статус ОП / Status of educational program МОП

Квалификация/ Degree level Магистр

Руководитель ВКР/ Thesis supervisor Басов Олег Олегович, доцент, доктор технических
наук, Университет ИТМО, факультет цифровых трансформаций, профессор
(квалификационная категория "ординарный профессор")

Обучающийся/Student


Документ подписан	
Сафдари Абизер	
18.05.2022	

(эл. подпись/ signature)

Сафдари
Абизер

(Фамилия И.О./
name and surname)

Руководитель
ВКР/Thesis
supervisor

Документ подписан	
Басов Олег Олегович	
18.05.2022	

(эл. подпись/ signature)

Басов Олег
Олегович

(Фамилия И.О./
name and surname)

**Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University**

**ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ /
OBJECTIVES FOR A GRADUATION THESIS**

Обучающийся / Student Сафдари Абизер

Факультет/институт/кластер/ Faculty/Institute/Cluster факультет
цифровых трансформаций

Группа/Group J42322c

Направление подготовки/ Subject area 01.04.02 Прикладная математика и информатика
Образовательная программа / Educational program Большие данные и машинное обучение 2020

Язык реализации ОП / Language of the educational program Английский

Статус ОП / Status of educational program МОП

Квалификация/ Degree level Магистр

Тема ВКР/ Thesis topic Обнаружение дипфейков с помощью глубокого обучения и CNN / Deepfake detection using deep learning and CNN

Руководитель ВКР/ Thesis supervisor Басов Олег Олегович, доцент, доктор технических наук, Университет ИТМО, факультет цифровых трансформаций, профессор (квалификационная категория "ординарный профессор")

Основные вопросы, подлежащие разработке / Key issues to be analyzed

This paper researches how to Deep-Fake is implement and how to detect these attacks to prevent threats.

The goal of the research is to improve the accuracy of DeepFake videos detection.

The object of the research is DeepFake videos detection technologies.

The subject of the research is the method of DeepFake videos detection based on modern machine learning algorithms.

Дата выдачи задания / Assignment issued on: 01.08.2021

Срок представления готовой ВКР / Deadline for final edition of the thesis 30.06.2022

Характеристика темы ВКР / Description of thesis subject (topic)

Тема в области фундаментальных исследований / Subject of fundamental research: нет /not

Тема в области прикладных исследований / Subject of applied research: да / yes

СОГЛАСОВАНО / AGREED:

Руководитель
ВКР/Thesis
supervisor

Документ подписан	
----------------------	--

	
Басов Олег Олегович	
01.03.2022	

(эл. подпись)

Басов
Олег
Олегович

Задание принял к
исполнению/
Objectivesassumed BY

Документ подписан	
Сафдари Абизер	
28.03.2022	

(эл. подпись)

Сафдари
Абизер

Руководитель ОП/
Headof educational
program

Документ подписан	
Насонов Денис Александрович	
29.04.2022	

(эл. подпись)

Насонов Денис
Александрович

**Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University**

**АННОТАЦИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ
РАБОТЫ SUMMARY OF A GRADUATION
THESIS**

Обучающийся / Student Сафдари Абизер

Факультет/институт/кластер/ Faculty/Institute/Cluster факультет
цифровых трансформаций

Группа/Group J42322с

Направление подготовки/ Subject area 01.04.02 Прикладная математика и информатика

Образовательная программа / Educational program Большие данные и машинное обучение 2020

Язык реализации ОП / Language of the educational program Английский

Статус ОП / Status of educational program МОП

Квалификация/ Degree level Магистр

Тема ВКР/ Thesis topic Обнаружение дипфейков с помощью глубокого обучения и CNN /
Deepfake detection using deep learning and CNN

Руководитель ВКР/ Thesis supervisor Басов Олег Олегович, доцент, доктор технических наук, Университет ИТМО, факультет цифровых трансформаций, профессор (квалификационная категория "ординарный профессор")

**ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ
РАБОТЫ DESCRIPTION OF THE GRADUATION THESIS**

Цель исследования / Research goal

This study aims to provide an overview of the progress of research related to deepfake detection techniques. Within this scope of work, we will discuss machine learning-based deepfake detection methods, among other intelligent systems. The sole purpose of this review is to provide and make available knowledge of the current literature. It will be a new resource for researchers working on image recognition and security issues. In addition, it provides an accurate understanding of the potential challenges of recent research and the latest research guidelines.

Задачи, решаемые в ВКР / Research tasks

This master's thesis develops a novel architecture which can generate a full sequence of video frames given a source image and a target video. We were inspired by the idea to merge the multiple deepfake detection technologies into one algorithm. In our work, we propose a unified model using ResNext CNN for Feature Extraction followed by LSTM for Sequence Processing for the purpose of identifying Unnatural facial expression and MDS-based Audio-Video sync. The training is done end-to-end and the key points are learnt in a self-supervised way. Evaluation is demonstrated on the Kaggle deepfake competition dataset. The main contribution of our work is to develop novel neural networks for generating deepfake images/videos. Furthermore, our main motive is to generate datasets for deepfake detection challenges.

Краткая характеристика полученных результатов / Short summary of results/findings

We first proposed a model based on Convolution Neural Network that extracts frames from the video and detected faces that are constant throughout the video. After this these extracted frames are sent to two different models. The first model is sent the whole video in which it splits the videos into small segments and splits it based on audio and video. Heat-Map is generated based on the audio input and then these two inputs are sent into ResNet inspired models features are learnt and Contrastive loss is calculated along with the Loss entropy. All the loss functions are aggregated together for a combined loss, and this is used to train the model. The second model takes frames as input and crops these frames according to the most prominent face in the video. The input is then sent to ResNext based Convolutional Neural Network where the resultant output is a 2048 dimensional feature vector matrix. This is sent to the LSTM with 2048 hidden layers.

The LSTM has 2 outputs real/fake and input is 2048 feature vector. The LSTM uses a SoftMax layer to determine the confidence to the results from the LSTM. As the research progressed, we carried out a number of experiments with the goal of determining the impact that various hyperparameters, dataset compositions, and network architectures have on the overall performance of convolutional neural networks when it comes to Deepfake identification. This study also examined the performance of models with data that had not been observed before in order to evaluate their resilience. We discovered that the diversity of the dataset has a greater impact on performance than the quantity of the dataset. Overfitting occurred very immediately after using a dataset that had 25 frames for each DFDC movie. In comparison, the network was able to achieve an accuracy of 72.7 percent by using a dataset that only had one frame per DFDC video. This dataset consisted of just 1897 frames that were split between two categories. It would seem that increases in runtime are essentially linear with increases in the amount of the dataset. The rest of our study was conducted using the standard DFDC. Overfitting may be reduced thanks to preprocessing. Nevertheless, in this particular instance, it does not assist in making the model more accurate. It does not seem that preprocessing has any effect on the total amount of time spent on training. In a similar vein, greater dropout rates are able to maintain a closer proximity between the training and validation learning curves, but this does not assist the model reach a better accuracy. It does not seem that higher dropout rates have any effect on the total amount of time required for training. It also seems that the highest precision that a model is capable of achieving is unaffected by the size of the batches. On the other hand, it does seem to have an effect on the number of epochs required for the model to begin learning: the smaller the batch size, the sooner the algorithm begins learning. Last but not least, it seems that the training runtimes become shorter as the batch size increases. The best level of accuracy that could be attained was 92.5 percent.

Обучающийся/Student

Документ подписан	
Сафдари Абизер	
18.05.2022	

(эл. подпись/ signature)

Сафдари
Абизер

(Фамилия И.О./
name and surname)

Руководитель
ВКР/Thesis
supervisor

Документ подписан	
Басов Олег Олегович	
18.05.2022	

(эл. подпись/ signature)

Басов Олег
Олегович

(Фамилия И.О./
nameand surname)

ABSTRACT

The results that deep learning has obtained up to this point have been fairly promising across a wide range of challenging problem domains. These results have been accomplished through deep learning. On the other hand, current breakthroughs in deep learning have also been utilized to construct software that endangers the privacy of individuals as well as the safety of the country. One of them is called deepfakes, and it is possible to produce fake still photographs as well as videos that people are unable to detect as being forged. These fakes may be produced using computer software. Fake statements that are ascribed to major leaders have the ability to jeopardize the existing global order and the peace that it helps to preserve. Both positive and bad outcomes may be achieved via the usage of deepfakes. For instance, they may be used unethically in the commission of fraud, or constructively in the translation of languages or post-production dubbing of films. Both of these applications are examples of possible applications of the technology. In the most recent election that took place in India, the latter example was recently applied, which made it feasible for politician statements to be translated into a broad range of Indian languages across the country. In the past, this activity was accomplished by using computer graphic technology and three-dimensional models. But because to recent advancements in deep learning and computer vision, most notably GANs, the earlier methodologies are progressively being phased out in favor of deep learning methods. This is because GANs are the most notable of these new developments. The major focus of this research will be on the use of deep neural networks to the creation of changed faces in both still photographs and moving images captured in movies.

Detecting such attacks should be a priority as it can be used to cause panic and fear or spread wrong information. There are several technologies of DeepFake creation, based on:

1. Autoencoders-Decoders,
2. GAN (generative adversarial network).

Current researches are focused on studying technologies for exploring DeepFake. They are based on detection features listed below:

1. Unnatural eye movement;
2. Unnatural facial expressions;
3. Awkward facial-feature positioning;
4. A lack of emotion;
5. Awkward-looking body or posture;
6. Unnatural body movement;
7. Unnatural coloring;
8. Unnatural hair;
9. Unnatural teeth;
10. Blurring or misalignment;
11. Inconsistent noise or audio;
12. Images that look unnatural when slowed down;
13. Hashtag discrepancies;
14. Digital fingerprints;
15. Reverse image searches.

Inconsistency in the physical aspect of DeepFake films; signal level; and data driven are the three primary categories that current DeepFake detection systems fall into (depending on the characteristics employed). These approaches come with a number of restrictions and drawbacks, including social media washing, poor quality deepfake datasets, performance measurement, and more.

The purpose of this master's thesis is to create an innovative architecture that, given a source picture and a target video, is capable of producing an entire sequence of video frames. The concept of combining the several methods now in use to identify deep fakes into a single algorithm sparked our interest. In the work that we have done, we have proposed a unified model that identifies unnatural facial expressions and MDS-based audio-video sync by using ResNext CNN for Feature Extraction followed by LSTM for Sequence Processing. The goal of this model is to find unnatural face expressions. The training is completed in its entirety, and the fundamentals are learned in a manner that is self-supervised. The evaluation is carried out using the dataset provided by the Kaggle deepfake competition.

The creation of innovative neural networks for the generation of deepfake pictures and videos is the primary contribution that our work makes. In addition, the production of datasets for the purpose of participating in deepfake detection competitions is our primary goal.

SYMBOLS AND ABBREVIATIONS

GAN - Generative Adversarial Network

LSTM - Long Short-Term Memory

CNN - Convolutional neural network

MDS - Modality Dissonance Score

DFDC - Deepfake Detection Challenge

GUI - Graphical User Interface

FSGAN - Face Swapping GAN

PCA - Principal Component Analysis

IRLS - Iteratively reweighted least squares

SVM - support vector machine

AUC - area under curve

VAE - Variational Autoencoder

RNN - Recurrent Neural Network

DTC - Discrete Cosine Transform

TABLE OF CONTENTS

ABSTRACT.....	7
INTRODUCTION	15
1 RELATED WORK (deepfake overview).....	20
1.1 Pre-processing	21
1.2 Deepfake Creation	22
1.2.1 Graphics-based methods	23
1.2.2 Encoder-decoder methods	23
1.2.3 Generative Adversarial Networks(GAN).....	25
1.3 Deepfake Tools.....	26
1.3.1 FaceSwap	26
1.3.2 Face2Face.....	27
1.3.3 Neural Textures	28
1.4 Dataset	32
1.5 Deepfake Detection Methods Review (Literature review).....	33
1.5.1 Convolutional Neural Networks.....	33
1.5.2 Generative Adversarial Network.....	35
1.5.3 Autoencoders.....	37
1.5.4 Recurrent Neural Networks.....	37
1.5.5 Unnatural eye movement	38
1.5.6 Detecting Face Warping Artifacts.....	38
1.5.7 MesoNet	39
1.5.8 Biological Signals	40
2 RESEARCH METHODOLOGY	41
2.1 Model.....	41
2.1.1 Model 1 Finding Unnatural Facial Expression	41
2.1.1.1 Pre-processing	41
2.1.1.2 Model Architecture	42

2.1.2 Model 2 Audio Video synchronization.....	46
2.1.3 Proposed Model	50
2.2 Experiment	51
2.2.1 Dataset Composition.....	51
2.2.2 Preprocessing	52
2.2.3 Dropout	52
2.2.4 Batch Size	53
2.2.5 DCT-Residuals	53
2.2.6 Generalization and Robustness.....	54
3 RESULTS	55
3.1 Dataset Composition	55
3.2 Preprocessing.....	58
3.3 Dropout.....	62
3.4 Batch Size	66
3.5 DCT-Residuals	69
3.6 Generalization and Robustness.....	71
CONCLUSION.....	72
REFERENCES	74

LIST OF FIGURES

Figure 1 – Examples of preprocessing operations	22
Figure 2 – Example of face swapping with a graphics-based method.....	23
Figure 3 – Training process of the encoder-decoder method	24
Figure 4 – Example of face swapping with a encoder-decoder method.....	25
Figure 5 – A generative adversarial network.....	26
Figure 6 – Face2Face example.....	28
Figure 7 – Expression modification using neural textures [26].....	29
Figure 8 – Examples from DFDC dataset. Adapted from [25].....	33
Figure 9 – Pre-processing of video	42
Figure 10 – ResNext Architecture.....	43
Figure 11 – ResNext Working [50].....	44
Figure 12 – Overview of ResNext Architecture [50].....	44
Figure 13 – Overview of LSTM Architecture[51].....	45
Figure 14 – Internal LSTM Architecture [51]	45
Figure 15 – Training and prediction Flow	46
Figure 16 – Architecture of audio and Video stream.....	49
Figure 17 – Architecture of the model	50
Figure 18 – Final Model Combination of both the model	50
Figure 19 – Accuracies and losses of our model using differently composed datasets	57
Figure 20 – Losses and accuracies with different preprocessing settings	60
Figure 21 – Losses and accuracies using different dropout rates	64
Figure 22 – Losses and accuracies using different batch sizes.....	68
Figure 23 – Losses and accuracies using DCT-residuals.....	70
Figure 24 – ROC Curves comparison between our best model and others	71

LIST OF TABLES

Table 1 – Summary of notable deepfake tools[49]	30
Table 2 – The training times of the model on differently composed datasets	58
Table 3 – Performance on DFDC using different preprocessing settings	61
Table 4 – The training times with different preprocessing settings.....	61
Table 5 – Performance using different dropout rates.....	65
Table 6 – The training times using different dropout rates.....	65
Table 7 – Performance using different batch sizes	68
Table 8 – The training times using different batch sizes	69
Table 9 – Performance using DCT-residuals	71
Table 10 – reports the accuracies, sensitivities, specificities, precisions, fall-outs, miss rates, and F1-measures.....	71

INTRODUCTION

The proliferation of cellphones has brought about a fundamental shift in the manner that we communicate with one another. These gadgets have permitted an unparalleled amount of digital content production, such as the taking of photographs and the recording of films. This digital information is currently regarded to be the gold standard for storing our memories, participating in social activities, and showing signs of our past. Because images and videos are so important, and because they play such a significant part in both social networks and the media that covers news, many. This program was built so that these digital materials might be edited and improved. However, there is a cost associated with using this new software innovation. Doctors of photography and videography Common behavior adopted for purely aesthetic or malevolent purposes. The proliferation of this fake digital media inhibits members of the general public from taking at face value any and all types of photographs and videos as a form of digital evidence. This mistrust is further fostered by the phenomena of "fake news" and the recent growth of methods based on machine learning techniques, such as generative adversarial networks, which may be exploited by malicious actors. Both of these factors contribute to the spread of these approaches.

In our rapidly digitalizing world, there is a rising need for a virtual assurance with respect to the identities of the people with whom we communicate online. This is a need that is only going to increase in the coming years. Deepfake movies have the potential to put this assurance at risk. Using a Deepfake rule set, it is possible to produce real-time video manipulations in which the appearance of a source character is altered by the appearance of a target character. This replacement might be accomplished by the use of face swapping, facial reenactment, or the production of a whole new video featuring the figure that is the focus of the investigation. The videos that followed are an eclectic mix, ranging from fake sexual movies to fake political lectures. Even if the quality of Deepfakes isn't always consistent, particularly since not when the video is dynamic, the need for Deepfake detection is an ever-present one.

The identification of Deepfake videos may be seen as a problem of binary classification owing to the fact that each image is either "real" or "faux." Convolutional neural networks provide a potential solution to problems of this kind using binary image classes. Convolutional networks teach themselves which filters work best for the extraction of relevant styles from digital photographs. They evaluate whether or not an image is real or fake based on the characteristics collected from it using these sorts of filters. In contrast to networks that just use feedforward, this enables them to find images regardless of their size or translation. For our implementations, we rely on the Python package Keras for its extensive experience in deep learning.

- Purpose

The goal of this thesis is to conduct an investigation into the feasibility and usefulness of using deep integration from facial recognition networks as a foundation for the identification of the faces that are traded in the video. In addition, the thesis intends to provide a solution to the question of whether or not face overlays include important information since they are studied over time. In addition, the thesis will provide a method for incorporating postural information into an exhaustive article detector.

- Motivation

There are many different ways that may be used to manipulate a person's face, such as procedures that can change a person's facial expression, hairdo, or perceived age. On the other hand, the approaches that were used in this investigation, which are often referred to as deepfake and face swap, are not one of them. They vary from the other practitioners in that they use face replacement rather than facial manipulation in their practice. They employ footage of two distinct identities with the intention of replacing the identity of the source person with the identity of a target person by switching the full face. They are able to achieve this while preserving other features such as head posture and face expression. Because of this,

the use of face identity embeddings in a classifier designed for this kind of data is warranted. The logic behind this is because if a deepfake approach is used to change a person's identity, then it is possible that it may leave traces and artifacts in these characteristics. Modern detection techniques often make use of handmade low-level features, generic pre-trained object identification embeddings, or are directly data driven by CNNs; nevertheless, they only rarely make use of recognition networks that have been explicitly trained for face identities. Because of this, it is necessary to conduct an analysis of embeddings that is based on identity separation in order to further improve our knowledge of deepfake detection.

A great number of techniques for replacing faces function on a frame-by-frame basis, meaning that they change the faces in each individual frame of a movie separately from the faces in the other frames. Because deepfake algorithms do not take time awareness into account, it is necessary for a detector to include some element of temporal analysis. Because of this, it is not unreasonable to think that a classifier that is based on identity embeddings may become more accurate with the addition of a temporal metric. These aspects are investigated further within the scope of this research.

It's possible that the movement of the subject's head in a video might be associated with the temporal shifts in the identity embeddings. In order to achieve additional progress in the development of a detector based on a temporal analysis, it is thus also motivated to integrate information regarding the head attitude. Because of this, it would be easy to differentiate between the temporal artifacts brought on by the switched face and the temporal fluctuation brought on by the movement of the head.

- Our contribution

Those who are not acquainted with this research but who wish to safeguard their goods and processes from being threatened by deep fake might use this study as a starting stone. This article provides an overview of the most recent improvements that have been made to deep learning approaches that are used for the

identification of deep fake in movies and photos. In addition to that, it describes the architecture and the structure of the face switching process. plus, also, one of the most important distinctions and contributions made by this book are broken out in great depth. Examine the dataset that was used for the discovery activity. The following is a list of the works: -

- It provides an updated and comprehensive review of the most recent works toward deepfake creation and detection;
- It exposes the most recent advances, main challenges, and tendencies of the field;
- It presents a detailed description of the most recent and popular architectures and frameworks for deepfake creation;
- Presented with the model capable of detecting deepfake;
- Testing the effect of different hyperparameters on the proposed model.

After that, we put the newfound information to use by applying it to a methodical search for the optimal mix of datasets, model architecture, and hyperparameters for identifying Deepfake movies using convolutional neural networks. Experiments are run regarding the composition of datasets, preprocessing settings, dropout rates, batch sizes, and DCT-residual. We integrate the convolutional bodies of both networks into a single one. In the last step, we examine the generality and resilience of the networks.

● Goal

The purpose of this study is to provide an overview of the progress that has been made in the research relating to deepfake detection approaches. In the context of this body of work, we shall cover, among other intelligent systems, approaches for the identification of deep fakes that are based on machine learning. The main goal of this review is to give knowledge of the most recent literature and to make that information accessible.

Researchers that are working on image recognition and security challenges will have access to a new resource thanks to this. In addition to this, it offers a precise

comprehension of the possible difficulties that the most current study and the most recent research recommendations may provide.

1 RELATED WORK (DEEPFAKE OVERVIEW)

Video manipulation is a broad term that can include several techniques. We want to distinguish between:

- video generation, which is the creating of fully synthesized videos from scratch,
- video manipulation, which is the altering of existing video content,
- video translation, which is the altering of video style.

Later on in this chapter, we will go through the several methods of video production and modification that are available. Detecting video manipulation is a job that requires binary classification: we input a picture into a neural network, and the network informs us whether it expects the image to be in the "real" or "false" category.

If we ignore the sound for a while, the videos that we're looking at are really simply sequences of still pictures. Computer vision refers to a set of activities that need computers to analyze real-world pictures and the spatial association between them. Two examples of computer vision tasks are image modification and image detection. Nevertheless, when attempting to make, alter, or detect videos and not pictures, there are significant distinctions between the two types of media that must be taken into consideration:

- Videos are compressed more than images, and compression can hide traces of violations.
- Because video is a time-limited series of pictures, it is not necessary to merely project a sequence of images in order to make the actual video; this is true regardless of how realistic the sequence of images may be. between successive frames is at least as significant as the quality of the individual frames. between successive frames is at least as important as.

Image and video image are going to be treated identically throughout this article. It should be obvious from the setting whether the word "image" refers to a picture that is shown on its own or to a frame from a video.

Throughout the whole of this study, we will be concentrating on detecting and manipulating video. On the other hand, in order to avoid difficulties that aren't strictly essential, we will sometimes discuss picture alteration and how to spot it. This is due to the fact that the algorithms used to manipulate video need to make changes to the video frame by frame; hence, if we understand how pictures are manipulated, we also understand how videos are changed. In a similar vein, the development of algorithms requires the analysis of video frames by frames. This suggests that if we understand how to process pictures that have been recognized, we may also comprehend how to recognize modified video. When the time constraints are in effect, we are going to have a conversation about video manipulation and how to spot it.

The next portion of this chapter will begin with describing how to set up a picture such that it may be processed by a neural network. This section first discusses the process of creating deep fakes and the various kinds of neural networks that are used for video manipulation. Next, deep fake tools are discussed, and then the discussion moves on to deep fake datasets. Finally, a comprehensive overview of the various kinds of detection strategies is presented.

1.1 Pre-processing

The amount of data that is readily accessible is quite low in comparison to that of other subfields within computer vision, which is one of the factors that contributes to the difficulty of detecting deep fakes. There are now more datasets available, which are also becoming bigger and more varied. These datasets are used for generic computer vision applications. However, even if the availability of data is not a concern, there is still a significant amount of memory space required for the storage of data. The use of preprocessing is one strategy that may be used to remedy this issue. The process of creating new photos from already existing images in a data

collection by applying a variety of various processes is known as preprocessing. Throughout the whole of this investigation, we use the preprocessing procedures of zooming, rotating, cropping, flipping horizontally, and offsetting vertically and horizontally. Figure 1 provides a graphic representation of these processes.



Figure 1 – Examples of preprocessing operations. From left to right: the original image, a rotated image, a horizontally flipped image, a sheared image, a horizontally shifted image, and a zoomed image

1.2 Deepfake Creation

When it comes to changing people's faces in films, there are primarily three distinct kinds of algorithms. The graph-based method [1, 2] is the simplest one, and it's the one that's most often seen in lightweight mobile apps like Snapchat. The second kind of algorithm makes use of latent feature spaces in order to differentiate between individuals' faces and their identities. These algorithms are believed to be of a higher level of sophistication and need for goal-specific training using automated codes [2, 3, 4, 5]. Last but not least, there are other approaches that are based on GAN [2, 6, 7], albeit they are often used for producing single pictures rather than films.

There are a lot of different open-source implementations, each of which uses a variant of these three fundamental techniques. These are the most prevalent kind of implementations; the general public can readily access it, and it comes with graphical user interface software that is simple to use. Implementations such as DeepFaceLab [3], deepfake [4], FaceSwap [1], and others like them are among the most popular on the Internet due to the fact that they are easy to use and readily available. There are further approaches that are more cutting-edge and complex, such as FSGAN [19], but these methods do not have a straightforward user interface.

1.2.1 Graphics-based methods

Because graph-based approaches don't make use of any deep learning in their algorithmic design, they are sometimes referred to by the name faceswap rather than deepfake. These approaches provide results that are seldom convincing and often include face deformation artifacts, as seen in the picture below: -



Figure 2 – Example of face swapping with a graphics-based method [1]. Images from [8] (Left: source image, Middle: swapped identity, right: target image)

A typical method incorporates a face-adaptive 3D model, as well as facial characteristics extracted from both the source picture and the target image. The landmarks are used as anchor points in order to first match the source surface to the target texture, and then to finally match the model to the target surface.

The graphics-based technique [1] is included into the dataset via the use of Face Forensics ++ [8]. When cropping the face region from the source video and the target video, a face detector that is based on HOG characteristics should be used. The facial stencil model may then be matched to the original facial texture by using the face markers that are located in the region that was cropped. After that, the model is projected onto the target surface in such a way that the distance between the projection and the target marking is reduced to its smallest possible value. In the last step, the produced face model is superimposed over the target picture, and color correction is made. This is done in order to erase any noticeable borders between the swapped region and the original face. The application of this takes place on a frame-by-frame basis.

1.2.2 Encoder-decoder methods

Encoder-decoder-based algorithms are built largely based on the idea that it is vital to separate face identity from facial expression while developing appealing

deepfake video. This job is a crucial step in the process. The facial expression of the subject is attempted to be extracted from the target picture and superimposed onto the identity of the person using these approaches. They make use of an autoencoder architecture, in which the encoder is universal to all identities but the decoder is particular to the target. The objective is to locate a probable feature space that, when run through the encoder, provides a generic description of the facial characteristics. The decoder is designed to reconstruct the picture based on the ID using this functional space as a starting point.

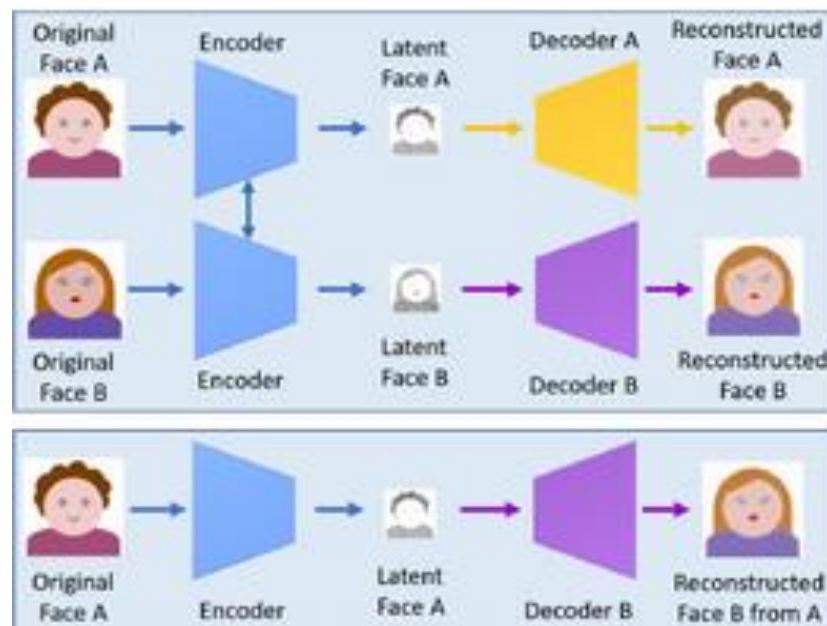


Figure 3 – Training process of the encoder-decoder method (face images from [5, 9])

The instruction procedure for the fundamental method is shown in the top portion of Figure 3. Unsupervised optimization of the L1 loss between the original picture and the replicated image is performed by the pair of encoder and decoder components. During training, each ID has access to the same encoder, but the decoder is tailored to a particular target. This makes it possible for the encoder to acquire the knowledge necessary to encode common attributes that are shared by all identities. This might be anything like a face expression, the orientation of the image, the lighting, or the shading. On the other hand, the decoder is taught how to translate general attributes to an image that has a specific ID.

The bottom portion of Figure 3 demonstrates how to construct switched faces by using an encoder and decoder that have been trained. First, the encoder is given the picture to be used as a target in order to extract identity-independent features. A decoder that has been trained for the source ID is used rather than one that has been taught for the same ID in order to avoid replicating the picture. In other words, it mixes the characteristics that are general to the target picture with the characteristics that are unique to the source image.

It is important to stress that the training and generation processes are only carried out on the facial parts that have been cropped out of the picture rather than the whole thing. Because of this, it is guaranteed that the programmer will automatically learn the face-related qualities in a stringent manner. In order to construct the whole picture, the regenerated faces are blended into the face region of the target image, and then contour smoothing is used to disguise the border where the two images blend together. Figure 4 is an example that may be derived from the information in [4].



Figure 4 – Example of face swapping with an encoder-decoder method [3]. Images from [8] (Left: source image, Middle: swapped identity, right: target image)

1.2.3 Generative Adversarial Networks(GAN)

Two separate neural networks make up generative adversarial networks, often known as GANs. Although autoencoders and VAEs are likewise made up of two neural networks, as shown in Figure 5, there is no comparison between the two types of neural networks.

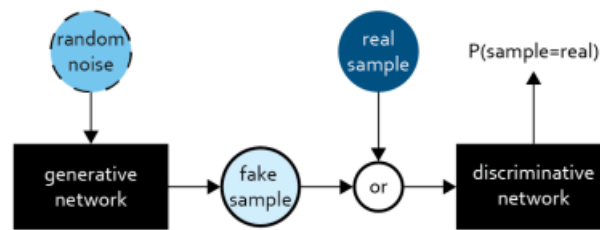


Figure 5 – A generative adversarial network

GANs are made up of two different types of networks: a generating network and a discriminative network. These two networks compete against one another in a game. The random noise, denoted by $x = Rn$, is taken in by the generative network, which then makes an effort to convert this into a particular variable. This variable would be a picture that belongs to a certain domain if we were talking about image alteration. Either the variable that was created by the generative network or an actual variable from the target domain is sent as an input to the discriminative network. The task of the network's discriminator is to identify which network was successfully received. It achieves this by returning a number that is somewhere between 0 and 1, where 1 indicates that there is an absolute certainty that the real sample was received, and 0 indicates that there is an absolute certainty that the produced template has been received. The discriminator network needs to be thrown into as much disarray as is humanly feasible so that it perpetually produces a value of 0.5.

1.3 Deepfake Tools

This section provides an overview of the most current and widely used architectures and frameworks that are used for the production of deepfake material.

1.3.1 FaceSwap

The word "faceswap" refers to any instance in which the face of a person seen in an image or video is replaced with that of another person. However, the term may also be used to describe more specific instances. In this context, though, it refers to a specific method for accomplishing this goal.

A group of students from the Warsaw University of Technology developed an application called FaceSwap using the programming language python. Face

swapping is accomplished by the software via the use of picture blending, Gauss Newton optimization, and face alignment based on a deep alignment network [10]. The first thing that the algorithm does with a given input picture is search for face regions and landmarks. Additionally, three-dimensional models are composed of waypoints, the vertices of which are projected into picture space to form texture coordinates. After this stage of the algorithm is finished, the algorithm continues on via the following five key steps:

- To identify the face and localize several landmarks on the surface of the face.
- Place the 3D model inside of certain landmarks in the area.
- So that the 3D model may be rendered.
- To produce photos from the camera as well as images that have been created using color correction and alpha mixing.
- Demonstrate the completed picture

The algorithm does not need any training, and it does not need to have previously seen either of the agents' faces. Additionally, it does not require a significant amount of processing power (depending on the face recognition algorithm). However, it has problems when there is a significant variation in the lighting between the two movies and has a substantial amount of reliance on the performance of the face identification algorithm.

1.3.2 Face2Face

One of the most well-known techniques for the reconstruction of faces is called Face2Face [11]. The primary purpose of the algorithm is to construct a three-dimensional model of the source and target actors' faces and then monitor the changes that occur in those faces. Build models for the original actors, then include those models into your target actors' corresponding representations. This does not occur throughout the learning process. Instead, you should make use of Principal Component Analysis (PCA) in order to monitor surface deformation and parameterize it. To narrow the focus of the challenge down to optimization, these

parameters are entered into the goal function. The issue was addressed using the iteratively reweighted least squares (IRLS) method. After the representation has been provided, the interior of the mouth is recorded by searching the video of the target actor and finding the optimum match with the current frame. This process is repeated until the mouth is completely mapped out. Because of this, it is essential to have the video of the target actor in advance. Nevertheless, you may utilize the source video to work with the video while it is being played in real time. Please refer to Figure 6 for an explanation of the method.

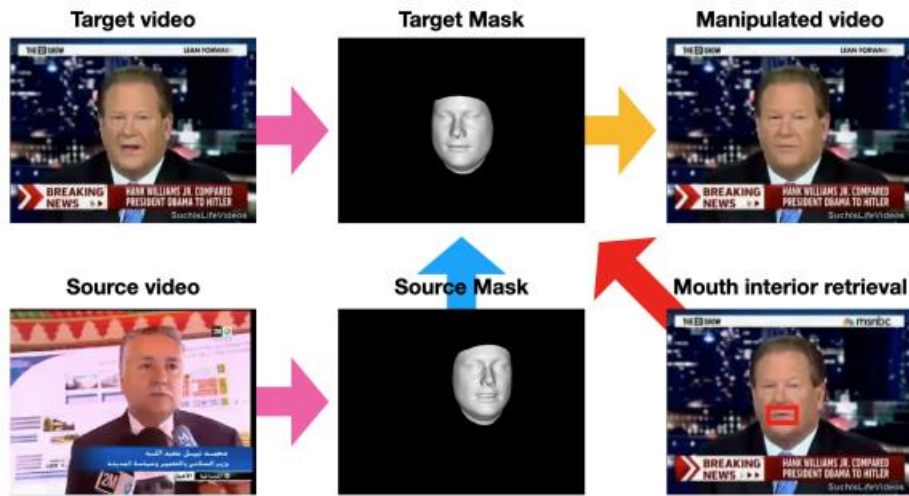


Figure 6 – Face2Face example

The operation of Face2Face is shown in figure 6, in which 3D representations of the source and target actors' faces are built (pink arrows). After that, the model of the target actor's face is warped for each frame to match the expressions of the source actor (blue), and after that, the mouth interior from the target movie that best fits the new facial expression is obtained (red). At last, the altered 3D face is composited back into the original footage (yellow).

1.3.3 Neural Textures

A texture map is a map that holds information on 3D objects, such as the colors and textures of the objects, their albedos, and any tiny surface displacements. This kind of map is used in computer graphics. The neural texture is a technique that takes a picture of the thing as its input and replaces it with a learnt version of the unique characteristics of the object. It was first introduced by Thies et al. [26]. Using

this UV map, which relates the neural textures, perspectives, and components in the neural texture map to points in the object, you are able to build textures that are particular to a certain viewpoint. By establishing a connection between this and a trained delayed neural renderer equipped with a neural texture, it is possible to produce a picture.

Because of this, neural textures may be used for purposes other than the alteration of face videos, such as Thies et al. Kindly include this as one of its potential applications. Figure 6 illustrates this point further. In this approach, much to the Face2Face technique, a training film is used in order to generate a 3D model of the face of the target actor. Having said that, in addition to that, it generates an individual-specific neural texture as well as an individual-specific delay neural renderer for the target actor. A UVMap is generated from the target footage and then changed to match the expression of the source cast member. This process, known as expression mapping, is similar to Face2Face and uses the same approach. To create a movie with altered content, you may make use of this UVmap, neural texture, and neural renderer.

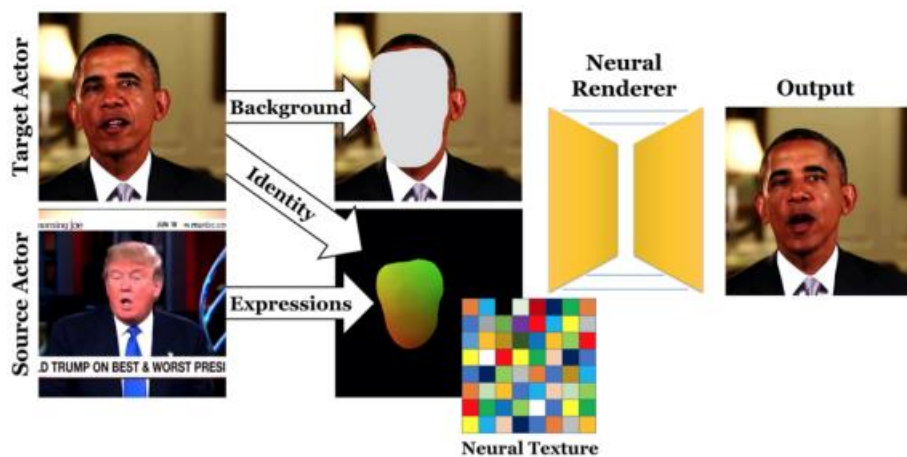


Figure 7 – Expression modification using neural textures [26]

As seen in the previous figure, On the actor that will serve as the target, a person-specific neural texture and neural renderer are trained. After that, a UVmap is created that matches the target actor's face while maintaining the facial emotions of the source actor. The altered video is obtained by sampling a texture from each of

them and then feeding that sample into the neural renderer. This process produces the final product. Image taken from page [26].

Table 1 – Summary of notable deepfake tools [49]

Tools	Links	Key Features
Faceswap	https://github.com/deepfakes/faceswap	-Use two encode-decode pairs. Install a shared encoder.
Faceswap-GAN	https://github.com/shaoanlu/faceswap-GAN	Encoder loss and perceptual loss (VGGface) added to autoencoder architecture.
Few-Shot Face Translation	https://github.com/shaoanlu/fewshot-facetranslation-GAN	Use a pre-trained face recognition model to extract latent embeddings for GAN processing. Combining semantic elements obtained by the FUNIT [12] and SPADE [13] modules
DeepFaceLab	https://github.com/iperov/DeepFaceLab	Extend the Faceswap method with new models, e.g., H64, H128, LIAEF128, SAE [14]. Supports multiple face extraction modes, e.g., S3FD, MTCNN, dlib or manual [14].
DFaker	https://github.com/dfaker/df	The DSSIM loss function [15] is used for face reconstruction. Keras. library based implementation
Deepfake tf	https://github.com/StromWine/DeepFake tf	Similar to DFaker but implemented based on tensorflow.
AvatarMe	https://github.com/lattas/AvatarMe	3D face reconstruction from arbitrary "in the wild" images.

		True 3D facial reconstruction at 4K x 6K resolution is possible from a single low-resolution image [16].
MarioNETte	https://hyperconnect.github.io/MarioNETte	Some scenes recreate the face to preserve the identity of the target. No further debugging is required for identity tuning [17].
DiscoFaceGAN	https://github.com/microsoft/DiscoFaceGAN	Create an image of a virtual person's face with latent variables independent of recognition, expression, posture, and lighting. Integrating 3D facilities into adversarial learning [18].
StyleRig	https://gvv.mpi-inf.mpg.de/projects/StyleRig	Create face portraits with tight control on a fixed StyleGAN, pre-trained through 3D blendable face models. Self-monitoring without manual annotation [19]
FaceShifter	https://lingzhili.com/FaceShifterPage	Swap faces with high fidelity by leveraging and integrating target attributes. Can be applied to any pair of new faces without specific training [20].
FSGAN	https://github.com/YuvalNirkin/fsgan	A face swap and build model can be applied to pairs of faces without requiring training on those faces. Adjustments for varying poses and expressions [21]

Transformable Bottleneck Networks	https://github.com/kyleolsz/TB-Networks	The present invention relates to a method for fine 3D manipulation of the content of an image. Applying spatial transformations in CNN models using a transformable bottleneck framework [22].
“Do as I Do” Motion Transfer	github.com/carolineec/EverybodyDanceNow	Automatically transfer motion from source to target by learning how to translate video to video. It is possible to create synchronized dance videos with multiple objects [23].
Neural Voice Puppetry	https://justusthies.github.io/posts/neuralvoice-puppetry	An audio-driven face video synthesis method is provided. Composite video of a talking head from other people's sounds using 3D facial representation. [24].

1.4 Dataset

The dataset for the Facebook Deepfake Detection Challenge13 (DFDC) [25] includes 5,000 films of actors with visual resemblance that have been altered. The dataset includes 66 actors, of whom 20 percent are American and 68 percent are White. Three percent of the actors are from South Asia, nine percent are from West Asia, three percent are from South Asia, and three percent are from West Asia. The procedure was carried out keeping in mind two different ways to conduct a face exchange. Method A, which produces a high-quality swapped picture on the face that is closer to the camera, and Method B, which produces a lesser quality exchange due to the fact that it does not regard the source face and the swapped face in the same proportions. In the end, we were able to compile a dataset that included 4,464 sample clips for training and 780 sample clips for testing. Each clip was 15 seconds

long and had a distinct resolution. Examples of datasets are shown below in Figure 8.



Figure 8 – Examples from DFDC dataset. Adapted from [25]

1.5 Deepfake Detection Methods Review (Literature review)

In this section, a literature review of the most recent works that comprise deep studying-primarily based strategies for deepfake detection is presented. The works that are reviewed include: Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs), Autoencoders, and Recurrent Neural Networks, as well as other approaches that make use of a physiological signal, are the methods that we use to classify the works that have been done in the field of deep learning in order to achieve a higher level of comprehension.

1.5.1 Convolutional Neural Networks

The author of study [28] recommended use a pre-trained version of ResNet50 in order to identify computer-generated pictures. The top layer has been replaced with a fully linked layer, and the classification was done using a support vector machine (SVM). Instead, the scientists stated that the SVM classifier attained an average accuracy of 94.05 compared to the 92.28 percent detection rate produced by the SoftMax layer.

In a similar manner, the author of the study [29] used AlexNet and VGG16 in order to extract characteristics from the face and locate indicators of bogus information. The suggested method bypasses the fine-tuning step by importing learned weights using transmission learning, substituting dense layers with SVMs, and classifying faces in a manner that may or may not be accurate. In addition, the scientists advised merging characteristics gathered from both networks in order to increase the accuracy of predictions and provide more information. The built-in

functions demonstrated the highest level of model performance, with an accuracy of 94.01 percent for the CASIA dataset.

Mo and his cohorts in crime. [30] suggests use a simple CNN model that is composed of three sets of convolutional and maximum convolutional layers in order to identify fraudulent faces. These spatial high-pass filters, which execute spatial operations to bring out minute features in the picture and hence magnify the image noise, were utilized by the authors. The suggested CNN architecture takes its input features from the residual noise that was previously collected. The authors reported an accuracy of 99.4 percent in authenticating photos taken from the CELEBAHQ dataset after the images were supplemented with synthetic faces produced by a GAN-based methodology [31].

In the meanwhile, Khodabakhsh [32] pointed out that some deep learning models are able to cope with the challenge of recognizing fraudulent faces in photos taken from YouTube videos. The purpose of the proposed research is to determine whether or not the model can be extended to private datasets. As a result, the author used a dataset consisting of 53,000 photos taken from 150 YouTube videos that were associated with fake faces that were created using CGI (Computer-Generated Imagery) and modification tools such as FakeApp and Face Replacement. The author made use of a number of well-known CNN architectures, including AlexNet, VGG19, ResNet, Xception, and Inception, all of which were trained using Imagenet datasets. In spite of the high accuracy obtained from the Imagenet 8 test images, the test images included in the newly proposed dataset significantly reduce the effectiveness of the model, making it difficult to predict newly introduced artifacts. This is because the test images in the newly proposed dataset are significantly more complex. is becoming clear. Ivanovetal's [33] research focuses on the classification of false material and provides a deep learning and super-resolution algorithm-based technique for identifying deepfake based on incompatibilities between distinct sections of the face and the head position of the subject.

1.5.2 Generative Adversarial Network

Mara et al. [34] conducted research on image-to-image conversion problems and compared a set of deep learning algorithms to the original image generated by the generative hostile network architecture as well as the corresponding fake image pair. This research was published in the journal *Computer Vision and Image Understanding*. The primary objective of this research is to analyze the effectiveness of web analytics and deep learning architectures in the identification of image-to-image translation. In addition, the author used a Twitter-based compression algorithm to evaluate the effectiveness of the model when it came to compressing photographs, specifically before uploading them to social networking sites. The research shown that deep learning models, particularly the XceptionNet architecture, are capable of achieving a high level of accuracy in the detection of false pictures for compression situations. This finding lends credence to the resilience and feasibility of such models.

Hsuetal. [35] prevented the CNN from making erroneous detections by adjusting the contrast loss function. The scientists advised using a combination of characteristics collected from actual and false pictures for future predictions made by fully connected layers that were coupled to the feature extraction network. The input picture pairs were subjected to contrast loss in order to ascertain whether or not they were comparable. In the context of the identification of fraudulent images, contrast loss may give the assistance necessary to discover aspects linked to image manipulation by comparing the actual qualities of an actual picture. The proposed deep learning model is able to handle bogus spots in the feature representation of the picture after the training has been completed. This results in great performance even with fake images created by the five different GAN architectures. The authors reported an average accuracy and recall of 0.88% and 0.87%, respectively, for their work. Later, the same author [36] expanded his previous work to identify fraudulent photos created by a computer in an effective and efficient manner.

In addition, Korshunov and Marcel [37] shown that the most cutting-edge face recognition systems based on deep learning are susceptible to being tricked by deepfake video. The author gave considerable consideration to several fundamental strategies after reviewing the datasets that were supplied. According to the findings of H. VidTIMIT, the strategy that performed the best was one that was based on the visual quality criteria that are often used to identify presentation assaults. It also demonstrates that typical face recognition algorithms have a difficult time detecting deepfake videos since they are created using GANs. In addition to this, they claim that the development of new deepfake technology has a tendency to make situations worse.

On the other hand, Yang et al. [38] created a technique to inject a synthetic face region into the original picture by using generative neural networks. They also employed a 3D model to predict the different head postures that indicated the inaccuracy. Introduced. A model of this kind generates a collection of characteristics that are then fed into the SVM classifier for further processing. Find the differences between actual and false pictures. In their paper [38], Frank and colleagues suggested research that would examine pictures produced by GAN in the frequency domain. The experimental findings have shown that the existing GAN architecture has some major artifacts that are induced by processes known as up sampling. This exemplifies the structural and fundamental issues that are present in the GAN imaging technique.

After separating the deepfake from the original video, I was able to identify the generator model that was used to create the deepfake. This finding is in line with research that has been done in the past. According to the findings of this study, the generator residue may include information that may be used to differentiate between the controlled artifact and the biological signal. For this investigation, a total of 32 raw photoplethysmogram (PPG) signals were collected from a variety of face postures. These signals are organized into spatiotemporal blocks, each of which has a certain spectral density. H. PPG cell. PPG cells are input into commercial neural

networks with the intention of identifying a variety of signatures provided by the source model.

1.5.3 Autoencoders

In their paper [40], Maksutov et al. offer a technique to identify deepfake video by taking into account an artificial data set that was produced using a GAN and an autoencoder. The method computes facial characteristics with the help of an encoder and then classifies these features with the help of a decoder and a CNN. This results in values of AUC (area under curve) and accuracy that are satisfactory.

In the framework of single-layer anomaly detection, Khalid and Woo [41] introduced a VAE (Variational Autoencoder) architecture for predicting fraudulent face pictures. The so-called OC-FakeDect is trained only on actual face photos, which are then used to predict which unseen fake face images are anomalous. This is done rather than utilizing a conventional binary classification job. In order to do further comparisons with the encoder latent space of the picture, the authors presented a second version of the VAE architecture as a supplementary contribution. This version of the design incorporates an encoder layer following the image reconstruction layer. d input. According to the findings of the research, the OC-FakeDect model performed much better than the current Xception CNN architecture on the DFD dataset when it came to the binary classification test.

1.5.4 Recurrent Neural Networks

CNNs were used to extract image-level features, and then the data was fed into recurrent neural networks, which were then used to classify the films. Guera and Delp [42] suggested a time-sensitive system that could automatically identify deepfake videos. This approach was tested against a huge number of deepfake movies gathered from a variety of sources, and it achieved results that were comparable to those of the competition for the challenge.

In a similar vein, Li et al. [43] detailed a technique for uncovering phony face films that were produced by deep neural network models. This technique can identify blinks in videos, which are often ignored by software used to create phony

videos. The technique of differentiating between open and closed eye states by merging complicated neural networks with long-term repeating convolutional neural networks, also known as LRCNs.

1.5.5 Unnatural eye movement

In this piece of work, Arthur outlines a process for rendering DeepFake video that involves identifying the lack of blinks from synthetic faces. It is difficult to imitate the act of blinking in a manner that seems natural to the observer. In addition to this, it might be difficult to capture situations as they seem in the eyes of a genuine person. Within the blink of an eye, the suggested technique is based on a novel deep learning model that blends a complex neural network (CNN) with a recursive neural network (RNN) to capture the rules of phenomena and time. The most recent techniques make use of a convolutional neural network (CNN) as a binary classifier to differentiate between open and closed eye states inside each frame. CNN, on the other hand, is only able to make predictions based on a single picture, therefore it cannot benefit from information in the temporal domain. The method that makes use of long-term repetitive accumulation neural networks (LRCNs) [44] differentiates between open and closed eye states by taking into account past time information. This is necessary due to the fact that human blinks have a high temporal association with previous states.

1.5.6 Detecting Face Warping Artifacts

In this study [45], the author presents a novel technique based on deep learning that can easily differentiate AI-generated fake movies from true videos. These movies are referred to as DeepFake films due to the author's designation of them as such. This tactic is predicated on the fact that the present iteration of the DeepFake algorithm can only produce a picture of poor quality. This image then has to be further twisted in order to be made to seem like the actual face that was captured in the source video. We demonstrate that such changes leave distinct artifacts in the final DeepFake videos, which can be captured fast using a powerful neural network. These artifacts may be seen in the video below (CNN). In contrast to other

techniques, which relied on a large number of actual and DeepFake-generated photos to train CNN classifiers, this method does not use DeepFake-generated images as negative training examples. Instead, it uses only genuine and real-world images. This is due to the fact that I am focusing on artifacts in face distortion association for feature discrimination in order to differentiate between actual and phony photographs.

1.5.7 MesoNet

[46] The authors describe the rationale for using the strategy that they developed on a mesoscopic level in their discussion. The reason behind this is that a microscopic technique is impossible since the picture noise grows worse in a compressed video, which is the reason why this cannot be done. Because of this, an analysis based on the picture noise cannot be performed. They [46] suggest two different approaches. These techniques are based on models that are successful when used to the categorization of images. Meso-4 is the name of the first technique that will be discussed. Convolution is one of its 4 layers that make up its structure. These make use of ReLU activation to bring in nonlinearity, which is then followed by Batch Normalization to ensure that the gradient effect does not disappear. Convolution blocks measuring 3 by 3 are used for the first two levels, while those measuring 5 by 5 are used for the latter two layers. The next layer, which is a single dense one, comes after these convolution blocks. When compared to the first approach, the second one, known as MesoInception-4, is superior since it employs an Inception module rather than the first two convolution blocks. In the final two iterations of the convolution process, the dilated convolution is performed using a 3 x 3 matrix rather than a 5 x 5 matrix. Because of this, there is a difficulty with the dimensions. In order to address this issue, the author [46] has included a 1 x 1 convolution block just before to the dilated convolution layers. The authors have included a further 1 x 1 convolution in parallel in order to bypass connections that are shared between two sequential models. These fast networks are evaluated using a dataset that has already been compiled as well as a dataset that was compiled by

us using videos found on the internet. According to the findings of the experiments, the detection rates of Deepfake and Face2Face are extremely high, coming in at over 98 percent and 95 percent, respectively.

1.5.8 Biological Signals

This article [47] introduces a method to detect fake, that detectors that blindly use deep learning are not very effective at detecting fake content, as the general patterns produce amazingly realistic results, and this article [47] also argues that these detectors are not very effective at detecting fake content. Because they are not geographically and temporally retained in the material, the major argument is that the bio-signals concealed in portrait films may be utilized as an implicit description of authenticity. This derives from the fact that the bio-signals are not present. fake. The following procedures were taken in this research in order to verify and validate this claim: executes numerous signal transformations for the bifurcation issue, obtaining an accuracy of 99.39 percent in the process.

- makes use of these findings in order to construct a generic classifier for bogus content by conducting an analysis of the suggested signal modifications and the feature sets that correspond to them.

- creates a brand-new signal map and makes use of CNN to boost the performance of our conventional classifier for the identification of fake material.

The results of applying this strategy to many datasets proved to be pretty intriguing due to the extremely high accuracy they provided. The best part is that this method generates significantly higher detection rates than the baseline and is independent of the source, generator, or attributes of the fake content. In addition, it analyzes signals from different face regions, under image distortion, with different segmentation durations, from different generators, based on unseen datasets, and according to some size reduction technique.

2 RESEARCH METHODOLOGY

2.1 Model

2.1.1 Model 1 Finding Unnatural Facial Expression

An LSTM-based artificial neural network is used in this method to handle the sequential temporal analysis of video frames. A pre-trained Res-Next CNN is then used to extract frame-level properties from the processed data. After the frame-level properties have been retrieved with the help of the ResNext Convolution neural network, they are utilized to train an artificial Recurrent Neural Network that is based on Long Short-Term Memory in order to determine whether or not the clip is a Deepfake or a genuine one.

2.1.1.1 Pre-processing

During this step, the movies will undergo preprocessing, during which any extraneous data and noise will be removed. The face is the only component of the video that is identified and excerpted since it is deemed to be essential.

The first thing that must be done in order to prepare the video is to cut it up into individual frames. Following the video has been cut up into individual frames, each frame is examined to locate the face, after which the frame is cropped to center on the face. In a later step, the frame that was chopped up is added to each individual frame of the video to form a new video. This technique is repeated for each video, which ultimately results in a dataset that only contains footage of faces after it has been processed. During the preprocessing stage, any frames that do not include the face will be ignored.

In order to maintain a constant number of frames, we decided on a threshold value based on the average number of total frames included in each movie. One further consideration when determining a threshold value is the amount of available processing capability. Processing all 300 frames at once in the experimental situation is computationally expensive since a movie with a duration of 10 seconds and a frame rate of 30 frames per second (fps) has 300 individual frames. Therefore, considering the computational capabilities of our Graphic Processing Unit (GPU) in the context of an experimental setting, we decided that the threshold number should be 150 frames. While we were storing the frames to the new dataset, we only preserved the first 150 frames of the film in the new video. In order to demonstrate the appropriate use of long-term and short-term memory, we did not randomly evaluate the frames but rather studied them in a sequential method, beginning with the first 150 frames (LSTM). The video that was just recently created has a resolution of 112 pixels on each side and a frame rate of 30 frames per second.

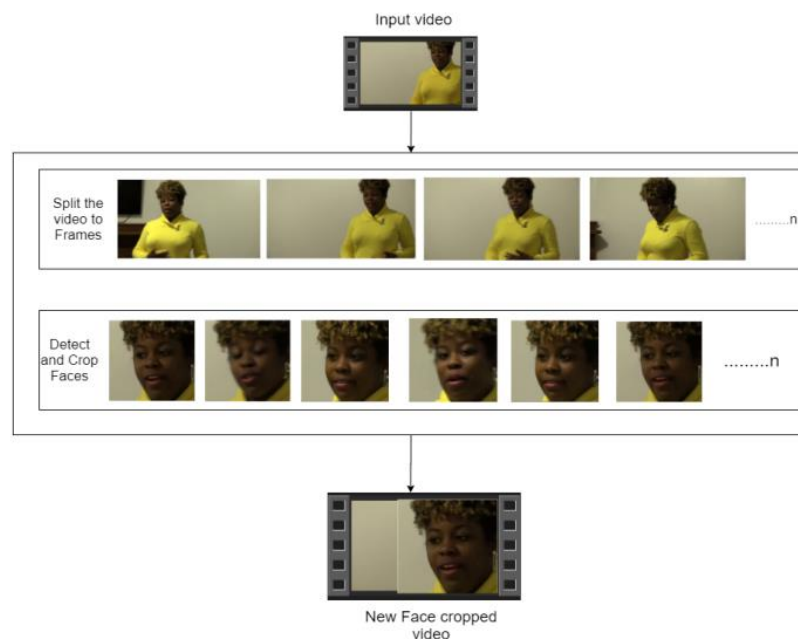


Figure 9 – Pre-processing of video

2.1.1.2 Model Architecture

The structure of our algorithm is a cross between a CNN and an RNN. We used a ResNext CNN model that had already been pre-trained to extract features at the frame level. After that, we trained an LSTM network to classify videos as either

deepfake or genuine based on the characteristics that were extracted from the videos. Utilizing the Data Loader on the training split of videos allows for the loading of the movie labels and their subsequent incorporation into the model for training.

- **ResNext :**

In order to extract features, we made use of the pre-trained model that ResNext provides. ResNext is a deep neural network Residual CNN network that was designed to achieve the highest possible level of performance. In order to carry out this experiment, the ResNext50 32x4d model was used. In order to complete this project, a ResNext that has 50 layers and 32 x 4 dimensions was used.

After that, we will finish fine-tuning the network by adding any extra layers that are required and establishing a realistic learning rate in order to make certain that the model's gradient descent correctly converges. The 2048-dimensional feature vectors that come after ResNext's final pooling layers are what the sequential LSTM takes in as its input.

stage	output	ResNeXt-50 (32×4d)
conv1	112×112	7×7, 64, stride 2
		3×3 max pool, stride 2
conv2	56×56	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128, C=32 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256, C=32 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4	14×14	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512, C=32 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1 \times 1, 1024 \\ 3 \times 3, 1024, C=32 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	global average pool 1000-d fc, softmax
# params.		25.0×10⁶

Figure 10 – ResNext Architecture

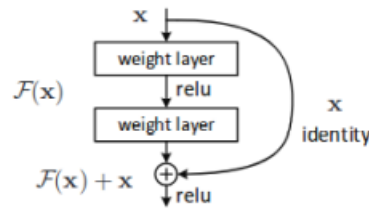


Figure 11 – ResNext Working [50]

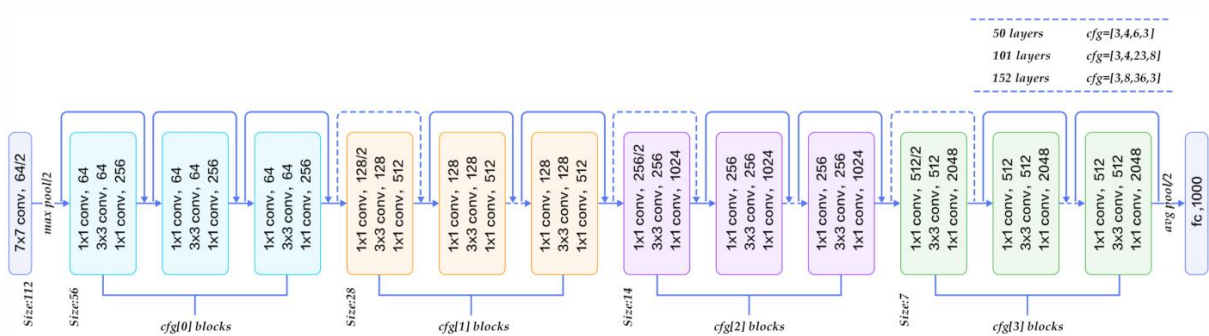


Figure 12 – Overview of ResNext Architecture [50]

- **Sequential Layer:** The Sequential class serves as a container for Modules, which may be piled on top of one another and run in parallel. The ResNext model provides a feature vector as an output, and this vector is saved in a sequential layer. in order for it to be gradually moved to the Long Short-Term Memory.
- **LSTM for Sequence Processing:** When doing analysis of sequences and searching for temporal differences between frames, the LSTM method is used. As input, the LSTM is given feature vectors that have a size of 2048. To accomplish what we set out to do, we are making use of a single LSTM layer that has a probability of dropout of 0.4 and contains 2048 latent dimensions in addition to 2048 hidden layers. The LSTM is used to do a sequential analysis of the frames in order to carry out a temporal analysis of the video. This is accomplished by contrasting the frame captured at time t with the frame captured at time t plus n seconds. Before the time marker t , n may represent any number of frames.

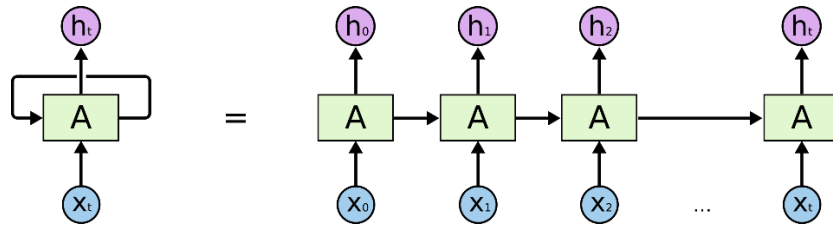


Figure 13 – Overview of LSTM Architecture[51]

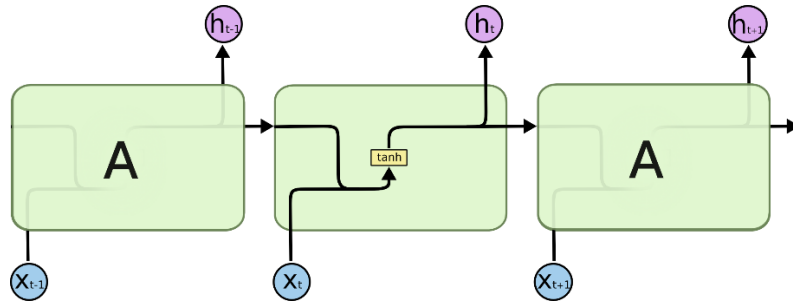


Figure 14 – Internal LSTM Architecture [51]

The model also accounts for the activation of Relu in a leaky fashion. The linear layer consists of 2048 input features and 2 output features, and the model uses this layer to learn the average rate of correlation that exists between the input and the output. The model makes use of an adaptive average pooling layer and has an output parameter value of 1. This provides the intended output size of the picture in the format of height x width. In order to process the frames in the correct sequence, a Sequential Layer is used. The batch training is carried out with a size of four participants each batch. During the process of prediction, a SoftMax layer is used to ascertain the level of confidence the model has.

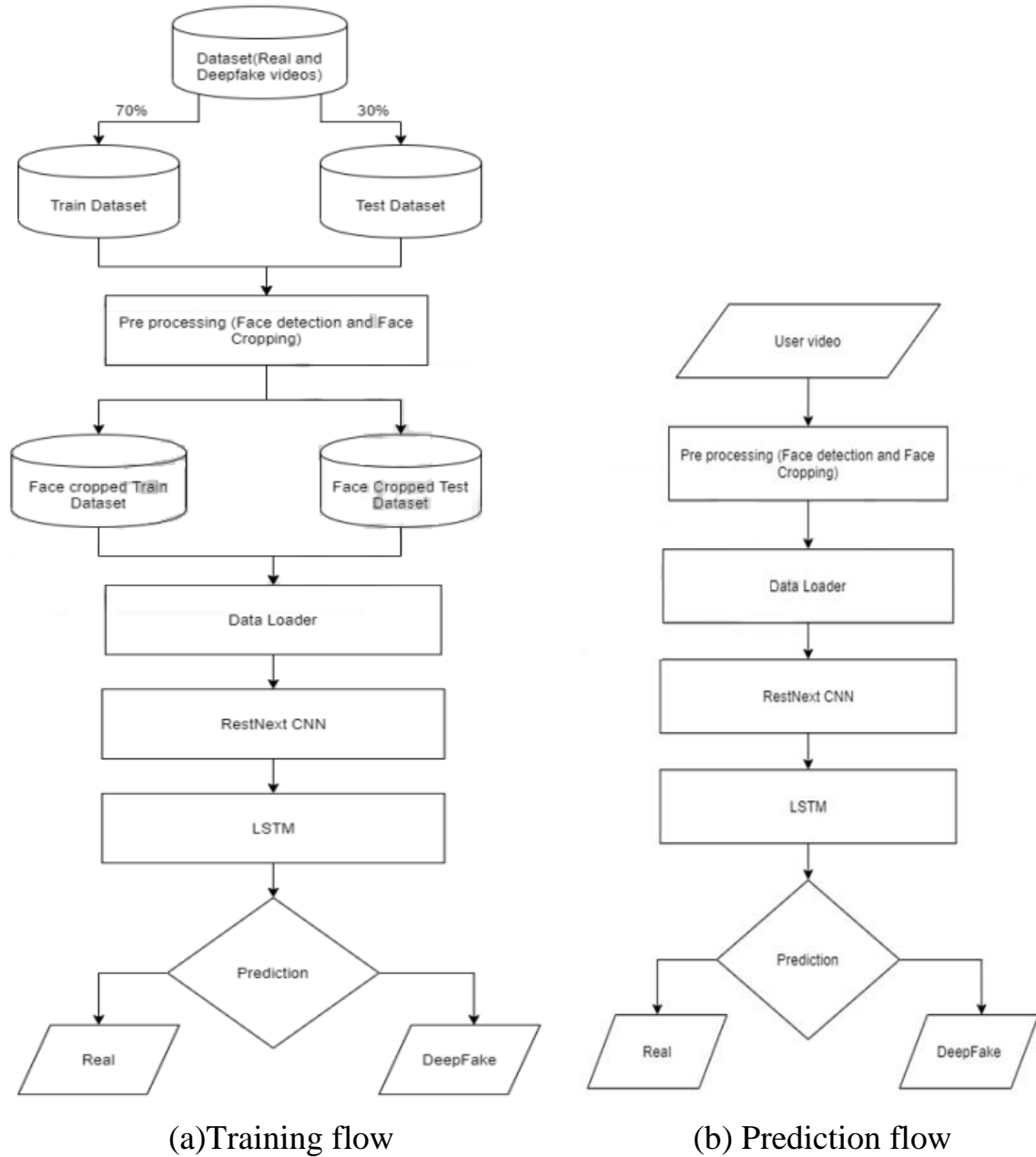


Figure 15 – Training and prediction Flow

2.1.2 Model 2 Audio Video synchronization

We calculated the aggregated dissimilarities between real and fake video over 1 second. This is done by calculating MDS (Modality Dissonance Score). Moreover, over model also learns discriminative auditory and visual features as these are entered separately. So, the model can distinguish even if the contrastive loss score is not calculated.

To do so we first take the video Dataset which contains v_i and y_i pairs. v_i indicates the input video while the y_i indicates is the label telling if the video is real

or fake i.e., $y_i = 0$ the video is real and the $y_i = 1$ the video is real. Next, we extract audio signals from v_i and split into multiple segments of 2 seconds each. This can be done using various libraries such as HandBrake, GStream etc. In this thesis we have used ffmpeg as it provides great flexibility. Similarly, we have performed the same splitting of video using MTCNN for face detection and cropping. This gives us two sets. One with the video frames that is represented by $\{V1_i, V2_i \dots, Vn_i\}$ and the other are corresponding audio segments related to the video frames which can be represented as $\{A1_i, A2_i, \dots, An_i\}$.

We pass these to a bi-stream network for video and audio. The video stream model is inspired by ResNet. The model has a dimension of $3 * h * w * 2 * f$, where 3 is the number of channels (RGB), h, w is the height the width, 2 is the segment length and f are the frame rate of the video. In the layer 8 the features learnt are used to calculate contrastive loss which is used to train the model.

For audio stream MFCC (Mel -frequency coefficient) are used as inputs. We decided to use 10 mel frequency band as it provided good results. We then convert this to heat-map. Figure 16 represents the architecture of audio and video stream.

There are two loss functions that we have used. Contrastive loss is used at layer 8 for both video and audio models and then Cross-Entropy loss is calculated for both the models and all these are weighted and added to get the final loss which is used to train the model. Equation (1) is used to represent the Contrastive loss function.

$$\frac{1}{N} \sum_{i=1}^N (y_i)(d_t^i)^2 + (1 - y_i) \max(Margin - d_t^i, 0)^2 \quad (1)$$

Where,

d_t^i is the dissimilarity score represented by Equation (4)

y^i is 0 or 1 depending on whether the video is real or fake

And N is the number of segments

The Cross-entropy loss is calculated using Equation (2) and Equation (3). We have used 2 same equations because we are going to weight the model differently that means we will prioritize different losses differently.

$$-\frac{1}{N} \sum_{i=1}^N y^i \log \hat{y}_v^i + (1 - y^i) \log(1 - \hat{y}_v^i) \quad (2)$$

$$-\frac{1}{N} \sum_{i=1}^N y^i \log \hat{y}_v^i + (1 - y^i) \log(1 - \hat{y}_v^i) \quad (3)$$

Where,

y^i is 0 or 1 depending on whether the video is real or fake

N is the number of segments

$$d_t^i = \|f_v - f_a\|_2 \quad (4)$$

Where,

f_v is the feature representation of the video

And f_a is the feature representation of audio

The total loss is calculated using Equation (5)

$$\text{LossT} = \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3 \quad (5)$$

Where L_1 and L_2 are Cross-Entropy loss

L_3 is the Contrastive loss

And $\lambda_1, \lambda_2, \lambda_3$ are the weights.

The MDS score is calculated using Equation (6)

$$MDS = \frac{1}{n} \sum_{t=1}^n d_t^i \quad (6)$$

Visual Stream	Audio Stream
	conv_1, 3×3, 1, 64
conv1	batch_norm_1, 64
	pool_1, 1×1, MaxPool
	conv_2, 3×3, 64, 192
conv2_x	batch_norm_2, 192
	pool_2, 3×3, MaxPool
	conv_3, 3×3, 192, 384
conv3_x	batch_norm_3, 384
	conv_4, 3×3, 384, 256
conv4_x	batch_norm_4, 256
	conv_5, 3×3, 256, 256
conv5_x	batch_norm_5, 256
	pool_5, 3×3, MaxPool
	conv_6, 5×4, 256, 512
average pool	batch_norm_6, 512
fc7, 256×7×7, 4096	fc7, 512×21, 4096
batch_norm_7, 4096	batch_norm_7, 4096
fc8, 4096, 1024	fc8, 4096, 1024
batch_norm_8, 1024	batch_norm_8, 1024
dropout, $p = 0.5$	dropout, $p = 0.5$
fc10, 1024, 2	fc10, 1024, 2

Figure 16 – Architecture of audio and Video stream

The ResNet model learns from the normal model of Neural Network. They compare two different models of NN which are of different layers. The model with less layers outperforms the model with more layers and to know why it does that they try to see the in-depth working of the model. They find out about the shortcuts in the NN which it uses when it's hard for the weights to be computed for some layers and hence it skips these layers. To solve it and make the model train better they make the model add more parameters while skipping which makes the model train better and hence increases its performance. Equation (7) represents the equation used to add more parameters and achieve better results.

$$y = F(x, \{w_i\}) + w_s x \quad (7)$$

These results can be seen in Figure 16 Figure 17 where 2 models of different layers are being compared and the model 2 are shown.

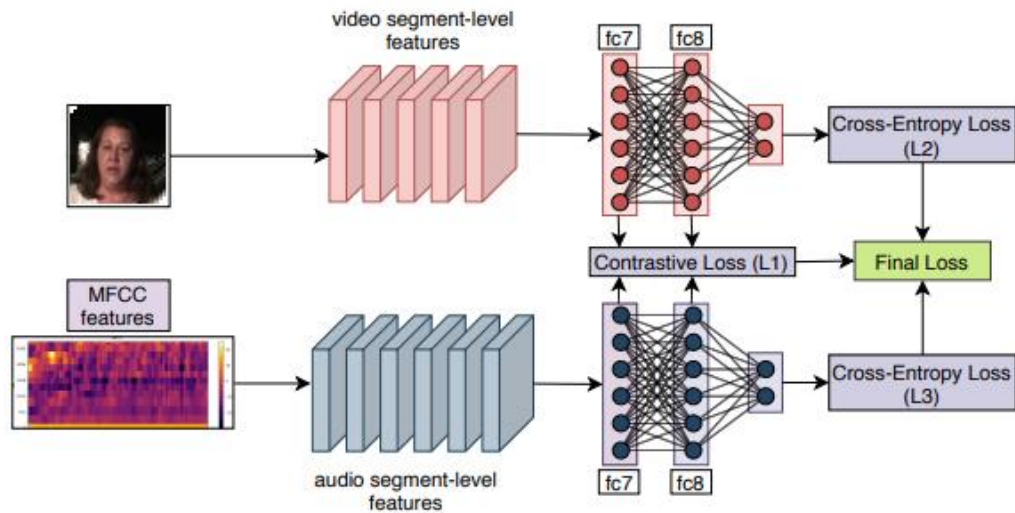


Figure 17 – Architecture of the model

2.1.3 Proposed Model

We suggest using multi-model CNN for deep-fake detection, which means that we will employ more than one strategy and then aggregate the results of those approaches to arrive at a single conclusion. In this solution, we used two distinct approaches to deep fake detection, including recognizing unusual facial expressions and audio-video emotion synchronization. Both of these approaches are described below.

After giving the model the training that was just shown and saving it, we utilized the models to make a prediction about the result. After obtaining the results of each person, aggregate them according to the weighted basis.

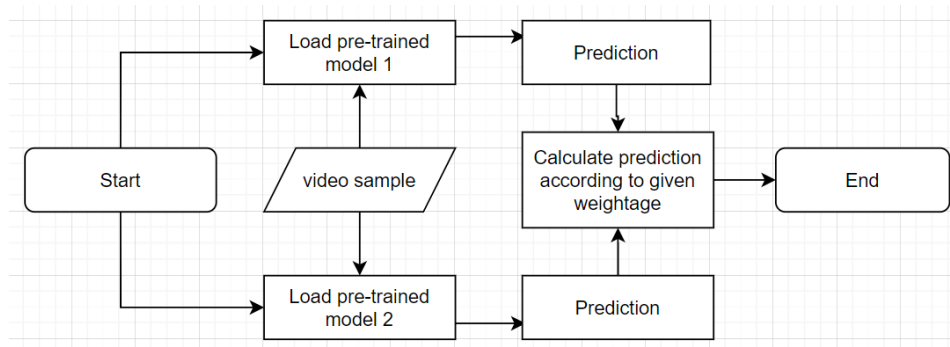


Figure 18 – Final Model Combination of both the model

2.2 Experiment

To determine the optimal dataset, network architecture, pre-processing settings, and hyperparameters for deepfake detection, it is necessary to carry out a series of tests in rapid succession. The scope of our investigation is not complete. Because the area of study is advancing at such a quick pace, it is now impossible to conduct out in-depth investigation. To demonstrate the proper direction for the parameters that are appropriate for deepfake detection, however, a sufficient number of trials need to be performed.

Experiments are carried out one after the other in sequence. In each subsequent experiment, the conditions are those that produced the best possible outcome in the earlier experiment. The answer to the question of which location is deemed to be the best hinges on three factors:

- Achieved the highest possible degree of precision.
- Reduced the amount of loss to the lowest possible level.
- Training stability (e.g., overfitting, overfitting)

TensorFlow 2.2.0, Python 3.8.0, and Keras 2.3.1 were the programming languages that we used. Each and every training is carried out using a split of training data composed of 70 percent and validation data composed of 30 percent. Throughout the whole of the training process, binary cross entropy was used as the error measurement, while RMS Prop served as the optimizer. Indicators that take into consideration factors such as accuracy, precession, recoil, and ROC are. Every single execution is carried out on an AWS instance of size 3.xlarge until you are forced to transfer to a c5.4xlarge because of time limitations. The description of the experiment makes it quite obvious that this is the situation here.

2.2.1 Dataset Composition

Even if a number of deepfake datasets have been made publicly available, it remains the researcher's obligation to choose how to make use of them. The DFDC datasets are configured in a variety of ways for our tests to operate. These recordings are available. It includes a variety of various numbers of frames taken from a variety

of different videos. In order for us to ascertain the weather Which is more significant, the quantity of the dataset or its diversity?

The values for the hyperparameters are a dropout rate of 0.1, a batch size of 100, and an image input size of 128 128. There is no preliminary processing involved. The training of the network is finished when fifty epochs have passed.

We are also provided with the training timings, which enables us to establish the link between the amount of time spent training and the size of the dataset.

2.2.2 Preprocessing

The variety of the data may be increased by the use of preprocessing. In this experiment, there are two separate runs: one of them involves extensive preprocessing, while the other does not. Light preprocessing utilizes rotation range 25, shear range 0.2, zoom range 0.2, width shift range 0.2, and height shift range 0.2. Heavy preprocessing makes use of rotation range 40, shear range 0.2, zoom range 0.2, width shift range 0.2, and height shift range 0.2. Because of this, we are able to examine the impact that various preprocessing techniques have on our performance measurements.

In order to carry out this experiment, the recommended model was used. The dataset known as DFDC was the one that was used. The dropout rate should be set to 0.1, the batch size should be 100, and the image input size should be 128128. After a total of 150 epochs, the training of the network is finished. In the event that the network had not yet finished its training, it was put through an additional 250 epochs of instruction.

Additionally, the training periods are recorded so that a correlation may be made between the amount of time spent training and the amount of time spent preprocessing.

2.2.3 Dropout

Through the use of dropout, overfitting may be prevented. The purpose of this experiment is to investigate the effect that different dropout rates have on classification accuracy.

Additionally, the training durations are recorded, which enables us to establish the extent to which there is a correlation between training time and dropout rate.

In order to carry out this experiment, the recommended model was used. We made use of the dataset provided by DFDC. The hyperparameters are set to a batch size of 100 and a photo input size of 128x128. We do a significant amount of reprocessing. Following a total of 250 epochs, the training was finally finished. In the event that the network had not yet finished its training, it was put through an additional 350 epochs of instruction.

2.2.4 Batch Size

Because we use mini-batch learning, each time we do a run, we are required to specify a batch size. Minibatch learning is an approach that attempts to combine the strengths of batch learning with stochastic learning. The objective of this experiment is to have a better understanding of the correlation that exists between the batch size that we use and the performance needs that we have. In order to achieve this goal, we operate both a 50-batch and a 250-batch production.

In this experiment, we use the proposed model that we developed. DFDC was used as the data set in this study. The size of the photo that is being entered is 128 by 128, and the dropout rate is 0.1. The training of the network was finished after 250 epochs. By examining the amount of time that has passed since training began, we may be able to establish the link that exists between the training period and the batch size.

2.2.5 DCT-Residuals

In this particular experiment, we trained the network using the picture residuals rather than the actual photos themselves. The idea behind picture residuals is that they provide the network with assistance in concentrating on statistical image information rather than the semantical content of images. Amerini et al. [47] and Zampoglou et al. [48] have both made use of DCT-residuals in their research. When attempting to identify Deepfakes, we rely not on the DCT-histograms but rather on the DCT-residuals directly.

After importing the images into Python and running the `scipy.fftpack` built-in `dct` function on them, we were able to compute the DCT-residues of the pictures. The pictures were then exported in PNG format using the `cv2` program (version 3.4.9.33). For the DCT-transform implementation in `scipy.fftpack`, we make use of the DCT-II algorithm. The following formula is used to supply the input pictures x and y :

$$y_k = 2 \sum_{n=0}^{N-1} x_n \cos\left\{\frac{\pi k(2n+1)}{2N}\right\}$$

where k is the pixel in image y that is being computed, N the number of pixels in x 15. In our case, this means $N = 40,000$, $0 \leq y \leq 39,999$.

2.2.6 Generalization and Robustness

After running tests to see how different design selections influenced the performance of our network, we were able to achieve an accuracy rate of more than 80 percent across all eight of the networks that we investigated. The training of our network, on the other hand, is not designed to increase its performance on any given dataset in particular. Instead, we want the network to be able to make use of this dataset as a mechanism for identifying falsified samples in any Deepfake dataset they may have access to. In order to evaluate how successfully our networks are able to carry out this task, we put their capacity for generalization as well as resilience to the test. We determine how well they did based on the results of blind tests so that we may reach our objective. This validation serves a twofold function, which are as follows:

- An evaluation of the network's effectiveness using data sets that are distinct from the one on which it was trained;
- An evaluation of the network's effectiveness using data sets that are distinct from the one on which it was trained;

3 RESULTS

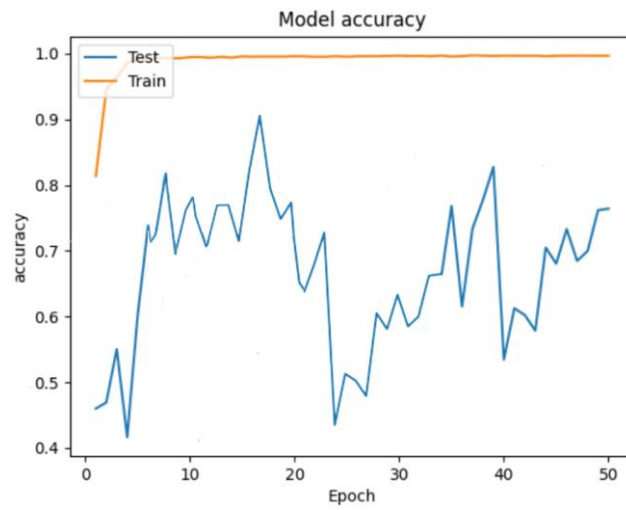
These experiments are conducted with the proposed model on the DFDC dataset. The findings of tests undertaken to investigate the implications of design decisions while constructing a model capable of identifying Deepfake movies are presented in this chapter. The studies look at how dataset size and diversity, as well as preprocessing, dropout rates, batch sizes, network design, and input types, affect the results.

3.1 Dataset Composition

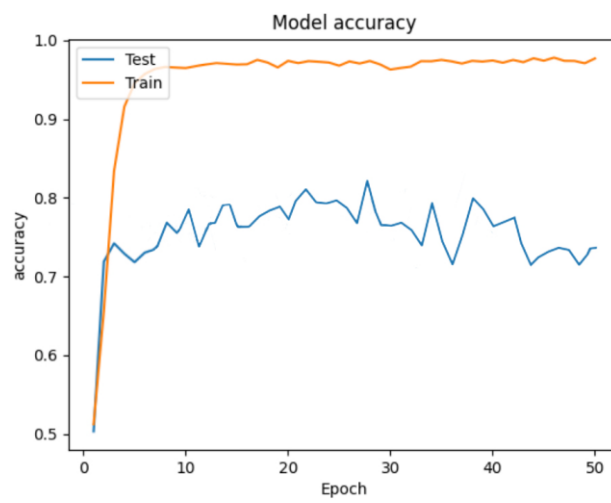
The first experiment looks at the consequences of various dataset compositions. To do this, we apply the proposed model on three datasets: DFDC, Larger DFDC, and Largest DFDC. Each of the three datasets is based on the same number of genuine and deepfake videos in DFDC. Dataset was built such that DFDC simply includes a single frame from 1897 videos, whereas larger DFDC includes all frames from 61 videos, and largest DFDC contains up to 25 frames from 1897 videos. Larger DFDC and Largest DFDC are 15.0 and 24.4 times larger than DFDC, respectively.

The model instantly overfits on Larger DFDC and Largest DFDC (see Fig. 19a and 19b respectively). However, the greatest losses on the two datasets differ: on Larger DFDC, the highest loss is 83.4 (see Fig. 19d); on Largest DFDC, the maximum loss is 12.5 (see Fig. 19e). As a result, the maximum error on the Larger DFDC dataset is 6.7 times larger than on the Largest DFDC dataset.

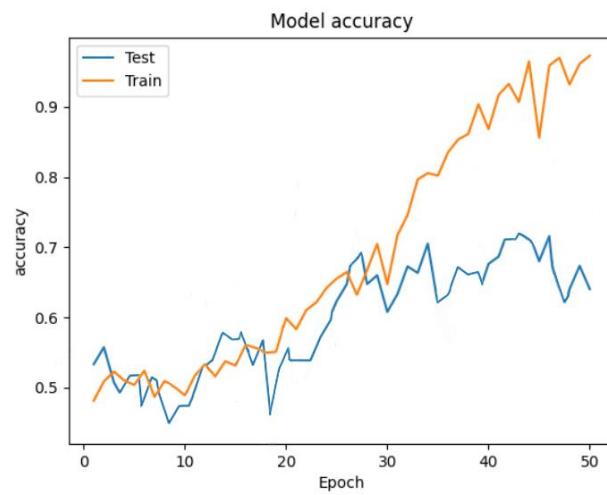
As shown in fig. 19c, the model attains accuracy of 74.5 percent before overfitting sets in, and its loss curve reaches a minimum of 0.54 before overfitting sets in shown in fig. 19f. This indicates that the DFDC dataset is the only one on which the model learns any differentiating patterns. DEDC is the smallest dataset, yet it is also the most diverse. This means that a network trained on a smaller, more diverse dataset can outperform one trained on a larger, less varied dataset.



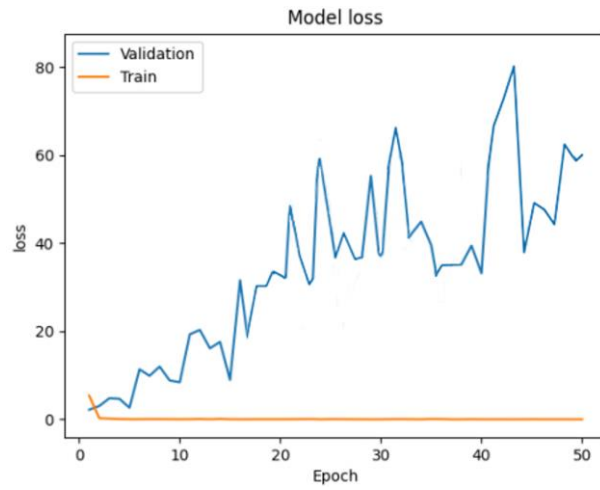
a) accuracy on larger DFDC



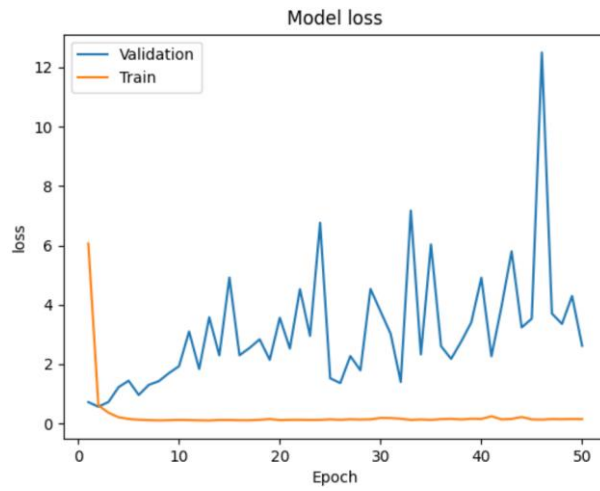
b) accuracy on largest DFDC



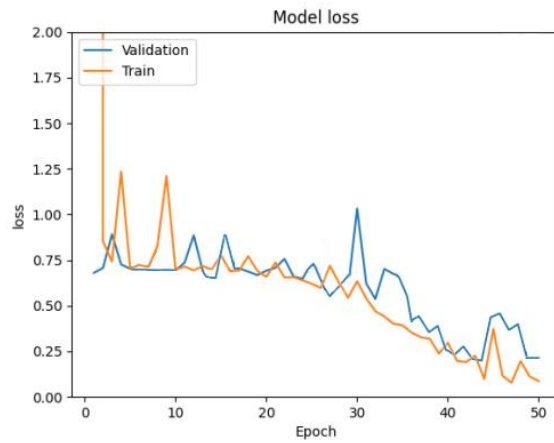
c) accuracy on DFDC



d) Loss on Larger DFDC



e) Loss on largest DFDC



f) Loss on DFDC

Figure 19 – Accuracies and losses of our model using differently composed datasets

Table 2 shows the runtimes for the different runs. The increase in runtime appears to be roughly linear with the increase in dataset size: the model takes about

13.7 times and 27.4 times longer to train on Larger DFDC and Largest DFDC respectively than on DFDC.

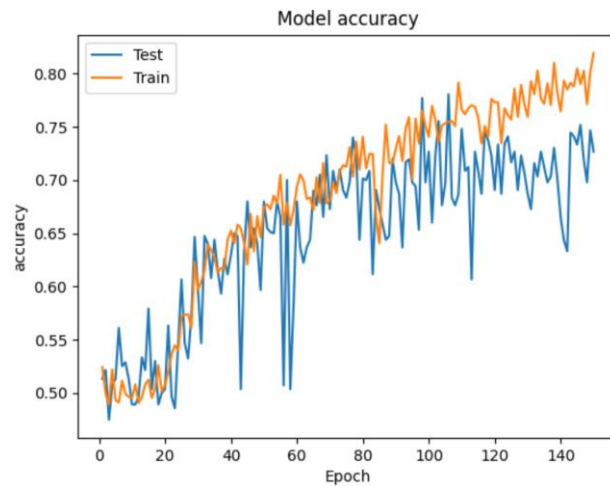
Table 2 – The training times of the model on differently composed datasets

Dataset	Epoch	Training time(s)	Average
DFDC	50	1185.1	23.7
Large DFDC	50	16223.9	324.5
Largest DFDC	50	32435.1	648.7

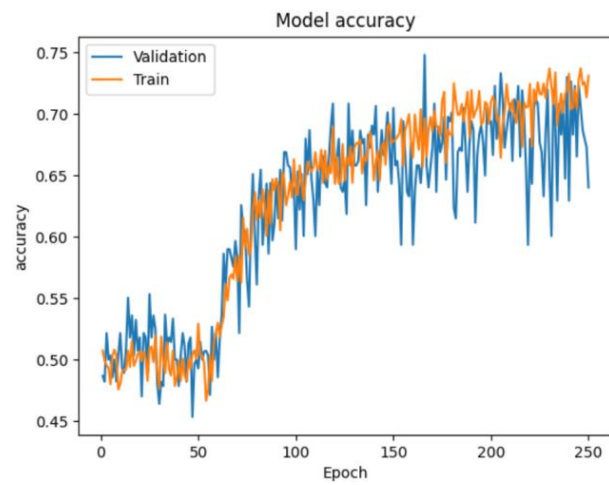
3.2 Preprocessing

Heavy preprocessing is used in one run while light preparation is used in the other. The accuracy curves that arise are presented in Fig. 20. Table 3 shows the model's performance with various preprocessing settings.

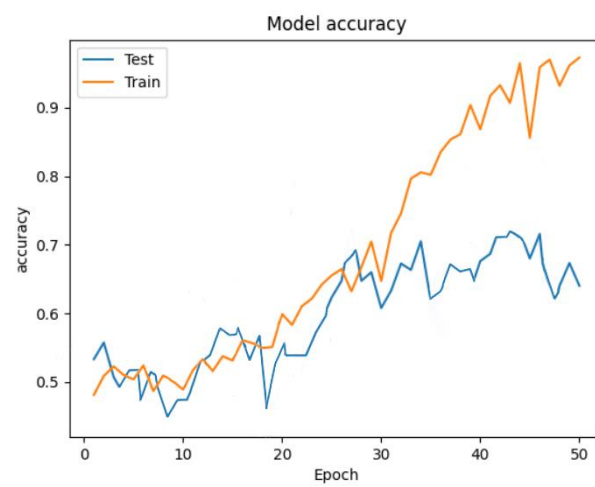
We can see from Fig. 20 that the more preprocessing there is, the longer it takes for the accuracy curve to start ascending. This is likely due to more data variety, which causes the network to view a significantly different subset of the data each epoch, making it take longer to uncover the defining global patterns.



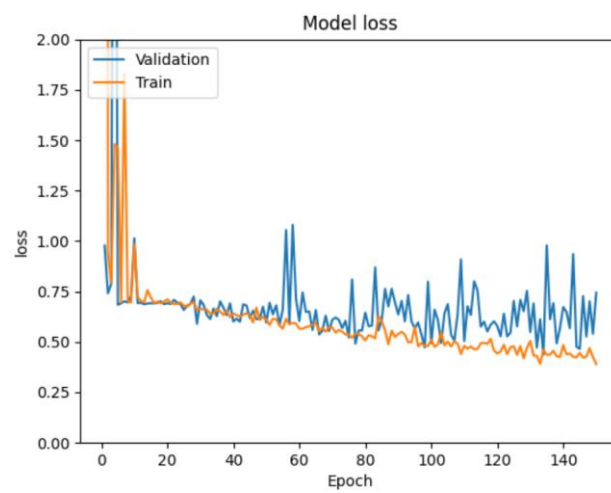
(a) Accuracy with light preprocessing



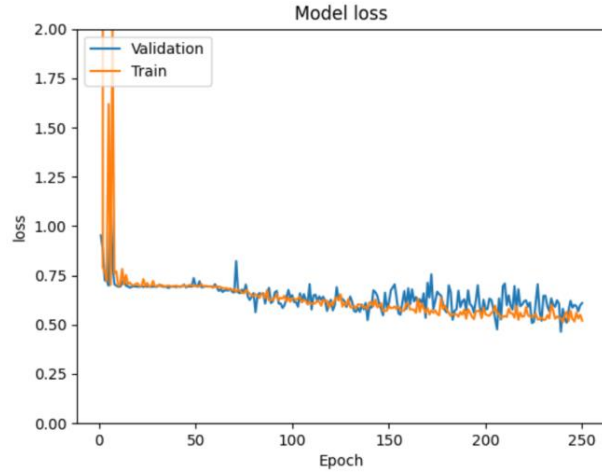
(b) Accuracy with heavy preprocessing



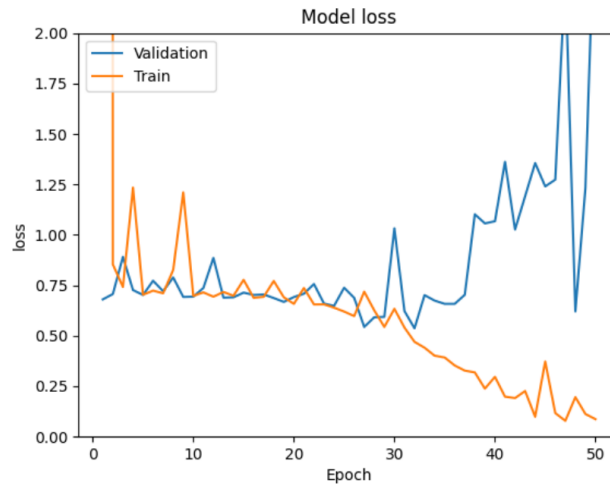
(c) Accuracy with no preprocessing



(d) Loss with light preprocessing



(e) Loss with heavy preprocessing



(f) Loss with no preprocessing

Figure 20 – Losses and accuracies with different preprocessing settings

Table 3 shows that extensive preprocessing achieves a maximum accuracy that is 2.1 percent greater than no preprocessing but 3.3 percent lower than light preprocessing. Heavy preprocessing, on the other hand, has a smaller minimum loss (by 0.1) than both light preprocessing and no preprocessing (by 0.8). Furthermore, high preprocessing prevents overfitting entirely; without it, overfitting occurs after epoch 29, while light preparation only prevents overfitting for another 81 epochs.

Table 3 – Performance on DFDC using different preprocessing settings

Preprocessing	Performance		
	Accuracy(Max(%))	Loss(Min)	Overfitting (epoch)
Heavy preprocessing	74.8	0.46	-
Light preprocessing	78.1	0.47	110
No preprocessing	72.7	0.54	29

The validation accuracy is the highest level reported, while the validation loss is the lowest level reported. The greatest accuracy and minimal loss provided are those before the model overfits. The epoch after which overfitting begins is reported in the last column; a hyphen denotes that no overfitting occurs.

We will choose extensive preprocessing over light or no preprocessing since it beats them on two out of three criteria; especially since 3.3 percent is not a significant difference considering the stochastic nature of the process.

Table 4 shows the amount of time it took to train with various preprocessing settings. Light preprocessing takes an average of 25.9 seconds per epoch; heavy preprocessing takes an average of 23.9 seconds per epoch; and no preprocessing takes an average of 23.7 seconds per epoch. This takes 2.2 seconds longer than light preprocessing and 0.2 seconds longer than strong preprocessing.

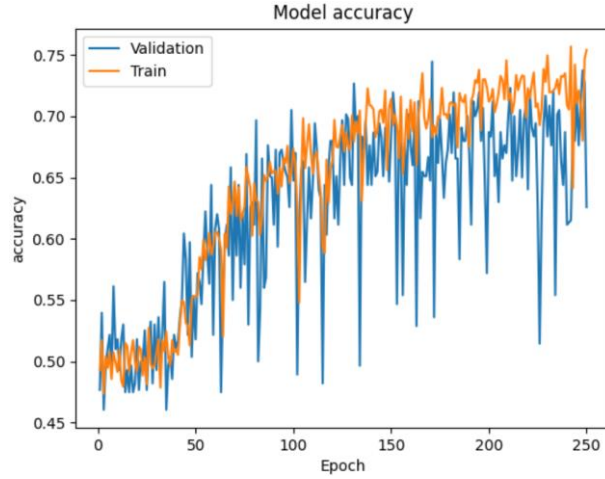
We will consider the difference in time elapsed a measurement mistake because it would be strange for light preparation to be more computationally expensive than heavy and no preprocessing. As a result, given the difference between no preprocessing and intensive preprocessing is just 0.2 seconds, we hypothesize that different preprocessing settings have little influence on the amount of time the model takes to train.

Table 4 – The training times with different preprocessing settings

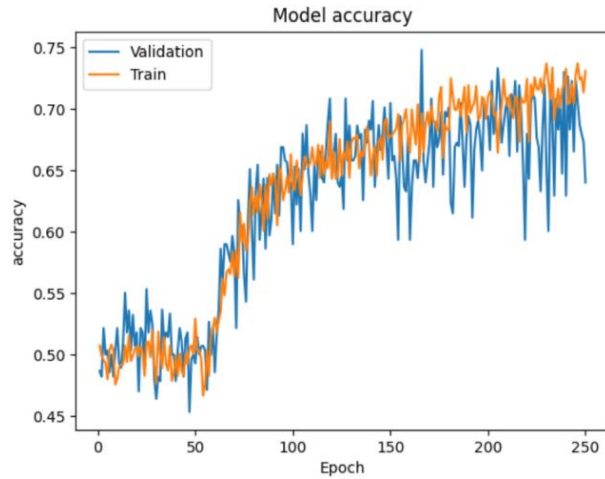
Preprocessing	Epochs	Training time(s)	Average
Heavy preprocessing	250	5981.7	23.9
Light preprocessing	150	3879.3	25.9
No preprocessing	50	1185.1	23.7

3.3 Dropout

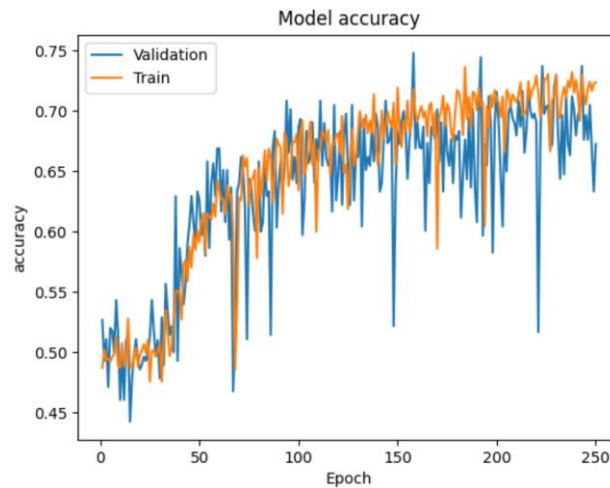
Figure 21 depicts the resultant loss and accuracy curves. The results are shown in Table 5. These runs may be compared to the above section with heavy preprocessing, which has a dropout rate of 0.1 but otherwise identical parameters.



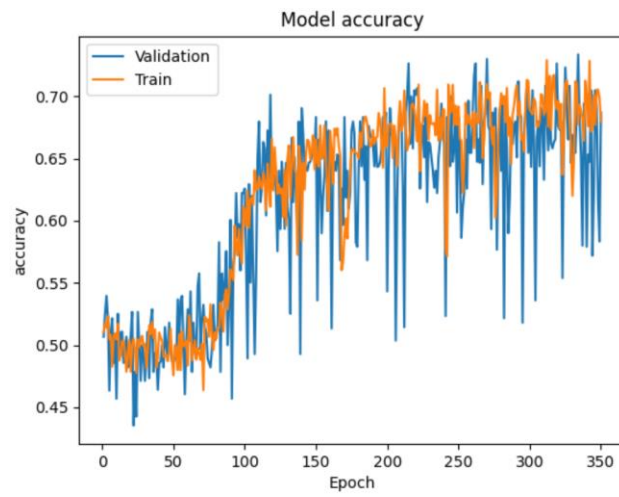
(a) Accuracy without dropout



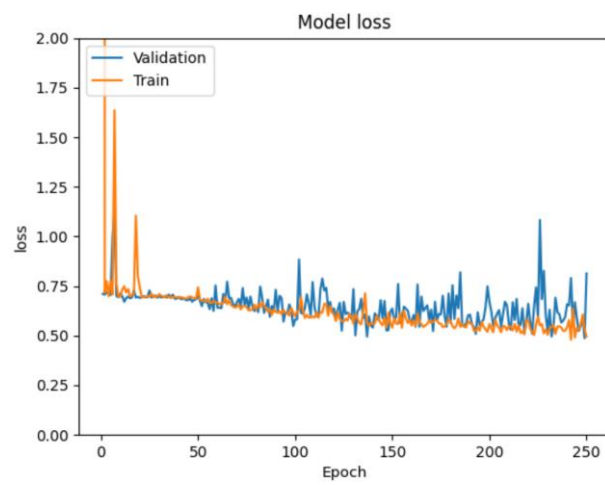
(b) Accuracy with dropout rate 0.1



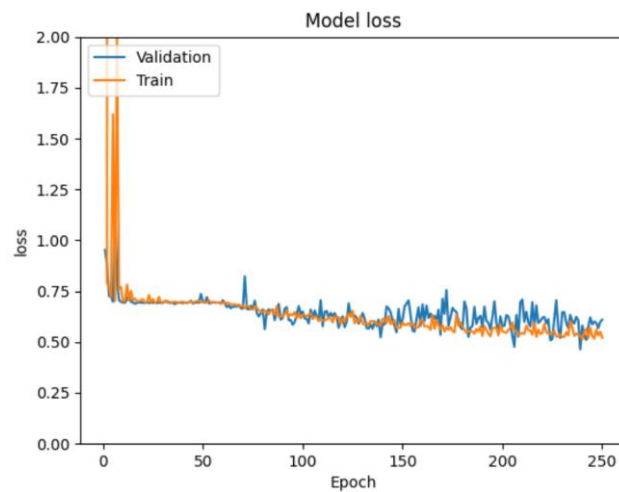
(c) Accuracy with dropout rate 0.2



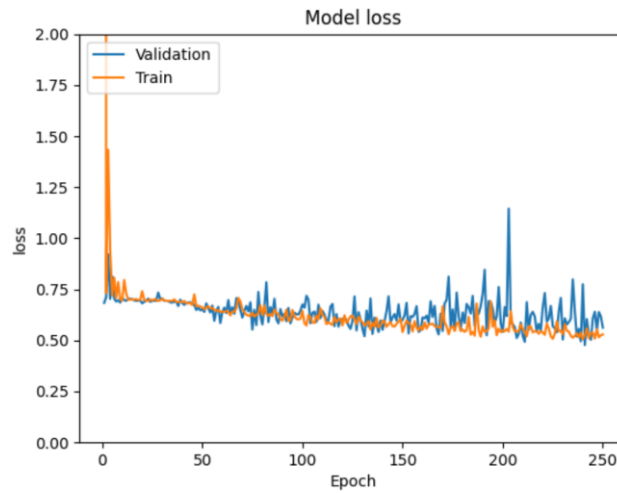
(d) Accuracy with dropout rate 0.5



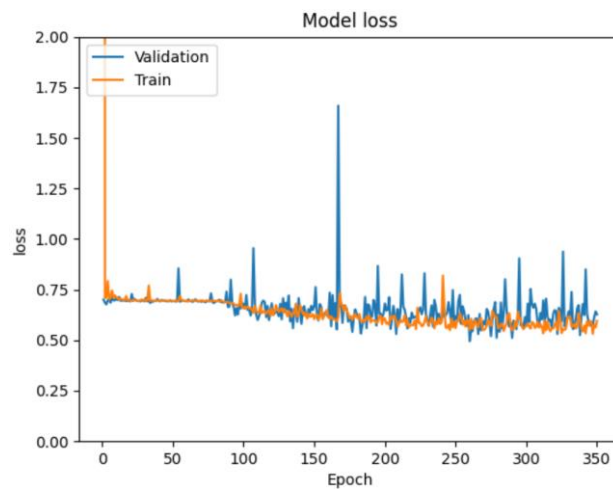
(e) Loss without dropout



(f) Loss with dropout rate 0.1



(g) Loss without dropout rate 0.2



(h) Loss with dropout rate 0.5

Figure 21 – Losses and accuracies using different dropout rates

According to Fig. 21, the accuracy curve begins to climb around epoch 40 without dropout; with a dropout rate of 0.1, the accuracy curve begins to climb around epoch 60; with a dropout rate of 0.2, the accuracy curve begins to climb around epoch 30; and with a dropout rate of 0.5, the accuracy curve begins to climb around epoch 80. This means that the model takes longer to recognize patterns for big dropout rates than for lower dropout rates.

Figure 21 also illustrates that using dropout keeps the training and validation accuracies closer together: when no dropout is used, the distance between them seems to be slightly bigger than when dropout rates of 0.1 or 0.2 are employed. The two curves do not appear to split at all when the dropout rate is set to 0.5.

According to Table 5, dropout has a little impact on network performance in terms of accuracy and loss: the greatest accuracies attained with all four dropout rates are between 73.4 and 74.8 percent, while the smallest losses are between 0.46 and 0.50 percent. These differences are insignificant. In addition, none of the models are overfit.

Because of the modest differences in accuracy and loss, dropout cannot be employed to achieve improved accuracy in this scenario. However, for a network that is more prone to overfitting in the first place, this may be different.

Table 5 – Performance using different dropout rates

Dropout Rate	Performance	
	Accuracy Max (%)	Loss Min
0	74.5	0.49
0.1	74.8	0.46
0.2	74.8	0.48
0.5	73.4	0.50

Table 6 – The training times using different dropout rates

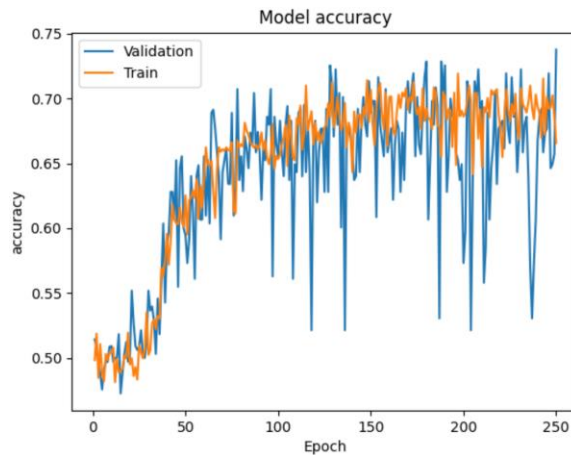
Dropout Rates	Epochs	Training time(s)	Average
0	250	6388.1	25.6
0.1	250	5981.7	23.9
0.2	250	6426.3	25.7
0.5	350	9006.7	25.7

Table 6 shows the amount of time it took for different dropout rates to complete training. Training without dropout took 25.6 seconds per epoch on average, 23.9 seconds per epoch with a dropout rate of 0.1 ,25.7 seconds per epoch with a dropout rate of 0.2, and 25.7 seconds per epoch with a dropout rate of 0.5. Training with a dropout rate of 0.1 takes 1.7 seconds per epoch less than training without one, whereas training with a dropout rate of 0.2 or 0.5 takes 0.1 seconds longer each epoch. We believe these differences to be insignificant.

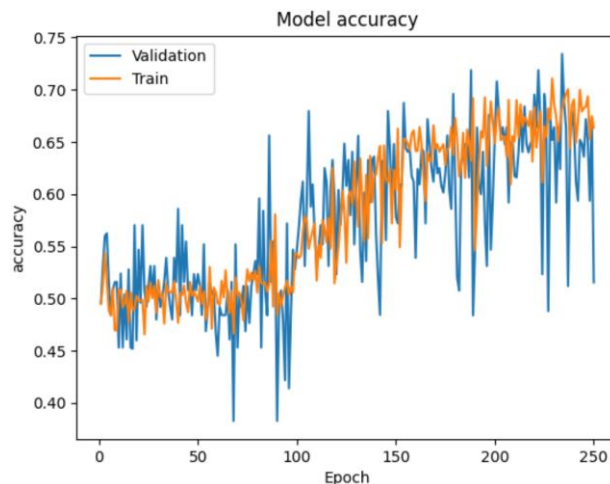
To close the gap between the training and validation curves, we'll utilize a dropout rate of 0.1 for the rest of the study. We chose dropout rate 0.1 because the difference between 0.1 and 0.2 appears to be small: it is the dropout rate used in past experiments and allows us to compare prospective trials to previous ones.

3.4 Batch Size

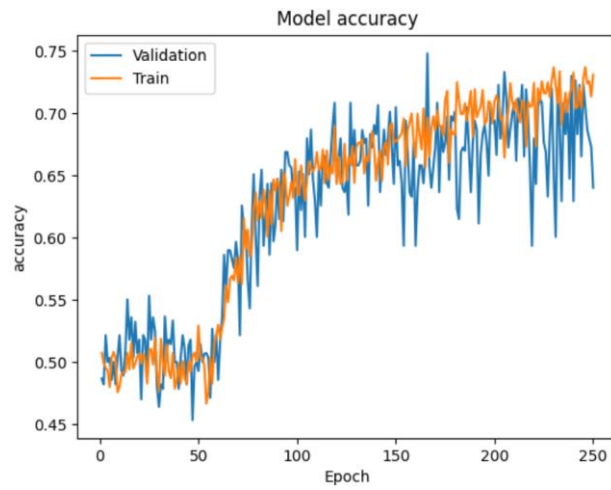
There are a variety of batch sizes utilized. Figure 22 depicts the loss and accuracy curves that were produced as a consequence. Table 7 contains the results of the performance analysis. These findings may be compared to those obtained during the run with heavy preprocessing in section 2, which has a batch size of 250 and, other than that, employs the same parameters throughout.



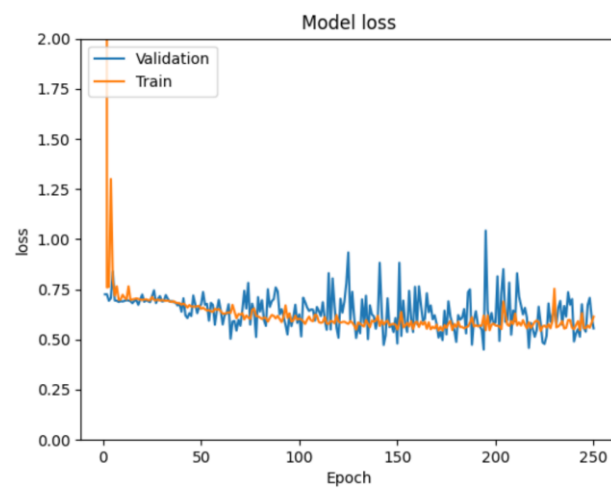
(a) Accuracy with batch size 50



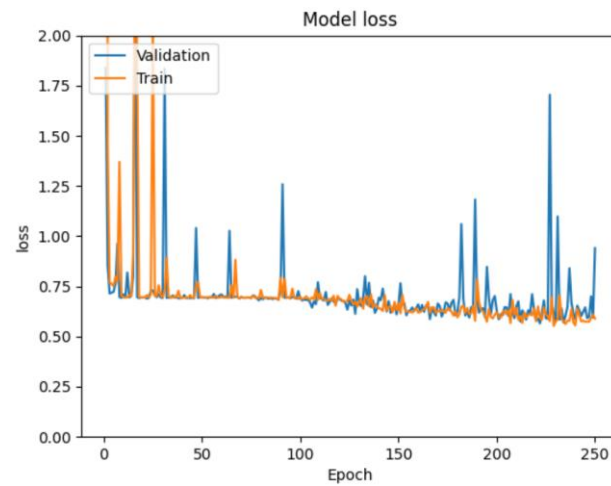
(b) Accuracy with batch size 100



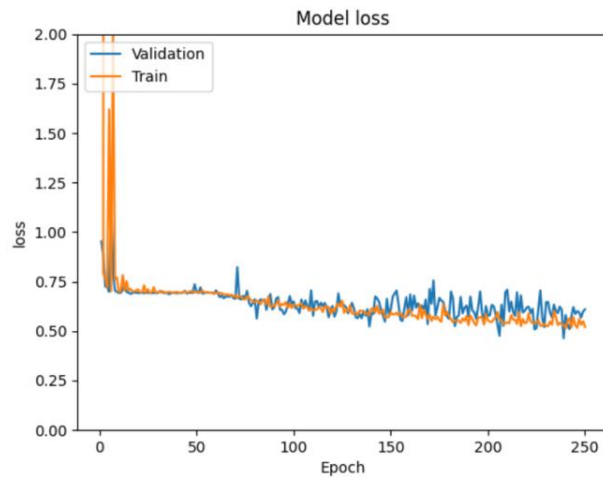
(c) Accuracy with batch size 250



(d) Loss with batch size 50



(e) Loss with batch size 100



(f) Loss with batch size 250

Figure 22 – Losses and accuracies using different batch sizes

Figure 22a demonstrates that the accuracy curve for a batch size of 50 has the sharpest slope between epochs 20 and 59; on the other hand, Figure 22b demonstrates that the accuracy curve for a batch size of 100 rises gently and does not have any notably steep slopes. Figure 22c demonstrates that the point on the accuracy curve that corresponds to batch size 250 has the greatest slope between epochs 60 and 70.

It has come to our attention that the amount of time necessary for the accuracy curves to begin ascending increases in proportion to the batch size. Because of this, it takes the algorithm a longer amount of time to learn how to differentiate between the two classes when working with bigger batch sizes. This is most likely due to the fact that the gradient descent method only changes the parameters once it has examined a whole set of photos. It is possible that seeing 250 photographs rather than 50 will make it more challenging to establish early patterns to work with.

Table 7 – Performance using different batch sizes

Batch Size	Performance	
	Accuracy Max. (%)	Loss Min.
50	73.8	0.45
100	73.4	0.57
250	74.8	0.46

We can see from Table 7 that there is not a large amount of variation in the accuracies that were reached. The difference in loss is much more pronounced: the batch size 250 produces a greater loss than the batch sizes 50 and 100. This may be because it is more challenging to optimize when seeing a large number of samples before making any adjustments to the parameters.

Table 8 contains a summary on the amount of time that has passed since training with various batch sizes. During training, the amount of time that passed was, on average, 26.2 seconds when the batch size was 50, and it was 22.8 seconds when the batch size was 100. It took 23.9 seconds for each epoch when the batch size was 250. It would seem that the amount of time needed for training is reduced in proportion to the batch size. This is perhaps because the practice of backpropagation is used less commonly these days. The fact that the changes are not very large may suggest that even if backpropagation does affect the amount of time that has passed since training began, it is not the primary component.

Table 8 – The training times using different batch sizes

Batch Size	Epoch	Training Time(s)	Average
50	250	6547.8	26.2
100	250	5691.0	22.8
250	250	5981.7	23.9

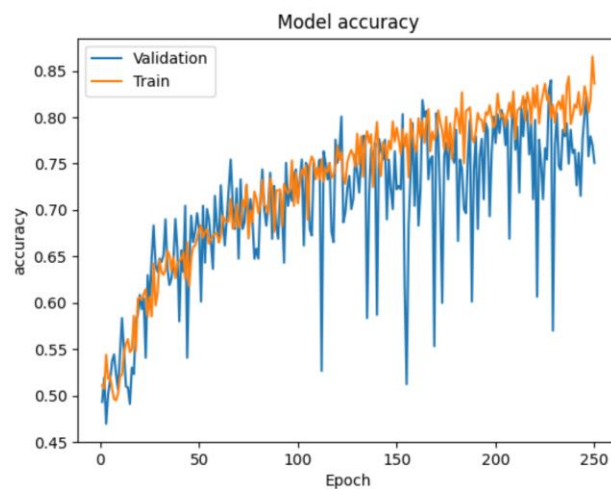
We are going to stick with a batch size of 250 for the rest of our research. This makes it possible for us to compare the findings of prospective studies with those of prior studies without affecting the accuracy of either set of data. In spite of the fact that one may argue that batch size 100 is better owing to the decreased training time per epoch, our argument is that this reduction is offset by the extra epochs that are required for the algorithm to begin learning.

3.5 DCT-Residuals

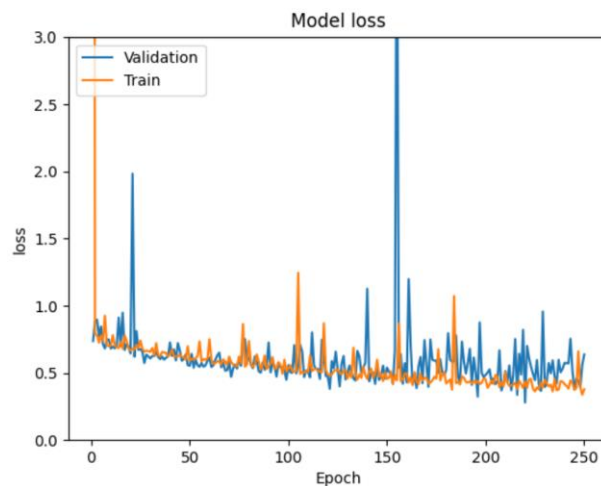
In the preprocessing settings, the rotation range is set to 40, the shear range is set to 0.2, the zoom range is set to 0.2, the width shift range is set to 0.2, the height

shift range is set to 0.2, and the horizontal flipping setting is used. The input size is 200 x 200 and the batch size is 100.

The network that was trained on DCT-residual pictures performs better than its counterparts that were trained on the original images, despite the fact that the accuracy differences are negligible: 1.4%, The difference in loss is equally insignificant in the latter case: 0.06, to the benefit of the model that was trained on DCT-residuals. At around epoch 30, overfitting occurs on both the DCT-residuals and the original. It is possible to draw the conclusion that models trained on DCT-residual are superior to their counterparts trained on the original pictures.



(a) Accuracy with DCT-residuals.



(b) Loss with DCT-residuals.

Figure 23 – Losses and accuracies using DCT-residuals

Table 9 – Performance using DCT-residuals

Accuracy Max. (%)	Loss Min.
84.0	0.28

3.6 Generalization and Robustness

Table 10 – reports the accuracies, sensitivities, specificities, precisions, fall-outs, miss rates, and F1-measures.

ACC (%)	SEN (%)	SP (%)	PR (%)	FO (%)	MR (%)	F1(%)
92.4	95.6	89.2	89.8	10.8	4.4	92.6

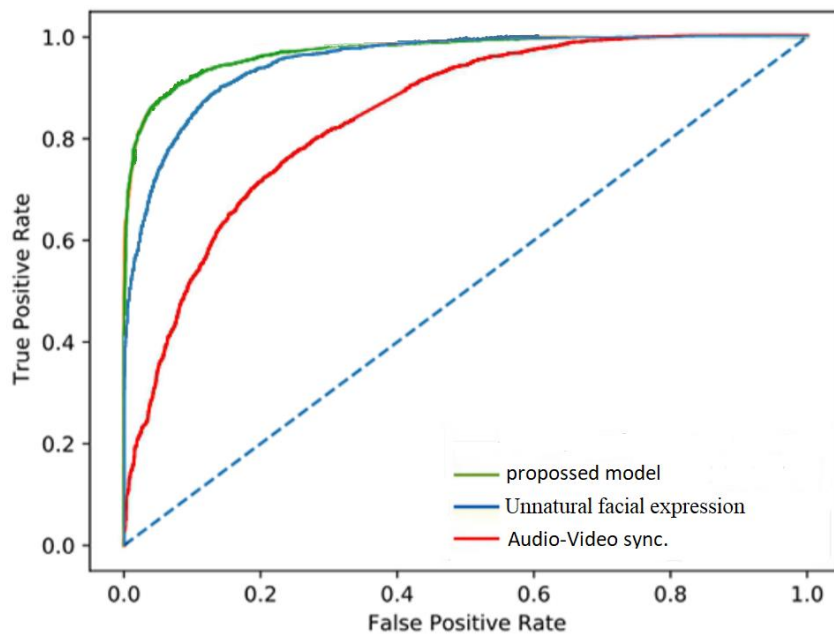


Figure 24 – ROC Curves comparison between our best model and others

CONCLUSION

We first proposed a model based on Convolution Neural Network that extracts frames from the video and detected faces that are constant throughout the video. After this these extracted frames are sent to two different models. The first model is sent the whole video in which it splits the videos into small segments and splits it based on audio and video. Heat-Map is generated based on the audio input and then these two inputs are sent into ResNet inspired models features are learnt and Contrastive loss is calculated along with the Loss entropy. All the loss functions are aggregated together for a combined loss, and this is used to train the model. The second model takes frames as input and crops these frames according to the most prominent face in the video. The input is then sent to ResNext based Convolutional Neural Network where the resultant output is a 2048-dimensional feature vector matrix. This is sent to the LSTM with 2048 hidden layers. The LSTM has 2 outputs real/fake and input is 2048 feature vector. The LSTM uses a SoftMax layer to determine the confidence to the results from the LSTM.

As the research progressed, we carried out a number of experiments with the goal of determining the impact that various hyperparameters, dataset compositions, and network architectures have on the overall performance of convolutional neural networks when it comes to Deepfake identification. This study also examined the performance of models with data that had not been observed before in order to evaluate their resilience.

We discovered that the diversity of the dataset has a greater impact on performance than the quantity of the dataset. Overfitting occurred very immediately after using a dataset that had 25 frames for each DFDC movie. In comparison, the network was able to achieve an accuracy of 72.7 percent by using a dataset that only had one frame per DFDC video. This dataset consisted of just 1897 frames that were split between two categories. It would seem that increases in runtime are essentially linear with increases in the amount of the dataset. The rest of our study was conducted using the standard DFDC.

Overfitting may be reduced thanks to preprocessing. Nevertheless, in this particular instance, it does not assist in making the model more accurate. It does not seem that preprocessing has any effect on the total amount of time spent on training.

In a similar vein, greater dropout rates are able to maintain a closer proximity between the training and validation learning curves, but this does not assist the model reach a better accuracy. It does not seem that higher dropout rates have any effect on the total amount of time required for training.

It also seems that the highest precision that a model is capable of achieving is unaffected by the size of the batches. On the other hand, it does seem to have an effect on the number of epochs required for the model to begin learning: the smaller the batch size, the sooner the algorithm begins learning. Last but not least, it seems that the training runtimes become shorter as the batch size increases.

The best level of accuracy that could be attained was 92.5 percent.

In the last step of this test, we evaluated the generalization and resilience capacities of the networks. We evaluated the network's resilience by having it analyze previously unknown examples from the dataset on which it was trained. The findings demonstrated that there was just a little decline in accuracy. As a result of the fact that the model showed very little exhibited very little deterioration, we have come to the conclusion that they generalize the best. None of the networks were reliable, since testing them on images taken from datasets other than the ones, they were trained on resulted in an accuracy that was about equivalent to random.

- FUTURE WORKS

We only used 2 sub model which focus on 2 detection features mentioned in abstract, further more ahead in the future more than 2 or maybe all 15 detection features can be used to detect the deepfake.

Based on our studies and the few datasets we have access to; it seems that it would be beneficial for us to use networks with a capacity that is on the lower end of the spectrum. Additional study might explore if there is a certain optimum

convolutional body with five to twelve convolutional layers that can perform very well when assigned the job of detecting deep fakes.

In addition, future research might do tests using other CNN networks, such as VGGs, DenseNets, and ResNets, in an effort to determine whether or not the model overfits better on other networks as opposed to the network that was employed in our model. Experimenting with more of these networks, each of which has a different depth but has an architecture similar to the others should yield more understanding.

In conclusion, it is possible that advancements in deep learning architecture in general, and convolutional network design in particular, will have applications in the area of Deepfake detection. For instance, research into developing stronger activation functions seems to have potential.

When viewing videos on Deepfake, the telltale signals of tampering are often temporal in nature. As a result, it is quite probable that the performance of the network will increase if interframe detection is implemented. The next stage in the investigation of Deepfake videos will hopefully include this intriguing future step.

We are still of the opinion that the use of an appropriate residual filter may increase performance. The use of residual filters may assist detection algorithms in concentrating their attention on statistical picture information, which should make it simpler to locate anomalies. It is necessary to do further study on the many types of residual filters, their qualities, and the influence that they have on the effectiveness of detection algorithms.

REFERENCES

1. MarekKowalski/FaceSwap. <https://github.com/MarekKowalski/FaceSwap>, accessed: 2020-04-06.
2. Brian Dolhansky, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. The deepfake detection challenge dataset, 2020. URL <https://arxiv.org/abs/2006.07397>.

3. DeepFaceLab. <https://github.com/iperov/DeepFaceLab>, accessed: 2020-03-30.
4. faceswap. <https://github.com/deepfakes/faceswap>, accessed: 2020-03-30.
5. Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A new dataset for deepfake forensics. arXiv preprint arXiv:1909.12962, 2019.
6. Pavel Korshunov and Sébastien Marcel. Deepfakes: a new threat to face recognition? assessment and detection. CoRR, abs/1812.08685, 2018. URL <http://arxiv.org/abs/1812.08685>.
7. Yuval Nirkin, Yosi Keller, and Tal Hassner. FSGAN: Subject agnostic face swapping and reenactment. In ICCV, 2019.
8. Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. FaceForensics++: Learning to detect manipulated facial images. In International Conference on Computer Vision (ICCV), 2019
9. Chorowski J. et al. Unsupervised speech representation learning using wavenet autoencoders //IEEE/ACM transactions on audio, speech, and language processing. – 2019. – T. 27. – №. 12. – C. 2041-2053.
- 10.M. Kowalski, J. Naruniec, T. Trzcinski, Deep alignment network: A convolutional neural network for robust face alignment, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017, pp. 88–97
- 11.Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., Nießner, M. (2016, November). Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). <https://niessnerlab.org/papers/2016/1/facetoface/thies2016face.pdf>

- 12.Liu M. Y. et al. Few-shot unsupervised image-to-image translation //Proceedings of the IEEE/CVF International Conference on Computer Vision. – 2019. – C. 10551-10560.
- 13.[43] Park T. et al. GauGAN: semantic image synthesis with spatially adaptive normalization //ACM SIGGRAPH 2019 Real-Time Live!. – 2019. – C. 1-1.
- 14.DeepFaceLab: Explained and usage tutorial.[Electronic resource] Access mode: <https://mrdeepfakes.com/forums/thread-deepfacelabexplained-and-usage-tutorial>
- 15.DSSIM [Electronic resource]. Access mode: https://github.com/keras-team/kerascontrib/blob/master/keras_contrib/losses/dssim.py.
- 16.Lattas A. et al. AvatarMe: Realistically Renderable 3D Facial Reconstruction" In-the-Wild" //Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. – 2020. – C. 760-769.
- 17.Ha S. et al. Marionette: Few-shot face reenactment preserving identity of unseen targets //Proceedings of the AAAI Conference on Artificial Intelligence. – 2020. – T. 34. – №. 07. – C. 10893-10900.
- 18.Deng Y. et al. Disentangled and controllable face image generation via 3d imitative-contrastive learning //Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. – 2020. – C. 5154-5163.
- 19.Tewari A. et al. Stylerig: Rigging stylegan for 3d control over portrait images //Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. – 2020. – C. 6142-6151.
- 20.Li L. et al. Faceshifter: Towards high fidelity and occlusion aware face swapping //arXiv preprint arXiv:1912.13457. – 2019
- 21.Nirkin Y., Keller Y., Hassner T. Fsgan: Subject agnostic face swapping and reenactment //Proceedings of the IEEE/CVF International Conference on Computer Vision. – 2019. – C. 7184-7193.

- 22.Olszewski K. et al. Transformable bottleneck networks //Proceedings of the IEEE/CVF International Conference on Computer Vision. – 2019. – C. 7648-7657.
- 23.Viazovetskyi Y., Ivashkin V., Kashin E. Stylegan2 distillation for feed-forward image manipulation //European Conference on Computer Vision. – Springer, Cham, 2020. – C. 170-186.
- 24.Thies J. et al. Neural voice puppetry: Audio-driven facial reenactment //European Conference on Computer Vision. – Springer, Cham, 2020. – C. 716-731.
- 25.B. Dolhansky, R. Howes, B. Pflaum, N. Baram, C. Canton-Ferrer, The deepfake detection challenge (dfdc) preview dataset, ArXiv abs/1910.08854.
- 26.Thies, J., Zollhöfer, M., Nießner, M. (2019, July). Deferred Neural Rendering: Image Synthesis using Neural Textures. ACM Transactions on Graphics. <https://arxiv.org/pdf/1904.12356.pdf>
- 27.Thanh Thi Nguyen, Quoc Viet Hung Nguyen, Dung Tien Nguyen, Duc Thanh Nguyen, Thien Huynh-The, Saeid Nahavandi, Thanh Tam Nguyen, Quoc-Viet Pham, Cuong M. Nguyen(sep 2019)Deep Learning for Deepfakes Creation and Detection: A Survey <https://arxiv.org/abs/1909.11573>
- 28.E. R. S. Rezende, G. C. S. Ruppert, T. Carvalho, Detecting computer generated images with deep convolutional neural networks, in: 30th Conference on Graphics, Patterns and Images, SIBGRAPI, 2017, pp. 71–78.
- 29.Sengur, Z. Akhtar, Y. Akbulut, S. Ekici, U. Budak, Deep Feature Extraction for Face Liveness Detection, in: 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), IEEE, 2018, pp. 1–4.

- 30.H. Mo, B. Chen, W. Luo, Fake Faces Identification via Convolutional Neural Network, in: Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, ACM, 2018, pp. 43–47.
- 31.T. Karras, T. Aila, S. Laine, J. Lehtinen, Progressive growing of gans for improved quality, stability, and variation, arXiv preprint arXiv:1710.10196.
- 32.A. Khodabakhsh, R. Ramachandra, K. Raja, P. Wasnik, C. Busch, Fake face detection methods: Can they be generalized?, in: 2018 international conference of the biometrics special interest group (BIOSIG), IEEE, 2018, pp. 1–6.
- 33.N. S. Ivanov, A. V. Arzhskov, V. G. Ivanenko, Combining deep learning and super-resolution algorithms for deep fake detection, in: IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2020, pp. 326–328.
- 34.F. Marra, D. Gragnaniello, D. Cozzolino, L. Verdoliva, Detection of gangenerated fake images over social networks, 1st Conference on Multimedia Information Processing and Retrieval, MIPR (2018) 384–389.
- 35.C.-C. Hsu, C.-Y. Lee, Y.-X. Zhuang, Learning to detect fake face images in the wild, in: 2018 International Symposium on Computer, Consumer and Control (IS3C), IEEE, 2018, pp. 388–391.
- 36.C.-C. Hsu, Y.-X. Zhuang, C.-Y. Lee, Deep fake image detection based on pairwise learning, Applied Sciences 10 (1) (2020) 370.
- 37.P. Korshunov, S. Marcel, Vulnerability assessment and detection of deepfake videos, in: International Conference on Biometrics (ICB), IEEE, 2019, pp. 1–6.
- 38.X. Yang, Y. Li, S. Lyu, Exposing deep fakes using inconsistent head poses, in: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 8261–8265.

- 39.J. Frank, T. Eisenhofer, L. Schonherr, A. Fischer, D. Kolossa, T. Holz, Leveraging frequency analysis for deep fake image recognition (2020).
- 40.U. A. Ciftci, I. Demir, L. Yin, How do the hearts of deep fakes beat? deep fake source detection via interpreting residuals with biological signals (2020).
- 41.A. A. Maksutov, V. O. Morozov, A. A. Lavrenov, A. S. Smirnov, Methods of deepfake detection based on machine learning, in: IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2020, pp. 408–411.
- 42.D. G'uera, E. J. Delp, Deepfake video detection using recurrent neural networks, in: 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), IEEE, 2018, pp. 1–6.
- 43.Y. Li, M. Chang, S. Lyu, In actu oculi: Exposing AI created fake videos by detecting eye blinking, in: IEEE International Workshop on Information Forensics and Security (WIFS), IEEEExplore, 2018, pp. 1–7.
- 44.J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in CVPR, 2015, pp. 2625–2634.
- 45.Yuezun Li, Siwei Lyu, Exposing DeepFake Videos By Detecting Face Warping Artifacts, <https://arxiv.org/abs/1811.00656>
- 46.Darius Afchar, 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), pages 1–7. IEEE, 2018.
- 47.I. Amerini, T. Uricchio, L. Ballan, and R. Caldelli. Localization of jpeg double compression through multi-domain convolutional neural networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1865–1871, 2017.

- 48.M. Zampoglou, F. Markatopoulou, G. Mercier, D. Touska, E. Apostolidis, S. Papadopoulos, R. Cozien, I. Patras, V. Mezaris, and I. Kompatsiaris. Detecting tampered videos with multimedia forensics and deep learning. In I. Kompatsiaris, B. Huet, V. Mezaris, C. Gurrin, W.-H. Cheng, and S. Vrochidis, editors, *MultiMedia Modeling*, pages 374–386, Cham. Springer International Publishing, 2019.
49. Thanh Thi Nguyen, Quoc Viet Hung Nguyen, Dung Tien Nguyen, Duc Thanh Nguyen, Thien Huynh-The, Saeid Nahavandi, Thanh Tam Nguyen, Quoc-Viet Pham, Cuong M. Nguyen. Deep Learning for Deepfakes Creation and Detection: A Survey.....
<https://arxiv.org/abs/1909.11573>
- 50.<https://blog.devgenius.io/resnet50-6b42934db431>
- 51.<https://colah.github.io/posts/2015-08-Understanding-LSTMs/>