

A PROJECT REPORT ON

COMPARATIVE ANALYSIS OF DEEPFAKE DETECTION ALGORITHMS

AND ITS APPLICATIONS

SUBMITTED TO THE

CUMMINS COLLEGE OF ENGINEERING FOR WOMEN, KARVENAGAR, PUNE

(an autonomous institute affiliated to Savitribai Phule Pune university)

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF TECHNOLOGY (COMPUTER ENGINEERING)

SUBMITTED BY

RASHMI DESHMUKH	C22018221322 (4321)
DISHA CHAUDHARI	C22018221327 (4312)
MANASI KASANDE	C22018221348 (4348)
PARUL SHARMA	C22018221368 (4374)
TANISHKA PATEL	C22018221370 (4369)



DEPARTMENT OF COMPUTER ENGINEERING

MKSS'S CUMMINS COLLEGE OF ENGINEERING FOR WOMEN

KARVENAGAR, PUNE 411052

SAVITRIBAI PHULE PUNE UNIVERSITY

2021-2022



CERTIFICATE

This is to certify that the project report entitled
**“COMPARATIVE ANALYSIS OF DEEPPFAKE DETECTION ALGORITHMS AND
ITS APPLICATIONS”**

Submitted by

RASHMI DESHMUKH	C22018221322 (4321)
DISHA CHAUDHARI	C22018221327 (4312)
MANASI KASANDE	C22018221348 (4348)
PARUL SHARMA	C22018221368 (4374)
TANISHKA PATEL	C22018221370 (4369)

is a bonafide student of this institute and the work has been carried out by her under the supervision of **Prof. Shilpa Pant** and it is approved for the partial fulfilment of the requirement of Cummins College of Engineering for Women, Karvenagar, Pune (an autonomous institute affiliated to Savitribai Phule Pune university.) for the award of the degree of **Bachelor of Technology** (Computer Engineering).

(Prof. Shilpa P. Pant)

Guide,
Department of Computer Engineering

(Dr. Supriya Kelkar)

Head of Department,
Department of Computer Engineering

(Dr. M. B. Khambete)

Principal,
Cummins College of Engineering for Women Pune – 52

Place: Pune

Date: 15/02/2022

Date: September 23, 2021

To,
Disha Chaudhari
Pune, India.

Sub: Offer Letter - Internship

Dear Disha,

Based on your application and subsequent interview that we had with you, we are pleased to offer you the position of '**Project Trainee**', at Inteliment - India as follows;

1. Your internship starts from **27th Sept 2021**.
You will be on internship for period of 7 months from the date of joining the company, which can be extended for a period of 2 weeks depending on your project progress.
2. Work Timings -
In the first month of your internship period, you will be required to report every Thursday at Inteliment Office, Pune. Your work timings would be 9.30 am to 6.30 pm. Virtual working hours will be 4 pm to 8 pm.
3. Stipend is not applicable for internship.
4. An appointment letter will be issued on your date of joining.
5. Breach of NDA -
The Company reserves the right to terminate your association on grounds of misconduct or breach of the terms and conditions of the Undertaking to be furnished by the intern and / or violation of any rules and regulations or standing orders of the Company without giving any notice thereof.
6. On successful completion of project, you are required to submit a copy of the Project Report to the company.
7. You shall be issued a Project Certification on successful completion of the project.
8. You shall be reporting to the assigned Project Guide and maintain the highest degree of professionalism.
9. You are required to submit following documents on your Joining date.
 - i. All educational certificates along with the copies of the latest mark sheets.
 - ii. Experience certificate(s) (If Applicable)
 - iii. Copy of Passport, Aadhar Card, PAN card and 2 passport size color photographs
 - iv. Copy of present/permanent address proof

We look forward to welcoming you as a member of Team Inteliment.

For Inteliment Technologies Pvt. Ltd.

Trupti Pansare
Director - Human Resources

I agree to accept my offer on the terms and conditions mentioned in this offer letter.

Signature: Disha Chaudhari

Date: 23rd September 2021

Date: September 23, 2021

To,
Rashmi Deshmukh
Pune, India.

Sub: Offer Letter - Internship

Dear Rashmi,

Based on your application and subsequent interview that we had with you, we are pleased to offer you the position of '**Project Trainee**', at Inteliment - India as follows;

1. Your internship starts from **27th Sept 2021**.
You will be on internship for period of 7 months from the date of joining the company, which can be extended for a period of 2 weeks depending on your project progress.
2. Work Timings -
In the first month of your internship period, you will be required to report every Thursday at Inteliment Office, Pune. Your work timings would be 9.30 am to 6.30 pm. Virtual working hours will be 4 pm to 8 pm.
3. Stipend is not applicable for internship.
4. An appointment letter will be issued on your date of joining.
5. Breach of NDA -
The Company reserves the right to terminate your association on grounds of misconduct or breach of the terms and conditions of the Undertaking to be furnished by the intern and / or violation of any rules and regulations or standing orders of the Company without giving any notice thereof.
6. On successful completion of project, you are required to submit a copy of the Project Report to the company.
7. You shall be issued a Project Certification on successful completion of the project.
8. You shall be reporting to the assigned Project Guide and maintain the highest degree of professionalism.
9. You are required to submit following documents on your Joining date.
 - i. All educational certificates along with the copies of the latest mark sheets.
 - ii. Experience certificate(s) (If Applicable)
 - iii. Copy of Passport, Aadhar Card, PAN card and 2 passport size color photographs
 - iv. Copy of present/permanent address proof

We look forward to welcoming you as a member of Team Inteliment.

For Inteliment Technologies Pvt. Ltd.

Trupti Pansare
Director - Human Resources

I agree to accept my offer on the terms and conditions mentioned in this offer letter.

Signature: Rashmi Deshmukh

Date: 23rd September 2021

Date: September 23, 2021

To,
Manasi Kasande
Pune, India.

Sub: Offer Letter - Internship

Dear Manasi,

Based on your application and subsequent interview that we had with you, we are pleased to offer you the position of '**Project Trainee**', at Inteliment - India as follows;

1. Your internship starts from **27th Sept 2021**.
You will be on internship for period of 7 months from the date of joining the company, which can be extended for a period of 2 weeks depending on your project progress.
2. Work Timings -
In the first month of your internship period, you will be required to report every Thursday at Inteliment Office, Pune. Your work timings would be 9.30 am to 6.30 pm. Virtual working hours will be 4 pm to 8 pm.
3. Stipend is not applicable for internship.
4. An appointment letter will be issued on your date of joining.
5. Breach of NDA -
The Company reserves the right to terminate your association on grounds of misconduct or breach of the terms and conditions of the Undertaking to be furnished by the intern and / or violation of any rules and regulations or standing orders of the Company without giving any notice thereof.
6. On successful completion of project, you are required to submit a copy of the Project Report to the company.
7. You shall be issued a Project Certification on successful completion of the project.
8. You shall be reporting to the assigned Project Guide and maintain the highest degree of professionalism.
9. You are required to submit following documents on your Joining date.
 - i. All educational certificates along with the copies of the latest mark sheets.
 - ii. Experience certificate(s) (If Applicable)
 - iii. Copy of Passport, Aadhar Card, PAN card and 2 passport size color photographs
 - iv. Copy of present/permanent address proof

We look forward to welcoming you as a member of Team Inteliment.

For Inteliment Technologies Pvt. Ltd.

Trupti Pansare
Director - Human Resources

I agree to accept my offer on the terms and conditions mentioned in this offer letter.

Signature: Manasi Kasande

Date: 23rd September 2021

Date: September 23, 2021

To,
Parul Sharma
Pune, India.

Sub: Offer Letter - Internship

Dear Parul,

Based on your application and subsequent interview that we had with you, we are pleased to offer you the position of '**Project Trainee**', at Inteliment - India as follows;

1. Your internship starts from **27th Sept 2021**.
You will be on internship for period of 7 months from the date of joining the company, which can be extended for a period of 2 weeks depending on your project progress.
2. Work Timings -
In the first month of your internship period, you will be required to report every Thursday at Inteliment Office, Pune. Your work timings would be 9.30 am to 6.30 pm. Virtual working hours will be 4 pm to 8 pm.
3. Stipend is not applicable for internship.
4. An appointment letter will be issued on your date of joining.
5. Breach of NDA -
The Company reserves the right to terminate your association on grounds of misconduct or breach of the terms and conditions of the Undertaking to be furnished by the intern and / or violation of any rules and regulations or standing orders of the Company without giving any notice thereof.
6. On successful completion of project, you are required to submit a copy of the Project Report to the company.
7. You shall be issued a Project Certification on successful completion of the project.
8. You shall be reporting to the assigned Project Guide and maintain the highest degree of professionalism.
9. You are required to submit following documents on your Joining date.
 - i. All educational certificates along with the copies of the latest mark sheets.
 - ii. Experience certificate(s) (If Applicable)
 - iii. Copy of Passport, Aadhar Card, PAN card and 2 passport size color photographs
 - iv. Copy of present/permanent address proof

We look forward to welcoming you as a member of Team Inteliment.

For Inteliment Technologies Pvt. Ltd.

Trupti Pansare
Director - Human Resources

I agree to accept my offer on the terms and conditions mentioned in this offer letter.

Signature: Parul Sharma

Date: 23rd September 2021

Date: September 23, 2021

To,
Tanishka Patel
Pune, India.

Sub: Offer Letter - Internship

Dear Tanishka,

Based on your application and subsequent interview that we had with you, we are pleased to offer you the position of **'Project Trainee'**, at Inteliment - India as follows;

1. Your internship starts from **27th Sept 2021**.
You will be on internship for period of 7 months from the date of joining the company, which can be extended for a period of 2 weeks depending on your project progress.
2. Work Timings -
In the first month of your internship period, you will be required to report every Thursday at Inteliment Office, Pune. Your work timings would be 9.30 am to 6.30 pm. Virtual working hours will be 4 pm to 8 pm.
3. Stipend is not applicable for internship.
4. An appointment letter will be issued on your date of joining.
5. Breach of NDA -
The Company reserves the right to terminate your association on grounds of misconduct or breach of the terms and conditions of the Undertaking to be furnished by the intern and / or violation of any rules and regulations or standing orders of the Company without giving any notice thereof.
6. On successful completion of project, you are required to submit a copy of the Project Report to the company.
7. You shall be issued a Project Certification on successful completion of the project.
8. You shall be reporting to the assigned Project Guide and maintain the highest degree of professionalism.
9. You are required to submit following documents on your Joining date.
 - i. All educational certificates along with the copies of the latest mark sheets.
 - ii. Experience certificate(s) (If Applicable)
 - iii. Copy of Passport, Aadhar Card, PAN card and 2 passport size color photographs
 - iv. Copy of present/permanent address proof

We look forward to welcoming you as a member of Team Inteliment.

For Inteliment Technologies Pvt. Ltd.

Trupti Pansare
Director - Human Resources

I agree to accept my offer on the terms and conditions mentioned in this offer letter.

Signature: Tanishka Patel

Date: 23rd September 2021

ACKNOWLEDGEMENT

The completion of this project would not have been possible without the participation and assistance of a lot of individuals contributing to this project. We as a team, are very grateful to our respectable teacher, Prof. Shilpa P. Pant, whose insightful leadership and depth of knowledge benefited us to complete this project successfully. We collectively thank you for your continuous support and guidance whenever needed.

We would also like to thank Mrs. Yogini Kulkarni for providing us a separate email-id so that we could store our datasets with ease.

We would also like to extend our gratitude towards our project coordinators Prof. Mahendra Deore and Mrs. Meenal Kamlakar for their timely coordination and guidance.

We are grateful to our principal, Dr. M. B. Khambete and our HOD, Dr. Supriya Kelkar for providing us with this great opportunity of putting our ideas forward.

Last but not the least, we are obliged to Inteliment Technologies for providing us with this sponsorship.

NAME OF THE STUDENTS

DISHA CHAUDHARI

RASHMI DESHMUKH

MANASI KASANDE

TANISHKA PATEL

PARUL SHARMA

ABSTRACT

In the recent years' photo shopping has been replaced by Deepfake. Deepfakes are created using Deep learning algorithm and consists of people doing or saying things which weren't done or said by them. These Deepfakes can be created by anyone and doesn't require special skill unlike photo shopping. This is because of the presence of various free tools available online. While most people use this for fun and enjoyment there are many others who have wrong intentions. Spread of mass misinformation, cyber-bullying and online scams are few such examples. Deepfake detection is the need of the hour to stop the spread of such misleading information. Ever since Deepfake was identified as an important problem many people have proposed different ways to detect Deepfake. We have observed that most of the proposed methods consisted of Convolutional Neural Network and Recurrent Neural Network Models. Therefore, we decided to perform comparative study on these two models by training them on a common dataset. Our common dataset is a mix dataset consisting of 3 state of the art Deepfake detection dataset- Deepfake Detection Challenge Dataset, Celeb-DF and Face Forensics++. We have chosen EfficientNet and Long Short Term Memory Network as our Convolutional Neural Network and Recurrent Neural Network architectures. These models were trained using Transfer Learning. In addition, we intend to develop an end to end web application and aim to get a high accuracy by using these methods.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS	i
LIST OF FIGURES	ii
LIST OF TABLES	iii
1. INTRODUCTION.....	1
1.1 OVERVIEW	1
1.2 MOTIVATION	2
1.3 PROBLEM DEFINITION AND OBJECTIVES.....	2
1.4 PROJECT SCOPE AND LIMITATIONS.....	3
1.5 METHODOLOGIES OF PROBLEM SOLVING.....	3
1.5.1 WORKFLOW	3
1.5.2 PARAMETERS IDENTIFIED FOR DEEPFAKE DETECTION	3
2. LITERATURE SURVEY	4
2.1 BACKGROUND OF THE DOMAIN	4
2.2 RESEARCH PAPERS REVIEWED.....	5
3. REQUIREMENTS.....	7
3.1 DESCRIPTION OF REQUIREMENT	7
3.2 SOFTWARE REQUIREMENT SPECIFICATION	7
3.2.1 SCOPE.....	7
3.2.2 FEATURES	7
3.2.3 FUNCTIONAL REQUIREMENTS	7
3.2.4 EXTERNAL INTERFACE REQUIREMENTS.....	8
3.2.4.1 USER INTERFACE	8
3.2.4.2 HARDWARE INTERFACE	8
3.2.4.3 SOFTWARE INTERFACE.....	8
3.2.4.4 COMMUNICATION INTERFACE.....	8

3.2.5 SOFTWARE QUALITY ATTRIBUTES.....	9
3.2.6 NON FUNCTIONAL REQUIREMENTS	9
3.2.6.1 PERFORMANCE REQUIREMENTS	9
3.2.6.2 SAFETY REQUIREMENTS.....	9
3.2.6.3 SECURITY REQUIREMENTS	9
3.2.7 SYSTEM REQUIREMENTS.....	10
3.2.7.1 SOFTWARE REQUIREMENTS	10
3.2.7.2 HARDWARE REQUIREMENTS	10
3.3 SYSTEM IMPLEMENTATION PLAN	11
3.4 USE CASES	12
3.5 ANALYSIS: SDLC MODEL TO BE APPLIED.....	12
4. SYSTEM DESIGN.....	13
4.1 SYSTEM ARCHITECTURE.....	13
4.2 ACTIVITY DIAGRAM.....	14
4.3 ENTITY RELATIONSHIP DIAGRAM.....	15
5. IMPLEMENTATION ASPECTS	16
5.1 OVERVIEW OF PROJECT MODULES.....	16
5.1.1 DATA COLLECTION	16
5.1.2 DATA PRE PROCESSING.....	18
5.1.3 TRANSFER LEARNING.....	19
5.1.4 EFFICIENTNET.....	19
5.1.5 LSTM.....	20
5.1.6 TRAINING AND TESTING.....	21
5.1.7 EXPERIMENTS	22
5.1.7.1 HYPER PARAMETER TUNING FOR EFFICIENTNET	22
5.1.7.2 HYPER PARAMETER TUNING FOR LSTM	23
5.2 ALGORITHM DETAILS	23
5.2.1 FRAME EXTRACTION	23
5.2.2 CREATE CSV	24
5.2.3 CROP VIDEOS	24
5.2.4 DATA LOADING	25

5.2.5 TRAINING	25
5.2.7 TESTING.....	26
5.2.8 FORWARD LOOP OF LSTM	26
6. SOFTWARE TESTING.....	27
6.1 TYPES OF TESTING USED	27
6.2 TEST PLANS.....	27
7. RESULTS	29
7.1 EFFICIENTNET	29
7.2 LSTM	30
7.3 OUTCOMES.....	31
7.4 SCREENSHOTS.....	32
8. CONCLUSION	37
APPENDIX A	38
APPENDIX B	38
APPENDIX C	39
REFERENCES.....	40

LIST OF ABBREVIATIONS

Abbreviation	Full Form
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GAN	Generative Adversarial Network
GPU	General Processing Unit
HOHA	Hollywood Human Actions
UI	User Interface
PC	Personal Computer
HTTP	Hypertext Transfer Protocol
VAE	Variational Auto Encoders
DFDC	Deepfake Detection Challenge
IDE	Integrated Development Environment
SSL	Secure Socket Layer
SDLC	Software Development Life Cycle
FPS	Frames Per Second

LIST OF FIGURES

FIGURE 1 - REAL IMAGE (LEFT) AND ITS DEEPPFAKE (RIGHT).....	1
FIGURE 2 - DEEPPFAKE (RIGHT) AND REAL IMAGE (LEFT)	1
FIGURE 3 DEEPPFAKE CREATION METHODS.....	4
FIGURE 4 USE CASE DIAGRAM	12
FIGURE 5 SYSTEM ARCHITECTURE.....	13
FIGURE 6 ACTIVITY DIAGRAM	14
FIGURE 7 ENTITY RELATIONSHIP DIAGRAM	15
FIGURE 8 SAMPLE VIDEO FROM CELEBDF DATASET	16
FIGURE 9 SAMPLE VIDEO FROM DFDC DATASET.....	17
FIGURE 10 SAMPLE VIDEO FROM FACE FORENSICS ++ DATASET.....	17
FIGURE 11PRE PROCESSING FLOW	18
FIGURE 12 PROPOSED EFFICIENTNET ARCHITECTURE.....	20
FIGURE 13 PROPOSED LSTM ARCHITECTURE	20
FIGURE 14 TRAINING AND TESTING FLOW	22
FIGURE 15 EFFICIENTNET B1 WITH 3 FULLY CONNECTED LAYERS STRUCTURE.....	23
FIGURE 16 VALIDATION LOSS FOR LSTM WITH SEQUENCE LENGTH 40	30
FIGURE 17 TRAINING AND VALIDATION LOSS FOR LSTM WITH SEQUENCE LENGTH 50	31
FIGURE 18 OUTPUT FOR DATA LOADING AND SPLITTING	32
FIGURE 19 CONFUSION MATRIX FOR LSTM WITH SEQUENCE LENGTH 40	32
FIGURE 20 CONFUSION MATRIX FOR LSTM WITH 50 SEQUENCE LENGTH	33
FIGURE 21 CONFUSION MATRIX FOR EFFICIENTNET B1, 3 FULLY CONNECTED LAYERS	33
FIGURE 22 CONFUSION MATRIX FOR EFFICIENTNET B1, 2 FULLY CONNECTED LAYERS	34
FIGURE 23 CONFUSION MATRIX FOR EFFICIENTNET B2, 2 FULLY CONNECTED LAYERS	34
FIGURE 24 CONFUSION MATRIX FOR EFFICIENTNET B3, 3 FULLY CONNECTED LAYERS	35
FIGURE 25 CONFUSION MATRIX FOR EFFICIENTNET B3, 2 FULLY CONNECTED LAYERS	35
FIGURE 26 CONFUSION MATRIX FOR EFFICIENTNET B4 WITH 3 FULLY CONNECTED LAYERS.....	36
FIGURE 27 CONFUSION MATRIX FOR EFFICIENTNET B4 WITH 2 FULLY CONNECTED LAYERS	36

LIST OF TABLES

TABLE 1 TEST CASES.....	27
TABLE 2 EFFICIENTNET RESULTS	29
TABLE 3 LSTM RESULTS	30

1. INTRODUCTION

1.1 OVERVIEW

Computers have become very intelligent and better at simulating reality and deepfake technology is one such example. In recent years, a technology known as “deep fake” has been making headlines for its ability to create believable fake media such as audios, images and videos. The word “deepfake” comes from the technology Deep Learning. Deep learning algorithms are trained to create swaps in faces thereby generating “deep fakes”. Deepfake videos consist of a person doing or saying things which weren’t said or done by that person. Moreover, these videos leave very few traces of manipulation. An example of a deepfake is shown in Figure 1.



Figure 1 - Real Image (Left) and its Deepfake (Right) [1]

Figure 2 - Deepfake (Right) and Real Image (Left)

Earlier, limited access to technologies such as Photoshop meant that the amount of fake media circulating on the internet was comparatively lesser than today. Limited access to such apps also meant that there were limited people who had the skills to generate fake media. The rise of the internet and the availability of free audios, videos and images has completely changed the landscape with regards to generation of fake media. The availability of free tools such as FaceApp, Wombo, Deepfakesweb, etc. [2] [3] [4] means that people can create fake media just by clicking a few buttons. FaceApp is a popular selfie editor which provides a plethora of AI filters, backgrounds and effects to create deepfakes. [4] One can create a deepfake video with ease on Deepfakesweb. [3] Wombo is an AI- powered lip sync app which creates ultra-realistic audio altered deepfakes. [2] Tools like these can open the door to more creative ways of

filmmaking when it comes to visual arts and generating visual effects in movies. These tools can also be used for light hearted fun and enjoyment. However, there are many others who misuse deepfakes to cause harm. Spreading fake news, cyber bullying, defamation of public figures, revenge porn and internet scams are more commonplace now. [5] [6] [7] [8] Most recently, deepfakes were weaponized to spread fake news and propaganda during the Ukraine-Russia war. Videos of some pro-Russian Ukraine citizens were viral on social media which were later found out to be deepfake. [9]

1.2 MOTIVATION

Deepfakes create a major social problem, especially for celebrities as their photos and videos are publicly available on the internet. It is evident that deepfakes circulated with a malicious intent on the internet can lead to dangerous consequences. Developing tools to detect such deepfakes is vital. Moreover, in a world where digital media like video, audio and images can be presented as evidences to prove or disprove something, it becomes doubly important to verify the authenticity of such media. Deepfake detection tools can help people like fact checkers and digital forensic experts in verifying the authenticity of such viral videos. These tools can also be integrated with social media platforms as social media is the most conducive environment for the spread of such fake media.

1.3 PROBLEM DEFINITION AND OBJECTIVES

Keeping the overall background of deepfakes and the importance of developing tools to detect them in mind, our project has the following objectives:

- i. Develop a Convolutional Neural Network (CNN) model for deepfake detection.
- ii. Develop a Recurrent Neural Network model (RNN) for deepfake detection.
- iii. Analyze the performance of the developed CNN and RNN models.
- iv. Deploy the best performing model

1.4 PROJECT SCOPE AND LIMITATIONS

The goal of our project includes developing CNN and RNN models that can classify videos into either of the two categories – deepfake or real. Also, we intend to perform a comparative analysis of both approaches which will help us determine the better approach among the two. One of the limitation of our project is that we won't be able to train our model on a huge and large scale dataset due to the size of computational resources available to us. Also, we only focus on detecting deepfake videos and do not focus on detecting manipulated audios.

1.5 METHODOLOGIES OF PROBLEM SOLVING

1.5.1 WORKFLOW

To solve this problem, we follow the typical workflow of deep learning lifecycle:

- Dataset gathering
- Dataset pre processing
- Model building
- Model Training
- Model Evaluation
- Model Deployment

1.5.2 PARAMETERS IDENTIFIED FOR DEEPPFAKE DETECTION

As mentioned in section 2, researchers have identified various parameters which can help in the identification of deepfakes. These parameters are:

- Glitches induced in outlines of faces
- Irregular distance between eyes
- Variation in skin tone
- Blinking of eyes – deepfakes generally have a high rate of blinking than normal people
- Mismatch in the agedness of skin and eyes
- Size of lips, ears, nose and eyes with respect to the whole face

2. LITERATURE SURVEY

2.1 BACKGROUND OF THE DOMAIN

To understand and develop techniques for deepfake detection, it is paramount to understand the intuition and logic behind deepfake creation. Deepfakes themselves are created using two other deep learning techniques – Generative Adversarial Networks (GANs) and Variational Auto Encoders. (VAEs)

GANs comprise of a discriminator and a generator. The discriminator's task is to classify the video into fake or real. The generator's task is to actually manipulate the real video to a fake video. A deepfake is created when the discriminator starts classifying a real video as fake, which indicates that the generator has created the video in such a way that the discriminator is not able to correctly classify the video. Some popular methods for deepfake creation using GANs are shown in Figure 2.

VAEs on the other hand consist of two encoder-decoder pairs. The encoder's task is to learn the minute features of the faces which are then passed on to the decoder. The decoder then tries to reconstruct a totally new face from the information given by the encoder. Some popular methods used for deepfake creation using VAEs are shown in Figure 2.

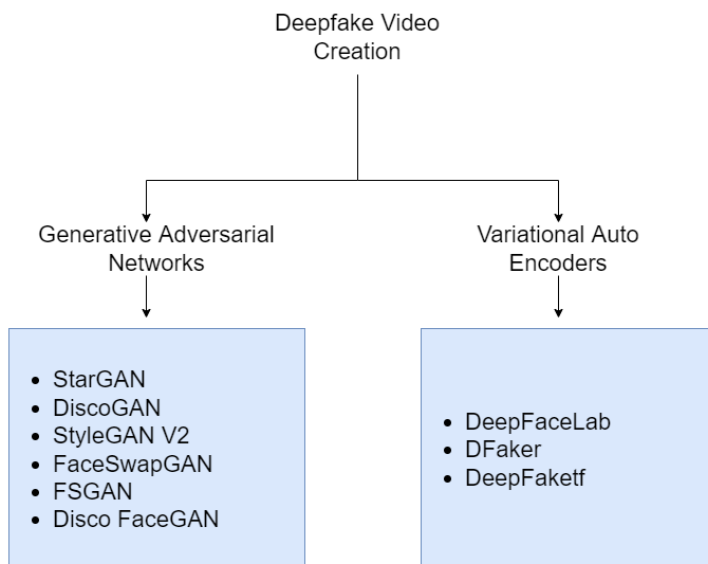


Figure 3 Deepfake creation methods

2.2 RESEARCH PAPERS REVIEWED

Most deepfake detection methods are based on CNN and RNN. Multiple CNN and RNN techniques for deepfake detection have been developed in the last few years. [10] tried an ensemble of different CNN models and achieved promising results. The authors considered Efficientnet B4 as the base model and obtain two different ensemble models from it. The ensemble models are obtained by making use of two concepts – attention mechanism and Siamese training. Both models were trained and tested on two datasets, FaceForensics++ [11] and Kaggle’s Deepfake Detection Challenge dataset. [12] It has been observed that deepfakes often have inconsistencies in facial features. [13] proposed a model that analyses the frames of the videos using CNNs to detect such inconsistencies in the videos. A pretrained VGG16 model [14] was used to train the model using transfer learning on the CelebDF dataset. [15] CNNs have been widely used to detect deepfake videos since they are capable of extracting frame level features. In order to speed up the development of deepfake detection techniques, Facebook hosted a Deep Fake Detection Challenge (DFDC). The models with the highest accuracy used EfficientNet models along with transfer learning. Based on this observation, [16] decided to perform a comparative study of different EfficientNet models for deepfake detection. For this, DFDC dataset was used. [12] Highest accuracy was obtained for the Efficientnet B5 model, followed by Efficientnet B4 and Efficientnet B6. There has been a rise in the use of transformers in the field of computer vision. Many deep learning models having a transformer based architecture have managed to deliver competitive and accurate results. [17] used this to their advantage and proposed a model composed of EfficientNet and Vision Transformers. The authors extracted faces from the videos with the help of MTCNN, which is a face detector. The faces are further passed as an input to the proposed model. The proposed architecture consists of EfficientNet B0 as a feature extractor. The features extracted by EfficientNet B0 are analyzed by the vision transformer. Eventually, the vision transformer outputs the probability of face manipulation. The proposed architecture was trained and tested on DFDC [12] and FaceForensics++ [11] datasets.

Along with CNNs, RNNs can be used to detect inconsistencies in frames of the videos. [18] proposed a ConvLSTM model which uses both these properties of CNNs and RNNs. The proposed model adopted the Inception V3 model [19] as a feature extractor which was followed

by a LSTM network to analyze temporal inconsistencies in the sequences. The model was trained on 300 deepfake videos collected from multiple websites and 300 real videos collected from the HOHA dataset. [20] also followed the approach of using CNN for feature extraction and LSTM for analyzing the extracted features. ResNext was used as a feature extractor followed by a 2 node neural network consisting of a LSTM layer and a fully connected layer to classify the videos. The authors used a mixed dataset consisting of videos from DFDC [12], CelebDF [15] and FaceForensics++. [11] So far, it is clear that using RNNs for deepfake detection is a sound strategy. However, every RNN architecture proposed for deepfake detection contains a CNN component wherein CNN performs feature extraction and RNNs perform the actual deepfake detection. With many high quality CNN models available in literature, it is important to know the best CNN-RNN combination for the task of deepfake detection. [21] tried to find out the answer to this question by experimenting with various Recurrent-Convolutional models for deepfake detection. The authors tried various combinations of ResNet50 and DenseNet with RNNs and concluded that DenseNet with alignment used in conjunction with bi-directional RNNs give the best results. Along with deepfake videos, it is important to detect deepfake audios as well. [22] Proposed a solution capable of detecting both deepfake videos and audio-manipulated deepfakes. On the basis of cost functions, the authors have combined latent CNN architectures with bidirectional RNNs. From audios and videos, the latent representations are chosen in such a way that the information is meaningful for the model. These representations were passed to a Recurrent Neural Network. The RNN was able to identify faults within different frames as well as the whole sequence of frames. For video detection, the authors used the FaceForensics++ [11] and Celeb-DF [15] video datasets and ASVSpooof 2019 Logical Access audio datasets [23] for deepfake audio detection.

3. REQUIREMENTS

3.1 DESCRIPTION OF REQUIREMENT

The purpose is to perform a comparative study between EfficientNet models and LSTM models for the task of deepfake detection. We also aim to get accurate results for deepfake detection with both kinds of models. The one with the highest accuracy will be deployed as a flask web application.

3.2 SOFTWARE REQUIREMENT SPECIFICATION

3.2.1 SCOPE

The purpose of this project is to categorize input videos as deepfake or not. The deep learning models will be trained and tested on a mixed dataset. Deepfakes not only include videos, but also audios, text and images. In this project, we aim to detect only deepfake videos. Researchers have proposed many algorithms to detect deepfakes, but we aim to perform a comparative analysis of EfficientNet and LSTM for deepfake detection. We will be implementing the best performing algorithms as a web application.

3.2.2 FEATURES

The basic features include a web page where a user can either upload a video from his computer or provide a link where the video is present on the Internet. Then the user will be given an option to check the video. The results will be displayed on the screen stating whether the video is a deepfake or not.

3.2.3 FUNCTIONAL REQUIREMENTS

There are 2 main functions that will be performed by the system-

- i. **Upload the video** - The user or client will be given an option to either upload a video from his local computer drive or he can provide a link to the Internet where the video is located.
- ii. **Check if it is deepfake or not** - A button will be provided which when clicked will provide a result to the user on the screen. It will predict if the video is deepfake or not

and will provide the result on the webpage. The user will also be given an option of uploading another video if he wants to check for any other video.

3.2.4 EXTERNAL INTERFACE REQUIREMENTS

Currently, we do not plan to use a database because the application is going to be just for Deepfake detection and no such storage of data is needed.

3.2.4.1 USER INTERFACE

The User Interface is simple. It will contain an “upload a video option” where the user will input the location of the video i.e., either from his local computer or on the Internet. Another button named “Check” will allow the user to check if the video is a deepfake or not. The result will be displayed on the User Interface.

3.2.4.2 HARDWARE INTERFACE

A computer with good processing power is needed as high amount of video data will be processed. For loading the web-app, any device compatible with the web-browser will suffice.

3.2.4.3 SOFTWARE INTERFACE

Operating System: Windows 7 and above

Programming Language: Python 3.0

Frameworks: PyTorch, Flask

IDE: Google Colab Pro

Storage Platform: Google drive

Libraries: OpenCV, face_recognition, pandas, os, glob

3.2.4.4 COMMUNICATION INTERFACE

There will be a front end and a backend for the application. HTTPs protocol will be used.

3.2.5 SOFTWARE QUALITY ATTRIBUTES

- **Usability** - The system should be easy to use. The user should easily be able to understand how to upload the video. All in all, the UI should be user friendly.
- **Adaptability** - The system should be accessible on any platform such as a mobile phone, laptop, PC or a tablet. It should also load on any operating system such as Windows or Linux.
- **Availability** - The user should be able to access the website anywhere and anytime
- **Correctness** - The classification of deepfake and original videos should be correct. It should not call a deepfake video as original or vice versa.
- **Reliability** - This system will be developed with the help of machine learning and deep learning techniques and hence there is no measurable reliability percentage as such.

3.2.6 NON FUNCTIONAL REQUIREMENTS

3.2.6.1 PERFORMANCE REQUIREMENTS

1. The website is fast and reliable, doesn't keep the user waiting.
2. Model should be able to correctly classify videos into their right category 7 out of 10 times.
3. When the user clicks on the upload button the video should be uploaded within 30 seconds (If the user has uploaded it from his local drive.)
4. When the user clicks on the check button, the result should be displayed within 10 seconds justifying if the video is deepfake or not.

3.2.6.2 SAFETY REQUIREMENTS

The videos uploaded for deepfake detection shouldn't be used for any other purpose.

3.2.6.3 SECURITY REQUIREMENTS

Website must run using HTTPS protocol.

SSL Certificate is mandatory.

3.2.7 SYSTEM REQUIREMENTS

3.2.7.1 SOFTWARE REQUIREMENTS

- Python
- OpenCV
- Flask
- Google Colab Pro
- Google Drive
- PyTorch
 - Transforms
 - Torch Vision
 - Dataset Class
 - DataLoader Class
 - Torch.nn
- Scikit Learn
- Seaborn
- Glob
- OS
- Pandas
- Matplotlib
- Face_recognition

3.2.7.2 HARDWARE REQUIREMENTS

GPU support – We will be using the GPU provided by Google Colab Pro. Colab Pro provides three GPUs, and one out of the three is allocated dynamically on runtime. The GPUs are:

- i. Nvidia Tesla K80
- ii. Tesla P100
- iii. Nvidia T4 Tensor Core GPU

3.3 SYSTEM IMPLEMENTATION PLAN

For the system the plan is divided into the following steps:

1. Obtain Dataset- In this step we will get the dataset that contains the training data. We will train the model on this obtained dataset only. The model can be then tested on any other dataset.
2. Data Analysis
3. Data Transformation
4. Handling missing and noisy data
5. Data Pre-processing
6. Data Review
7. Data Splitting in training, validation and testing data.
8. Loading testing data.
9. LSTM and EfficientNet training
10. Calculate error
11. Prediction of fake and real video

3.4 USE CASES

For this system, the client side will have primarily 2 use cases: upload and check. In the CNN model there will be 5 main use cases i.e., Data exploration, pre-processing, LSTM, Load trained model and Prediction. The figure given below shows the use case diagram for the system-

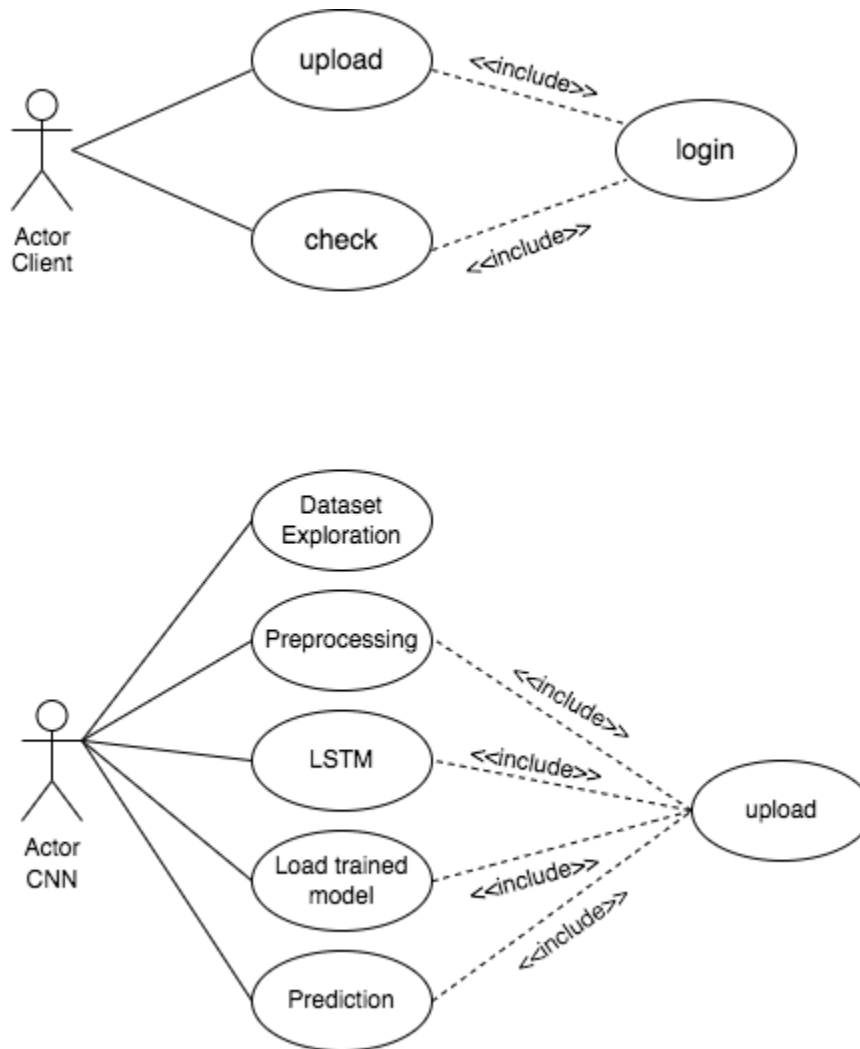


Figure 4 Use case Diagram

3.5 ANALYSIS: SDLC MODEL TO BE APPLIED

We will be applying iterative model for the development of this project.

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

A dataset is created with well labelled videos after which the dataset is pre-processed and stored in a pre-processed database. Then the data is loaded in the LSTM model for training and to make predictions. Figure 5 shows the system architecture diagram of our deepfake detection system.

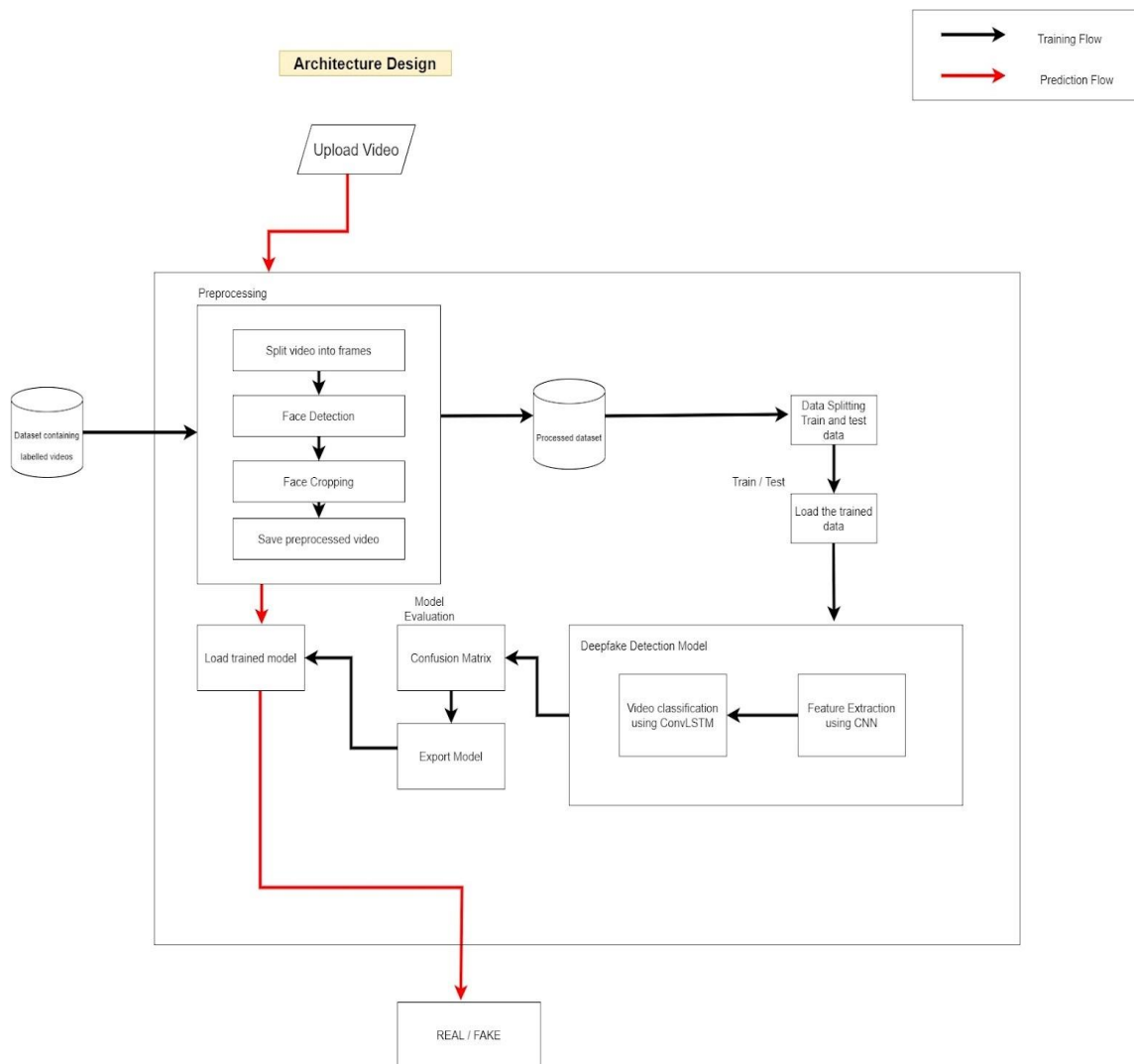


Figure 5 System Architecture

4.2 ACTIVITY DIAGRAM

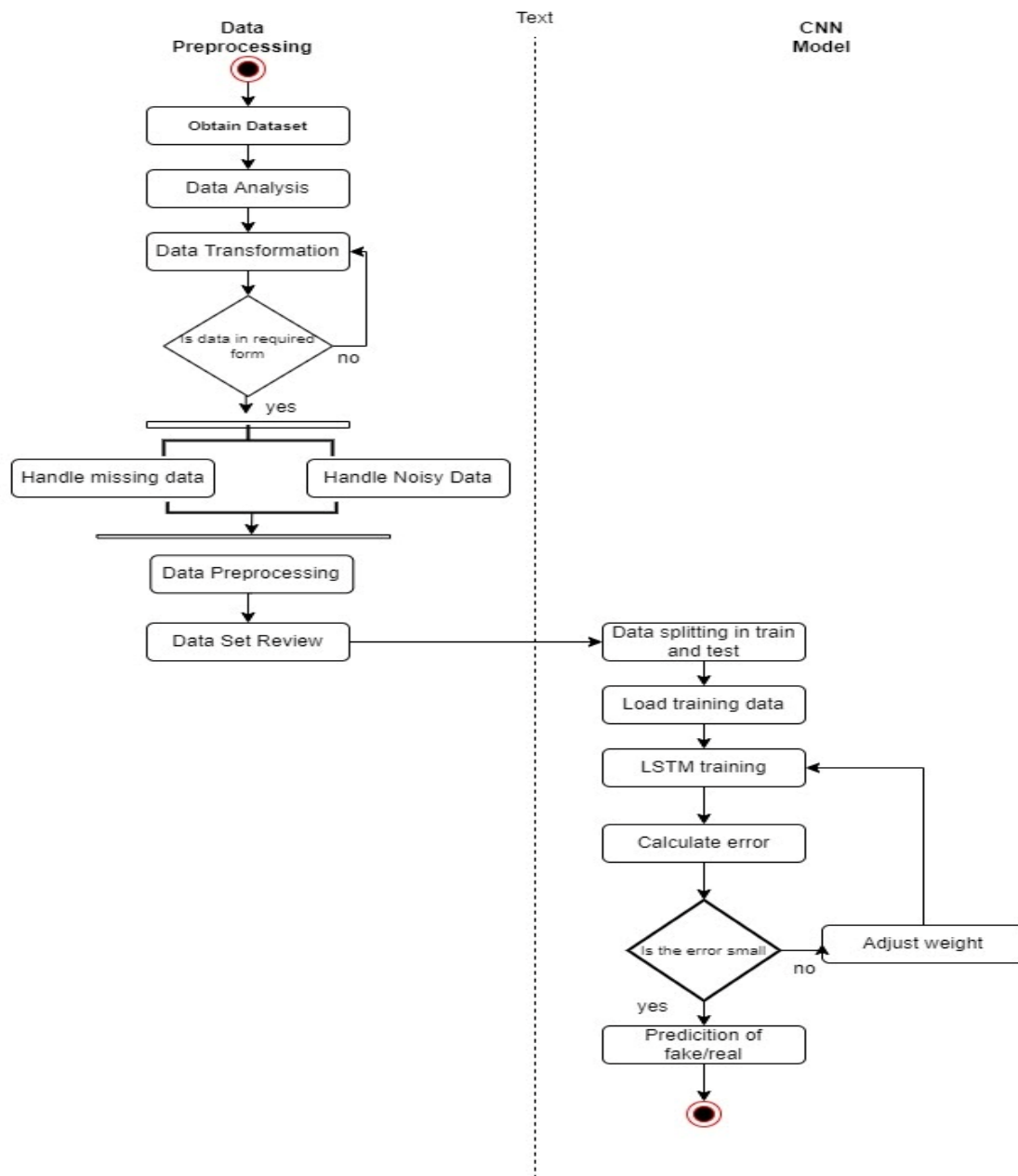


Figure 6 Activity Diagram

4.3 ENTITY RELATIONSHIP DIAGRAM

Figure 7 given below shows the entity relationship diagram of the system- The entities here are User and Video. The user uploads a video. The video entity has some attributes such as FPS (Frames per second), Size, Duration and extension i.e., .avi or .mov.

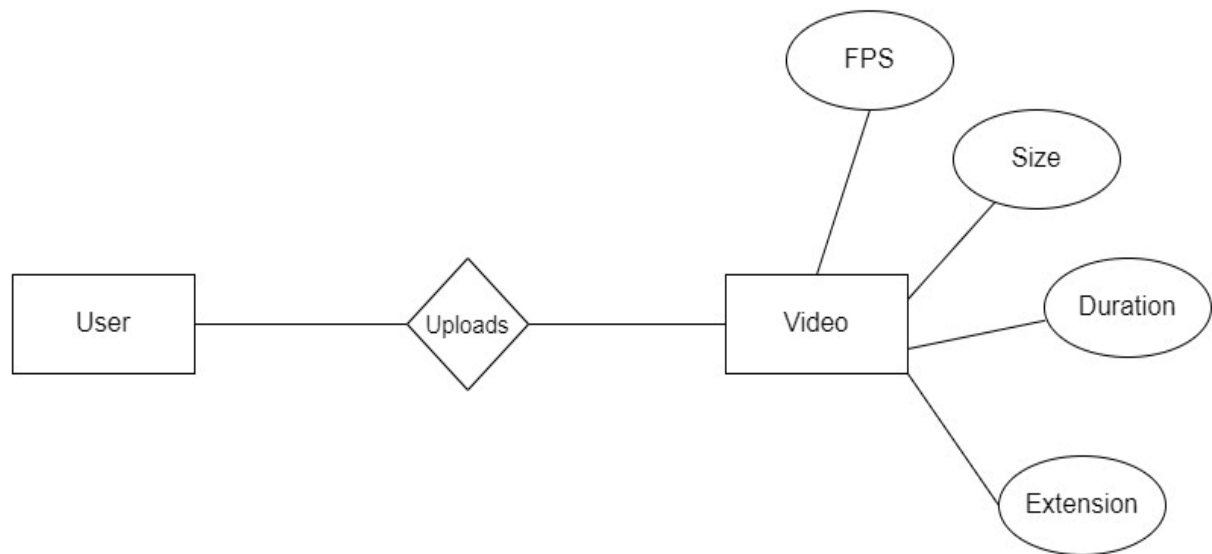


Figure 7 Entity Relationship diagram

5. IMPLEMENTATION ASPECTS

5.1 OVERVIEW OF PROJECT MODULES

5.1.1 DATA COLLECTION

As mentioned in Section 2, there are various deep learning methods which are used to create deepfake videos. Hence, it is important that our deepfake detection model is trained on videos which have been created by various deepfake creation methods instead of training it on just one method. This will help us in improving the testing accuracy of the model. In order to train the model on a variety of deepfake videos, our dataset consists of videos from three different state of the art deepfake detection datasets – CelebDF [14], DFDC [11] and FaceForensics ++. [10]

CelebDF consists of deepfake videos of celebrities and their corresponding real videos. The real videos in these dataset are obtained from YouTube interviews of 59 different celebrities belonging to a varied range of age, gender and ethnicity. The dataset has two versions – CelebDF v1 and CelebDF v2. We have chosen the v1 dataset and its structure is as follows:

- 158 Real videos
- 795 Fake videos



Figure 8 Sample video from CelebDF Dataset

The Deepfake Detection Challenge Dataset (DFDC) is currently the largest dataset that is available for deepfake detection. It has over 100,000 total videos created from 3,426 paid actors. The deepfake videos are created using different methods. This ensures that the models are trained on a wide range of methods. The size of the whole dataset is 470 GB. However, the dataset is also available as 50 smaller chunks of approximately 10 GB each. Keeping in mind the computational resources and storage available to us, we have included the 25th chunk in our dataset and randomly chosen 921 videos from the said chunk.

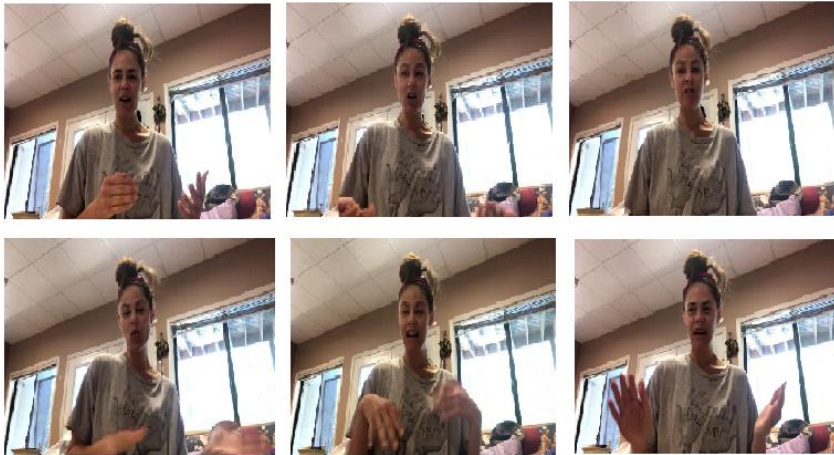


Figure 9 Sample Video from DFDC dataset

FaceForensics++ is also created using four deepfake creation methods - Face2Face, FaceSwap, DeepFakes, and NeuralTextures. The structure of the dataset is as follows:

- 363 Real Videos
- 3076 Fake Videos

In our dataset, we include all 363 real videos and randomly choose 507 fake videos. Our final dataset consists of 2731 videos out of which 737 are real videos and the remaining are fake.



Figure 10 Sample video from Face Forensics ++ dataset

5.1.2 DATA PRE PROCESSING

Preprocessing

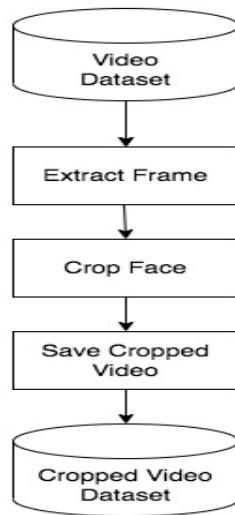


Figure 11 Pre Processing flow

Preprocessing includes extracting frames from the videos, detecting faces from those frames, cropping the faces from the frames and finally combining the cropped images into a video. The duration videos in our dataset ranges from 10 seconds to 30 seconds. If we decide to detect each and every frame from our video, the number of frames generated would prove a bit difficult to handle for our model. For example, consider that each video is of duration 10 seconds with a fps of 60. Then the total number of frames in our dataset would be:

$$\begin{aligned} & 2731 \times \text{duration} \times \text{fps} \\ &= 2731 \times 10 \times 60 \\ &= 16,38,600 \end{aligned}$$

It doesn't make computational sense to handle these many amount of frames while preprocessing. In order to reduce the computational complexity, we are only considering the first 150 frames of each video. The frames are extracted at a rate of 60 frames per second. The dimensions of the cropped video are 500x500. We have extracted the frames in a sequential manner because the order of frames is important for our problem. Next, in order to normalize our dataset, we converted the frame to a tensor and then the tensor was normalized with the mean and standard deviation of the ImageNet dataset.

5.1.3 TRANSFER LEARNING

While practically implementing the model, we can implement transfer learning for detecting Deepfakes. Transfer learning uses pre-trained weights of the neural network for training a fine tuned version of same model on different dataset for some specific application. The model is usually trained on a very large dataset and implementing this pre-trained model on our comparatively smaller dataset improves the performance and reduces the time required for training. Various models like XceptionNet, ResNext, MesoNet, EfficientNet are now available to reuse for specific applications. We have implemented EfficientNet & LSTM using transfer learning on our fine-tuned models for Deepfake Detection. In our solution, we freeze the weights of the convolutional layers of the EfficientNet Model and fine tune the fully connected layers according to our requirement.

5.1.4 EFFICIENTNET

EfficientNet is a CNN architecture and a family of 8 CNN models. The model is designed in such a way that all dimensions of depth, width and resolution are scaled uniformly using a simple compound coefficient. [23] Efficientnet is trained on the ImageNet dataset consisting of 1000 classes. Results have shown that EfficientNet models consistently achieve better accuracy while using lesser parameters when compared with other state of the art CNN models like ResNet, VGGNet, InceptionNet and DenseNet. Moreover, EfficientNet models perform admirably on other transfer learning datasets – notably CIFAR-100 (91.7%), Flowers (98.8%). [23] As EfficientNet is trained on the ImageNet dataset, its last fully connected layer outputs results for 1000 classes. We fine tune our EfficientNet model for 2 classes, as our goal is to classify a video into fake or real. Also, we develop 4 fine-tuned EfficientNet models based on EfficientNet B1, B2, B3 and B4.

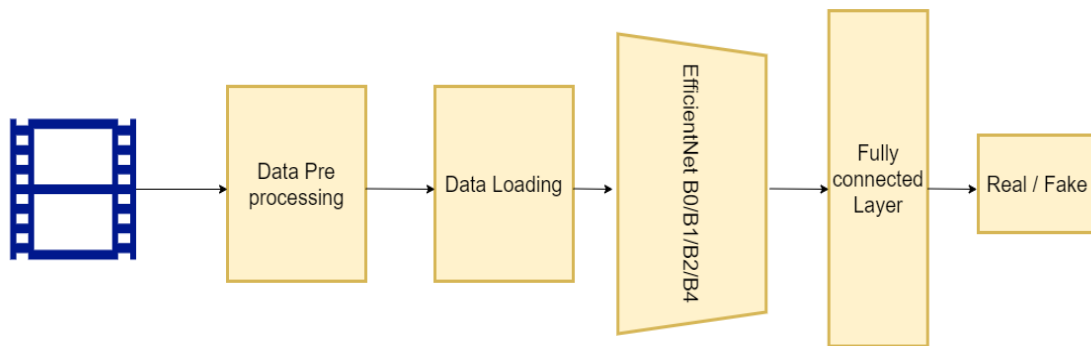


Figure 12 Proposed EfficientNet architecture

5.1.5 LSTM

LSTM is a unique kind of Recurrent Neural Network as it handles long term dependencies and also solves the vanishing gradient problem of other RNNs. [24] Because of this, LSTMs are mainly used in learning dependencies and patterns in large sequences. Since a video is a sequence of frames, LSTMs can perform well on the task of deepfake video detection. We propose a convolutional-LSTM model to detect deepfake videos. EfficientNet B4 is used for feature extraction followed by a 1792-wide LSTM unit for sequence processing. The width of the LSTM unit was chosen as 1792 because EfficientNet B4 outputs a feature vector of the same number of features. The LSTM unit is followed by a fully connected layer to classify the videos as real or fake. The fully connected layer has 1792 input neurons and 2 output neurons.

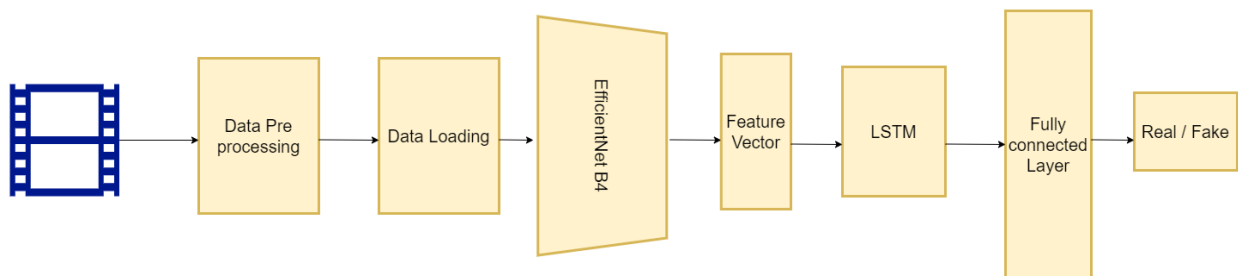


Figure 13 Proposed LSTM architecture

5.1.6 TRAINING AND TESTING

First, we check if the cropped video dataset has corrupted videos and remove them. Then, the cropped video dataset is split into three sets – training set, validation set and testing set in the 80:10:10 ratio. Three corresponding instances of data loader are created for training, testing and validation. The data loader extracts frames from the cropped videos and stacks them together to create the corresponding tensor of the video. The stacked tensor and label of the video is loaded in the model. Before loading the data in the model, the frames are cropped again to suit the input requirements of our models:

- LSTM – 112x112
- EfficientNet B1 – 240x240
- EfficientNet B2 – 260x260
- EfficientNet B3 – 300x300
- EfficientNet B4 – 380x380

For both EfficientNet and LSTM, we train our model on 20 epochs with a learning rate of 0.00001. We use the Adam optimizer for reducing the loss and modifying the weights during the learning process. To avoid overfitting, a weight decay of 0.00001 is used. Cross Entropy Loss is used to measure the loss after every epoch. For training, we use a mini-batch size of 4. After training and validation is concluded, we save the model and test it on our testing dataset.

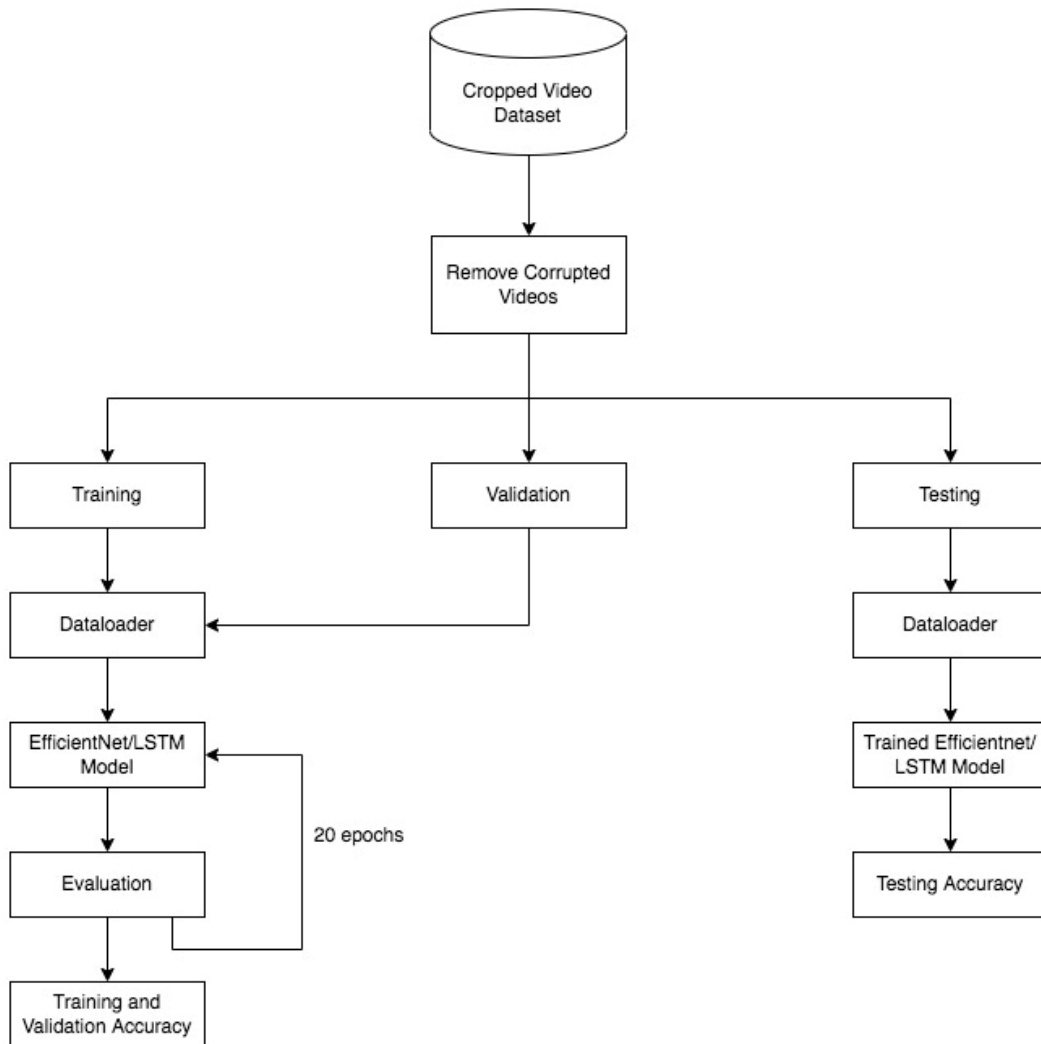


Figure 14 Training and Testing flow

5.1.7 EXPERIMENTS

5.1.7.1 HYPER PARAMETER TUNING FOR EFFICIENTNET

For all four EfficientNet models, we perform hyper parameter tuning at the fully connected layers. Each EfficientNet model has two variations – one with three fully connected layers and the second with two fully connected layers. The number of input and output neurons for the first variation where the left hand term denotes input neurons and right hand term denotes output neurons:

- Layer 1: (in_features, 512)
- Layer 2: (512, 128)

- Layer 3: (128,2)

Where in_features represents the number of features output by the CNN layers of the EfficientNet model used. The number of input and output neurons for the second variation are:

- Layer 1: (in_features, 512)
- Layer 2: (512,2)

```
Loaded pretrained weights for efficientnet-b1
Sequential(
  (0): BatchNorm1d(1280, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (1): Linear(in_features=1280, out_features=512, bias=True)
  (2): ReLU()
  (3): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (4): Linear(in_features=512, out_features=128, bias=True)
  (5): ReLU()
  (6): BatchNorm1d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (7): Dropout(p=0.4, inplace=False)
  (8): Linear(in_features=128, out_features=2, bias=True)
)
```

Figure 15 EfficientNet B1 with 3 fully connected layers structure

5.1.7.2 HYPER PARAMETER TUNING FOR LSTM

For LSTM, we perform hyper parameter tuning on the sequence length. The sequence length refers to the number of frames passed to the model. Our hypothesis is that more the number of frames passed to the model, greater the accuracy of the model. Hence, we train our model on two different sequence lengths – 40 and 50.

5.2 ALGORITHM DETAILS

5.2.1 FRAME EXTRACTION

1. Input path of the video
2. Create an open CV VideoCapture object
3. While the object is open:
 - 3.1. Read frame from VideoCapture object
 - 3.2. If the frame is read successfully
 - 3.2.1. Yield frame

5.2.2 CREATE CSV

1. Store the path of the folder where all the videos are stored in a path variable
2. Create a python list of all videos in the folder by passing the path variable to glob.glob()
3. Names = []
4. Labels = []
5. for video name in the list
 - 5.1. append video name to names[]
 - 5.2. extract folder name from path of video name
 - 5.3. if folder name == FF_REAL or DFDC_REAL or CELEB_REAL
 - 5.3.1. append “Real” to labels []
 - 5.4. if folder name == FF_FAKE or DFDC_FAKE or CELEB_FAKE
 - 5.4.1. append “Fake” to labels []
6. Create a pandas data frame whose columns are labels[] and names []
7. Convert data frame to csv

5.2.3 CROP VIDEOS

1. Store the path of the folder where all the videos are stored in a path variable
2. Create a python list of all videos in the folder by passing the path variable to glob.glob()
3. Store output directory path in another variable
4. For each video in folder
 - 4.1. If file already exists in output directory
 - 4.1.1. Continue
 - 4.2. Create a video writer object with fps 60 and dimensions 500x500
 - 4.3. For each frame of video
 - 4.3.1. Extract face
 - 4.3.2. Crop face
 - 4.3.3. Write cropped face to video writer object

5.2.4 DATA LOADING

Input to the data loader – list of videos, labels, sequence length, transforms

- 1.1. For every video in list of videos
- 1.2. Retrieve label of that video from the labels list
- 1.3. Initialize an empty list to store frames
- 1.4. For every frame of video
 - 1.4.1. Transform the frame
 - 1.4.2. Append the frame to the frames list
- 1.5. Slice the frames list such that length of list is equal to sequence length
- 1.6. Convert frames list to a stack of tensors
- 1.7. Return the stack of tensor and its corresponding label

5.2.5 TRAINING

Input to the training function –number of epochs, data loader, model, criterion, optimizer

- 1.1. For i in number of epochs
 - 1.1.1. Call model.train() function
 - 1.1.2. Initialize variable to store loss and accuracy
 - 1.1.3. For input and target in batch
 - 1.1.3.1. Output = model(input)
 - 1.1.3.2. Calculate cross entropy loss using criterion
 - 1.1.3.3. Calculate accuracy
 - 1.1.3.4. Set gradients to zero
 - 1.1.3.5. Perform back propagation
 - 1.1.3.6. Update weights of network
 - 1.1.4. Save the model in a .pt file

5.2.7 TESTING

Input to the training function – number of epochs, data loader, model, criterion

1.2. For i in number of epochs

1.2.1. Call `model.eval()` function

1.2.2. Initialize variable to store loss and accuracy

1.2.3. For input and target in batch

1.2.3.1. `Output = model(input)`

1.2.3.2. Calculate cross entropy loss using criterion

1.2.3.3. Calculate accuracy

5.2.8 FORWARD LOOP OF LSTM

Input to the forward loop – stack of tensor with dimensions

(batch size, sequence length, channels, height, width)

1. Reshape tensor. New dimensions – *(batch size * sequence length, channels, height, width)*
2. Pass tensor to EfficientNet B4 for feature extraction. EfficientNet B4 returns a feature vector of 1792 features.
3. Perform average pooling of feature map
4. Reshape tensor to dimensions *(batch size, sequence length, 1792)*
5. Pass the reshaped tensor to lstm model
6. Pass output of lstm to linear layer
7. Return output of linear layer and feature map

6. SOFTWARE TESTING

6.1 TYPES OF TESTING USED

Functional Testing

1. Unit Testing
2. Integration Testing
3. System Testing
4. Interface Testing

Non-functional Testing

1. Performance Testing
2. Load Testing
3. Compatibility Testing

6.2 TEST PLANS

Table 1 Test cases

ID	Test case name	Description	Steps	Expected Result	Actual Result
1	Accept training video	To verify that the video has the correct size and extension	Data loaded in training model	Upload successful	Upload successful
2	Accept Training video	To verify that the video has the correct size and extension	Data loaded in training model	Upload successful	Upload unsuccessful
3	Extract two faces	Verify if all faces have been	Face extraction	2 faces detected	Two faces detected

		extracted successfully			
4	Extract two faces	Verify if all faces have been extracted successfully	Face extraction	2 faces detected	Only one face detected
5	Fake video	The video uploaded is deepfake	Video uploaded and evaluated by the model	Video is deepfake	Video is deepfake
6	Incorrect Result	Model gives the wrong result	Video uploaded and evaluated by the model	1. Fake 2. Real	1. Real 2. Fake
7	Wrong extension	User has uploaded video of unsupported extension	User uploads video, extension is checked	User uploads video of mp4 extension only	Unsupported extension

7. RESULTS

7.1 EFFICIENTNET

Table 2 shows the results obtained for EfficientNet experiments. Highest accuracy was obtained for EfficientNet B2 with two fully connected layers. This is in contrast to results obtained for the ImageNet dataset. For the ImageNet dataset, EfficientNet B4 had the highest accuracy among EfficientNet B1, B2, B3 and B4. This shows that there can be difference in the results for transfer learning and results for pre trained models. We also observe that each EfficientNet model with two fully connected layers gives a higher accuracy than the corresponding model with three fully connected layers.

Table 2 Efficientnet results

Model	Number of fully connected layers	Training loss	Validation loss	Testing Accuracy
B1	3	0.000288	0.002155	67.77%
	2	0.000272	0.002164	70.00%
B2	3	0.000297	0.002052	74.44%
	2	0.000274	0.002056	75.55%
B3	3	0.000295	0.002371	67.03%
	2	0.000268	0.002003	71.48%
B4	3	0.000288	0.002064	70.37%
	2	0.000271	0.002282	72.96%

7.2 LSTM

Table 3 shows the results obtained for LSTM. Highest accuracy was obtained for LSTM model with a sequence length of 50. This confirms our hypothesis that the accuracy increases when the sequence length is higher.

Table 3 LSTM Results

Sequence Length	Training Accuracy	Validation Accuracy	Testing Accuracy
40	93.46%	74.34%	76.66%
50	97.12%	76.20%	80.37%

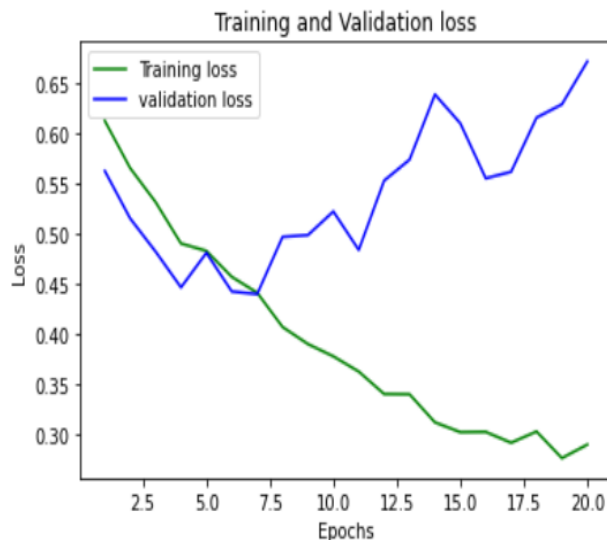


Figure 16 Validation loss for LSTM with sequence length 40

Figure 16 and Figure 17 depict the training and validation losses for LSTM with sequence length 40 and 50 respectively. As one can observe from the graph, the gap between the values of training and validation losses indicates that the model is slightly overfitting the data. A possible reason for this is the imbalance between the real and fake videos in the dataset. There is scope for increasing the number of videos in the dataset, so that we can obtain a balanced dataset with a large number of videos.

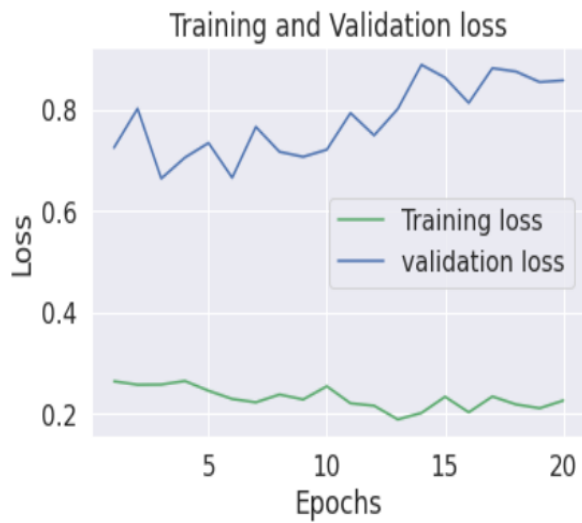


Figure 17 Training and Validation loss for LSTM with sequence length 50

7.3 OUTCOMES

Both EfficientNet and LSTM show fairly good accuracies for deepfake detection. As LSTM has a higher accuracy than EfficientNet, it can be preferred for the task of deepfake detection. One possible explanation for this is that RNNs are better suited for analyzing sequences, which is why LSTM models were able to predict more accurately than EfficientNet.

7.4 SCREENSHOTS

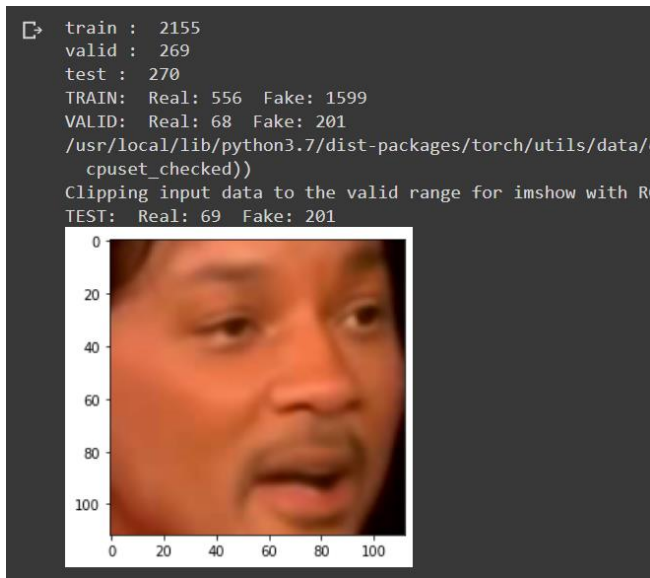


Figure 18 Output for Data loading and splitting

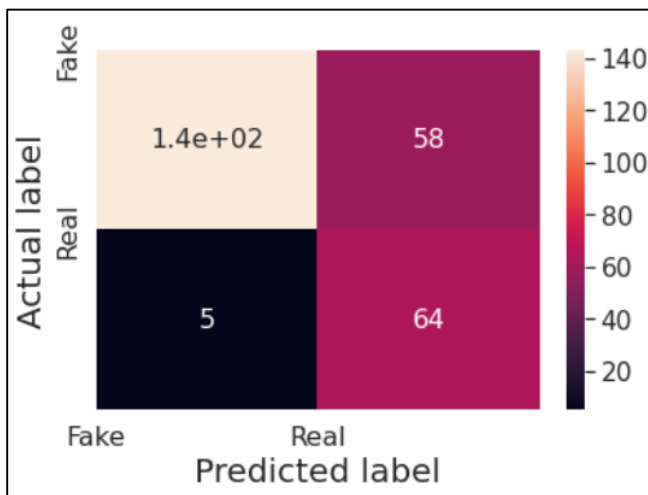


Figure 19 Confusion matrix for LSTM with sequence length 40

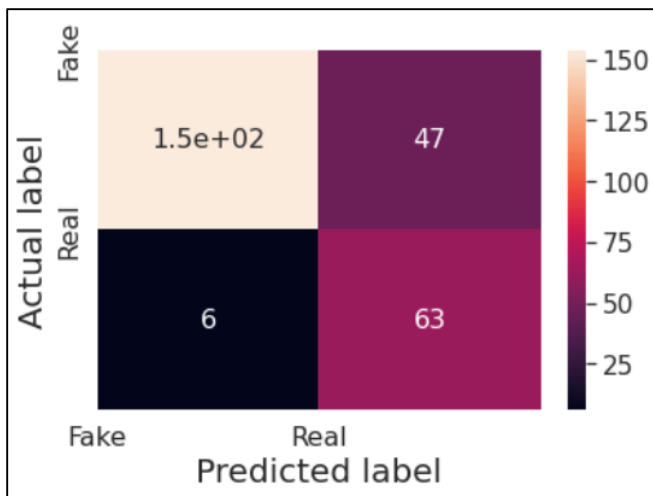


Figure 20 Confusion matrix for LSTM with 50 sequence length

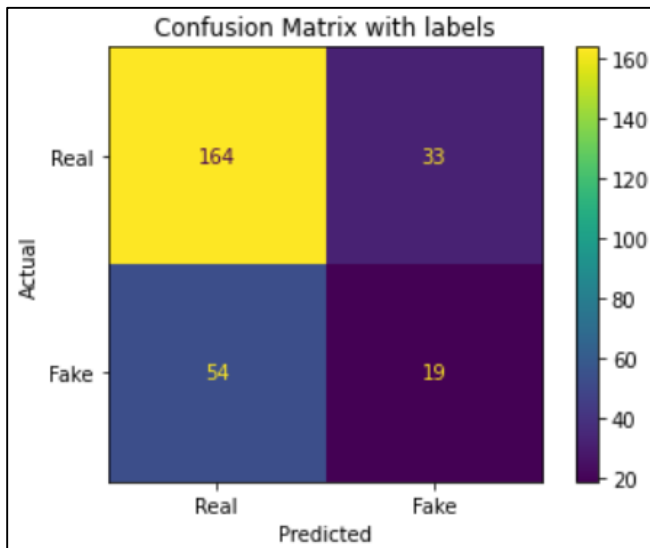


Figure 21 Confusion Matrix for EfficientNet B1, 3 fully connected layers

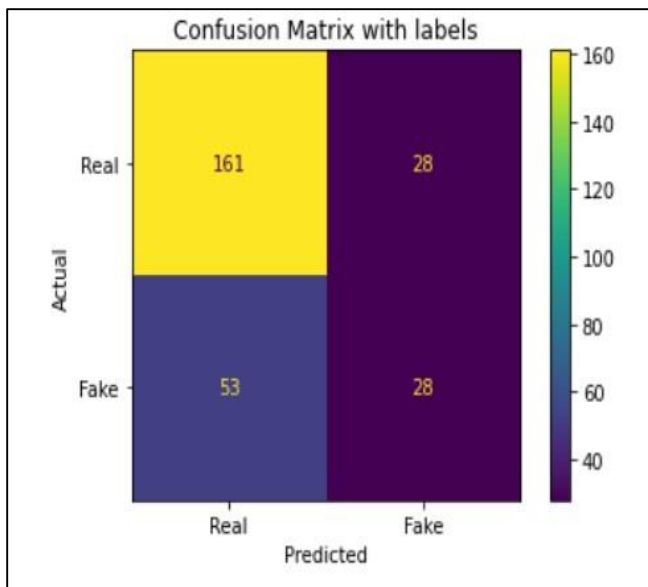


Figure 22 Confusion Matrix for EfficientNet B1, 2 fully connected layers

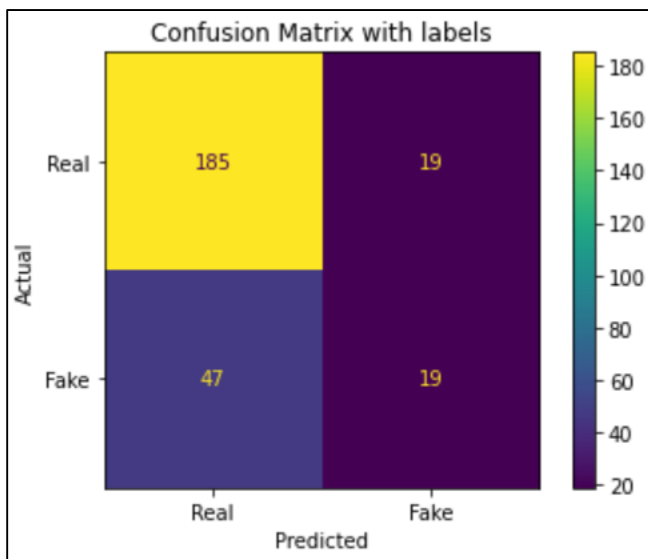


Figure 23 Confusion Matrix for EfficientNet B2, 2 fully connected layers

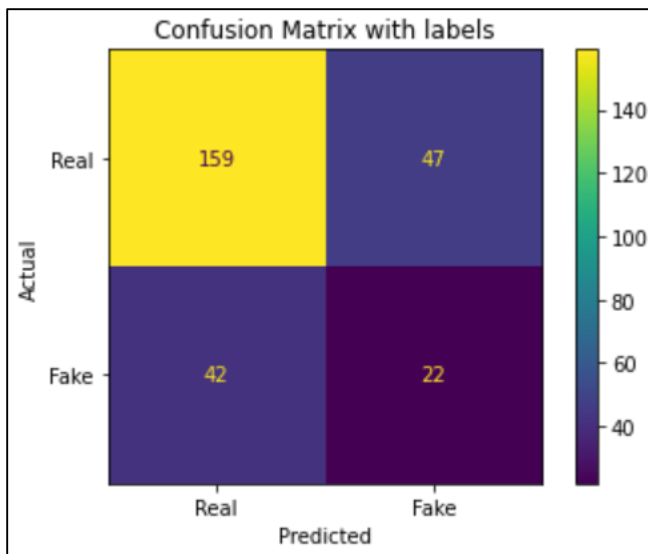


Figure 24 Confusion Matrix for EfficientNet B3, 3 fully connected layers

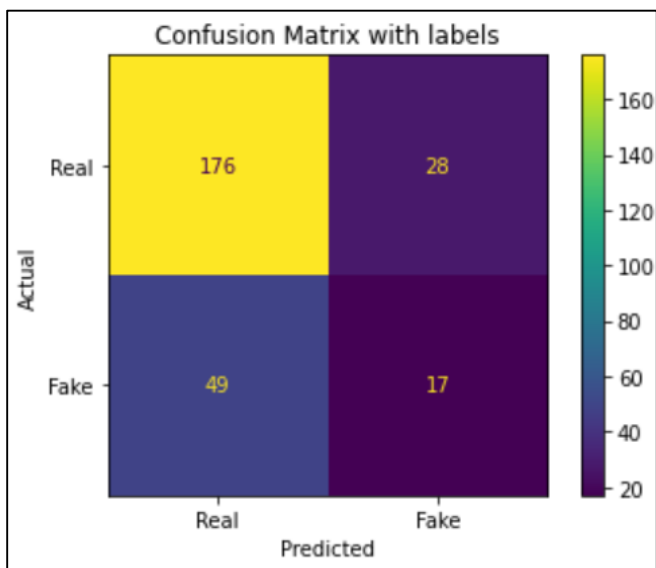


Figure 25 Confusion Matrix for EfficientNet B3, 2 fully connected layers

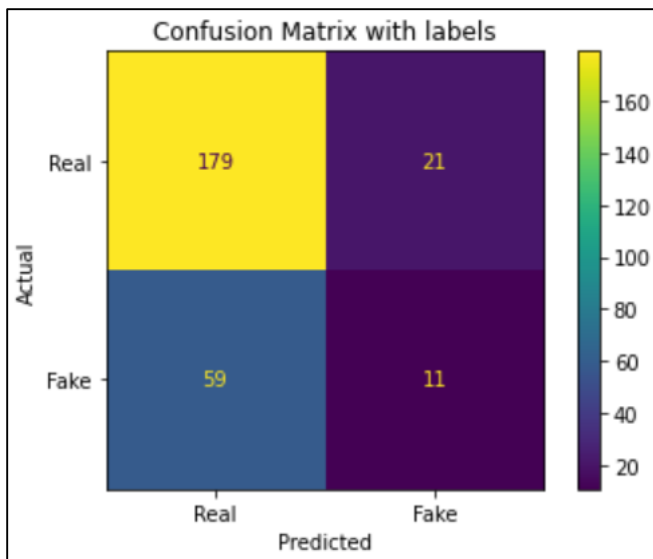


Figure 26 Confusion Matrix for EfficientNet B4 with 3 fully connected layers

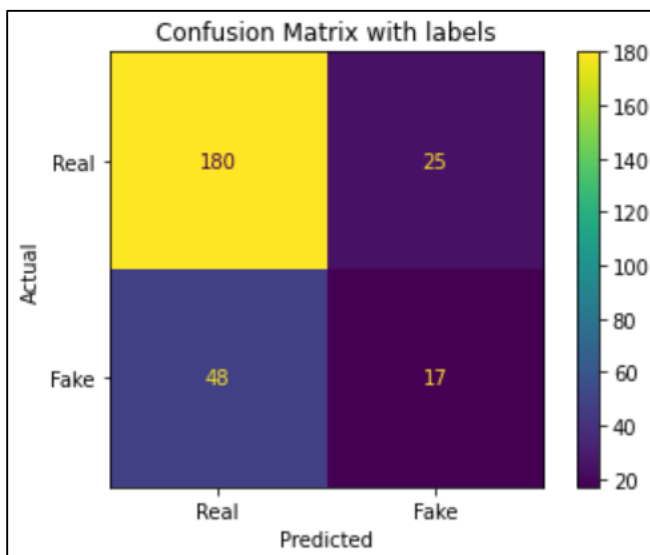


Figure 27 Confusion Matrix for EfficientNet B4 with 2 fully connected layers

8. CONCLUSION

With the growing advances in technology, creation and manipulation of digital data is easier than earlier times. Developing robust and accurate tools to detect such manipulated media is crucial to prevent them from negatively affecting the society. Deepfake detection tools are of great use to social media companies, as social media is the major platform where people interact and discover deepfakes. If each social media platform on the internet can integrate deepfake detection tools in their news feeds, it can go a long way in stemming the spread of such deepfakes. For instance, Microsoft unveiled a deepfake detection tool in 2020 to help content creators in identifying the authenticity of a video. [25]

In this project, we have successfully demonstrated the effectiveness of CNN and RNN models for deepfake detection. We conducted a detailed study on the performance of EfficientNet and LSTM models for the task of deepfake detection. Both models were trained and tested on a mixed dataset of 2731 videos. The mixed dataset comprised of videos from three different open source datasets created for deepfake detection. We also conclude that RNNs like LSTM are better suited for detecting deepfake videos accurately. Lastly, we have also deployed the trained model of LSTM with a sequence length of 50 as a web app.

There is scope for further improvement in the accuracy of our LSTM model. Currently, we have trained our model on a subset of three different datasets. The model can become more accurate and robust if trained on all the videos of these three datasets. Another promising line of research is to perform hyper parameter tuning for LSTM by increasing the sequence length to values greater than 100. Lastly, there is scope for trying out different EfficientNet models as a feature extractor for the LSTM model. A study can be conducted on how the performance of LSTMs changes when the number of features output by the feature extractor change.


APPENDIX A

Our research paper has been accepted at the **Thirteenth International Conference on Advances in Computing, Control, and Telecommunication Technologies - ACT 2022** which will be held during Jun 27-28, 2022.

The research paper will be published in the **GRENZE International Journal of Engineering and Technology (GIJET)**, and will be indexed in Scopus.

APPENDIX B

Plagiarism report:

 **Similarity Report ID:** oid:8054:17408895

● **5% Overall Similarity**

Top sources found in the following databases:

- 3% Internet database
- 1% Publications database
- Crossref database
- Crossref Posted Content database
- 4% Submitted Works database

APPENDIX C

The full source code of this project is available as a GitHub repository.

Link of the GitHub repository - [manasikasande/deepfake-detection: Final Year Project on Deepfake Detection \(github.com\)](https://github.com/manasikasande/deepfake-detection: Final Year Project on Deepfake Detection)

The trained models are available as a .pt file on google drive. Link for the same -

https://drive.google.com/drive/folders/117qInK9-VrLMhOBeROBA_jUADs_6H9QB?usp=sharing

One can download the trained model from the google drive folder and use it for inference.

REFERENCES

- [1] "Wombo," [Online]. Available: <https://www.wombo.ai/>. [Accessed 15 May 2022].
- [2] "Deepfakes Web," [Online]. Available: <https://deepfakesweb.com/>. [Accessed 15 May 2022].
- [3] "Face App," [Online]. Available: <https://www.faceapp.com/>. [Accessed 15 May 2022].
- [4] K. Fagan, "A video that appeared to show Obama calling Trump a "dipsh-t" is a warning about a disturbing new trend called 'deepfakes'," Business Insider, 18 April 2018. [Online]. Available: <https://www.businessinsider.in/tech/a-video-that-appeared-to-show-obama-calling-trump-a-dipsh-t-is-a-warning-about-a-disturbing-new-trend-called-deepfakes/articleshow/63807263.cms>. [Accessed 15 May 2022].
- [5] M. Grothaus, "Deepfake videos are the latest cruel form of school bullying – parents and teachers must watch out," iNews UK, 9 November 2021. [Online]. Available: <https://inews.co.uk/news/technology/deepfake-videos-school-bullying-cyberbullying-ai-apps-parents-teachers-children-1290664>. [Accessed 15 May 2022].
- [6] B. S. Samuel, "A guy made a deepfake app to turn photos of women into nudes. It didn't go well.," Vox, 27 Jun 2019. [Online]. Available: <https://www.vox.com/2019/6/27/18761639/ai-deepfake-deepnude-app-nude-women-porn>. [Accessed 10 May 2022].
- [7] Y. Pashaeva, "Scammers Are Using Deepfake Videos Now," Slate, 13 September 2021. [Online]. Available: <https://slate.com/technology/2021/09/deepfake-video-scams.html>. [Accessed 15 May 2022].
- [8] "Digital war: How Russia is using deep fakes in Ukraine for propaganda," Business Today, 02 March 2022. [Online]. Available: <https://www.businesstoday.in/latest/world/story/digital-war-how-russia-is-using-deep-fakes-in-ukraine-for-propaganda-324531-2022-03-02>. [Accessed 15 May 2022].

- [9] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini and S. Tubaro, "Video Face Manipulation Detection Through Ensemble of CNNs," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021.
- [10] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," in *IEEE/CVF International Conference on Computer Vision*, 2019.
- [11] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang and C. C. Ferrer, "The DeepFake Detection Challenge (DFDC) Dataset," in *arXiv 10.48550/ARXIV.2006.07397*, 2020.
- [12] A. Karandikar, V. Deshpande, S. Singh, S. Nagbhikar and S. Agarwal, "Deepfake Video Detection Using Convolutional Neural Network," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, 2020.
- [13] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *arXiv 10.48550/ARXIV.1409.1556*, 2014.
- [14] Y. Li, X. Yang, P. Sun, H. Qi and S. Lyu, "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
- [15] A. A. Pokroy and A. D. Egorov, "EfficientNets for DeepFake Detection: Comparison of Pretrained Models," in *IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering*, 2021 .
- [16] D. Coccomini, N. Messina, C. Gennaro and F. Falchi, "Combining EfficientNet and Vision Transformers for Video Deepfake Detection," in *arXiv 10.48550/ARXIV.2107.02612*, 2021.
- [17] D. Güera and E. J. Delp, "Deepfake Video Detection Using Recurrent Neural Networks," in *15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2018.
- [18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *arXiv 10.48550/ARXIV.1512.00567*, 2015.

- [19] A. Jadhav, A. Patange, J. Patel, H. Patil and M. Mahajan, "Deepfake Video Detection Using Neural Networks," *International Journal for Scientific Research and Development*, vol. 8, no. 1, 2020.
- [20] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi and P. Natarajan, "Recurrent Convolutional Strategies for Face Manipulation Detection in Videos," in *arXiv 10.48550/ARXIV.1905.00582*, 2019.
- [21] "Recurrent Convolutional Structures for audio spoof and video spoof detection," *IEEE Journal of Selected Topics in Signal Processing*, 2020.
- [22] X. a. Y. J. a. T. M. a. D. H. a. N. A. a. E. N. a. S. M. a. V. V. a. K. T. a. L. K. A. a. J. L. a. A. P. a. P. Y.-H. a. H. Wang, "ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech," in *arXiv 10.48550/ARXIV.1911.01601*, 2019.
- [23] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *International Conference on Machine Learning*, 2019.
- [24] "LSTM," [Online]. Available: https://en.wikipedia.org/wiki/Long_short-term_memory.
- [25] L. Kelion, "Deepfake detection tool unveiled by Microsoft," BBC, 1 September 2020. [Online]. Available: <https://www.bbc.com/news/technology-53984114>. [Accessed 20 05 2022].
- [26] Crescendo, "YouTube," 16 April 2020. [Online]. Available: <https://www.youtube.com/watch?v=xkqfIKC64IM>. [Accessed 02 April 2022].