**A PROJECT REPORT ON**

**Deepfake Video Detection System Using Deep Learning**

**SUBMITTED TO**

**SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE**



**IN THE PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE OF BACHELOR OF ENGINEERING IN INFORMATIONTECHNOLOGY**

**SUBMITTED BY,**

| **1. Navnit Amrutharaj** | **2. Aditya Gite** |
| --- | --- |
| **3. Rushikesh Jamnekar** | **4. Tushar Kamble** |

**UNDER THE GUIDANCE OF**

**Prof. Sourabh Natu**



**DEPARTMENT OF INFORMATION TECHNOLOGY**

**INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY, PUNE**

**ACADEMIC YEAR: 2024 – 2025**

1



**CERTIFICATE**

*This is to certify that the project report entitled* **Deepfake Video Detection System Using Deep Learning** *SUBMITTED BY,*

1. **Navnit Amrutharaj B400950296**
2. **Aditya Gite B400950255**
3. **Rushikesh Jamnekar B400950266**
4. **Tushar Kamble B400950272**

is a bonafide work carried out by them under the supervision of

**Prof. Sourabh Natu**

and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the Degree of Bachelor of

Engineering (Information Technology).

The project work has not been earlier submitted to any other institute oruniversity for the award of degree or diploma.

**Prof. Dr. Jyoti Surve Dr. Vaishali Patil Internal Guide Head of the Department Principal**

**Prof.**

**External Examiner**

**Place: Date:**

**Acknowledgement**

With immense pleasure, we present this project report on Deepfake Detection using Deep Learning as part of the curriculum for the B.E. in Information Technology at the International Institute of Information Technology, Hinjewadi, Pune. We are privileged to have completed this report under the invaluable mentorship of Dr. Jyoti Surve, Head of the Department of Information Technology. Her guidance, cooperation, and encouragement have been instrumental in steering this project toward success. We extend our heartfelt gratitude to Dr. Vaishali Patil, Principal of the Information Technology Department, for providing the necessary facilities and support to ensure the smooth progress of this report. Additionally, we express our sincere thanks to all the faculty members of the Information Technology Department, the management, our friends, and our family members, whose guidance and unwavering support, both direct and indirect, have been vital from the initial conception of this idea to the preparation of this report.

**Name of students**

**Navnit Amrutharaj Aditya Gite**

**Rushikesh Jamnekar**

**Tushar Kamble**

**ABSTRACT**

The growing computation power has made the deep learning algorithms so powerful that creating an indistinguishable human synthesized video popularly called as deep fakes has become very simple. Scenarios where these realistic face swapped deep fakes are used to create political distress, fake terrorism events and blackmail people are easily envisioned. In this work, we describe a new deep learning-based method that can effectively distinguish AI-generated fake videos from real videos. Our method is capable of automatically detecting the replacement and reenactment deep fakes. We are trying to use Artificial Intelligence(AI) to fight Artificial Intelligence(AI). Our system uses a ResNext Convolution neural network to extract the frame-level features and these features and further used to train the Long Short Term Memory(LSTM) based Recurrent Neural Network(RNN) to classify whether the video is subject to any kind of manipulation or not, i.e whether the video is deepfake or real video. To emulate the real time scenarios and make the model perform better on real time data, we evaluate our method on large amount of balanced and mixed dataset prepared by mixing the various available data-set like Face-Forensics++[1], Deepfake detection challenge[2], and Celeb-DF[3]. We also show how our system can achieve competitive results using a very simple and robust approach.

**Contents**

1. **Synopsis 1**
2. **Technical Keywords 2**
   1. [Area of Project . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2](#_nf1bvd4aqxdr)
   2. [Technical Keywords . . . . . . . . . . . . . . . . . . . . . . . . . . 2](#_4ik8mx4juje6)
3. **Introduction 3**
   1. [Project Idea . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3](#_3rdttvqrkmen)
   2. [Motivation of the Project . . . . . . . . . . . . . . . . . . . . . . . 4](#_6jeqsp51gv0a)
   3. [Literature Survey . . . . . . . . . . . . . . . . . . . . . . . . . . . 5](#_xx79bbwhlw09)
4. **Problem Definition and scope 7**
   1. [Problem Statement . . . . . . . . . . . . . . . . . . . . . . . . . . 7](#_lgq4czcgciew)
      1. [Goals and objectives . . . . . . . . . . . . . . . . . . . . . 8](#_pbkv19wlmkxd)
      2. [Statement of scope . . . . . . . . . . . . . . . . . . . . . . 8](#_xcgva3y2lha5)
   2. [Major Constraints . . . . . . . . . . . . . . . . . . . . . . . . . . . 9](#_9lc456qo0106)
   3. [Methodologies of Problem solving . . . . . . . . . . . . . . . . . . 9](#_sy9ky9xs2wqo)
      1. [Analysis 10](#_y8sjzul09aeu)
      2. [Design 10](#_pyuqmzc0lhn)
      3. [Development 10](#_l0bwsrlsqshe)
      4. [Evaluation 10](#_lbt92sqpcku3)
   4. [Outcome 10](#_dvy7gdwod02y)
   5. [Applications 10](#_7ag7syw33unp)
   6. [Hardware Resources Required 10](#_3ipmumyauy0a)
   7. [Software Resources Required 11](#_x4r7g624hr5e)
5. **Software requirement specification 12**
   1. [Introduction 12](#_ee83ngwkeckg)
      1. [Purpose and Scope of Document 12](#_m0c9epp54vje)
      2. [Use Case View 12](#_io8ccukvw4xe)
   2. [Functional Model and Description 13](#_ucq9vt14e2pw)
      1. [Data Flow Diagram 13](#_5jhswqri7ci1)
      2. [Activity Diagram 15](#_zh8m42n0gdsk)
      3. [Non Functional Requirements 16](#_r62nm46s7jd)
      4. [Sequence Diagram 17](#_53u8y6brmk31)
6. **Detailed Design Document 18**
   1. [Introduction 18](#_1ozt48xvasgh)
      1. [System Architecture 18](#_8sylbde3cfpm)
   2. [Architectural Design 19](#_d2js91bk5kh5)
      1. [Module 1 : Data-set Gathering 20](#_nkt17hn1clr3)
      2. [Module 2 : Pre-processing 20](#_6rk6jrm2o8fl)
      3. [Module 3: Data-set split 21](#_vclg3vpq0zlg)
      4. [Module 4: Model Architecture 22](#_tcr8ohyf25e9)
      5. [Module 5: Hyper-parameter tuning 23](#_lidsu89e6n3n)
7. **Project Implementation 26**
   1. [Introduction 26](#_ap1ca5tqyrhk)
   2. [Tools and Technologies Used 27](#_6m9p5k20fuvi)
      1. [Planning 27](#_rurltjptfot8)
      2. [UML Tools 27](#_2qewya9bd03i)
      3. [Programming Languages 27](#_ta6z3p2sx7hf)
      4. [Programming Frameworks 27](#_5oa5qablbcd4)
      5. [IDE 27](#_kd49scke2l33)
      6. [Versioning Control 27](#_qkhcjs4q8t1k)
      7. [Cloud Services 27](#_ahbxpeu5j7gn)
      8. [Application and web servers 28](#_lc4fi1qeix08)
      9. [Libraries 28](#_pis7khe13ny3)
   3. [Algorithm Details 28](#_i3rwbrhvupty)
      1. [Dataset Details 28](#_7ebekv7mvlpi)
      2. [Preprocessing Details 28](#_vpf9zpavqg2g)
      3. [Model Details 29](#_au3f9xk553ke)
      4. [Model Training Details 32](#_cgbv2huqkn4i)
      5. [Model Prediction Details 34](#_wotus54ei39r)
8. **Software Testing 35**
   1. [Type of Testing Used 35](#_bwjd44pydyqq)
   2. [Test Cases and Test Results 36](#_ck7p34yze4z3)
9. **Results and Discussion 37**
   1. [Screen shots 37](#_wd6u0mobxzhv)
   2. [Outputs 41](#_ccu4j9j97kk3)
      1. [Model results 41](#_3ha5c0cu4uik)
10. **Conclusion and Future Scope 43**
    1. [Conclusion 43](#_l8vtaustivaa)
    2. [Future Scope 43](#_ezl4h2xs9v8p)
11. **References 44**
12. **Project Planner 45**
13. Timeline **47**

**List of Figures**

* 1. **Use case diagram** 12
  2. **DFD Level 0** 13
  3. **DFD Level 1** 14
  4. **DFD Level 2** 14
  5. **Training Workflow** 15
  6. **Testing Workflow** 15
  7. **Sequence Diagram** 17
  8. **System Architecture** 18
  9. **Deepfake generation** 19
  10. **Face Swapped deepfake generation** 19
  11. **Dataset** 21
  12. **Pre-processing of video** 22
  13. **Train test split** 23

**6.6 Overview of our model** 24

* 1. **ResNext Architecture** 29
  2. **ResNext Working** 30
  3. **Overview of ResNext Architecture** 30
  4. **Overview of LSTM Architecture** 31
  5. **Internal LSTM Architecture** 31
  6. **Relu Activation function** 31
  7. **Dropout layer overview** 32
  8. **Softmax Layer** 33
  9. **Home Page** 37
  10. **Uploading Real Video** 38
  11. **Real Video Output** 38
  12. **Uploading Fake Video** 39

| 8.5 | **Fake video Output** . . . . . . . . . . . . . . . . . . . . . . . . . . | 39 |
| --- | --- | --- |
| 8.6 | **Uploading Video with no faces** . . . . . . . . . . . . . . . . . . . | 40 |
| 8.7 | **Output of Uploaded video with no faces** . . . . . . . . . . . . . . | 40 |
| 8.8 | **Uploading file greater than 100MB** . . . . . . . . . . . . . . . . . | 41 |
| 8.9 | **Pressing Upload button without selecting video** . . . . . . . . . . | 41 |
| B.1 | **Complete Project Plan** . . . . . . . . . . . . . . . . . . . . . . . . | 45 |
| B.2 | **Project Plan 1** . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 46 |
| B.3 | **Project Plan 2** . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 47 |

* 1. Hardware Requirements 11
  2. Test Case Report 36
  3. Trained Model Results 4

**Chapter 1 Introduction**

# Background

In the world of ever growing Social media platforms, Deepfakes are considered as the major threat of AI. There are many Scenarios where these realistic face swapped deepfakes are used to create political distress, fake terrorism events, blackmail people are easily envisioned. Some of the examples are Brad Pitt, Angelina Jolie deepfake videos.

It becomes very important to spot the difference between the deepfake and pristine video. We are using AI to fight AI. Deepfakes are created using tools like FaceApp[11] and Face Swap [12], which use pre-trained neural networks like GAN or Auto encoders for these deepfakes creation. Our method uses a LSTM based artificial neural network to process the sequential temporal analysis of the video frames and pre-trained ResNext CNN to extract the frame level features. ResNext Convolution neural network extracts the frame-level features and these features are further used to train the Long Short Term Memory based artificial Recurrent Neural Network to classify the video as Deepfake or real. To emulate the real time scenarios and make the model perform better on real time data, we trained our method with a large amount of balanced and combination of various available dataset like FaceForensic++[1], Deepfake detection challenge[2], and Celeb-DF[3].

Further to make it ready to use for the customers, we have developed a front end application where the user will upload the video. The video will be processed by the model and the output will be rendered back to the user with the classification of the video as deepfake or real and confidence of the model.

# Relevance

The increasing sophistication of mobile camera technology and the ever- growing reach of social media and media sharing portals have made the creation and propagation of digital videos more convenient than ever before. Deep learning has given rise to technologies that would have been thought impossible only a handful of years ago. Modern generative models are one example of these, capable of synthesizing hyper realistic images, speech, music, and even video. These models have found use in a wide variety of applications, including making the world more accessible through text-to-speech, and helping generate training data for medical imaging.

Like any transformative technology, this has created new challenges. So called "deep fakes" produced by deep generative models that can manipulate video and audio clips. Since their first appearance in late 2017, many open-source deepfake generation methods and tools have emerged now, leading to a growing number of synthesized media clips. While many are likely intended to be humorous, others could be harmful to individuals and society. Until recently, the number of fake videos and their degrees of realism has been increasing due to the availability of editing tools, the high demand for domain expertise.

Spreading of the deepfakes over the social media platforms have become very common leading to spamming and speculating wrong information over the platform. Just imagine a deepfake of our prime minister declaring war against neighboring countries, or a deepfake of reputed celebrity abusing the fans. These types of deepfakes will be terrible, and lead to threatening, misleading common people.

To overcome such a situation, deepfake detection is very important. So, we describe a new deep learning-based method that can effectively distinguish AI-generated fake videos (deepfake videos) from real videos. It’s incredibly important to develop technology that can spot fakes, so that the deepfakes can be identified and prevented from spreading over the internet.

# Literature Survey

Face Detection and Generation using DeepfakeDG [2023] proposed a model based on CNNs and the VGG network to improve the accuracy of detecting deepfakes as well as generating fake images. Their work focused on enhancing detection capabilities but had some notable limitations such as the use of limited and less diverse datasets, challenges in adapting to real-time scenarios, and a vulnerability to adversarial attacks that could reduce the model’s robustness. Moreover, their approach mainly concentrated on static image features, leaving room for improvement in temporal consistency analysis across frames.

Deep Fake Detection using Deep Learning Methods: A Systematic and Comprehensive Review [2022] provided a broad survey of different algorithms like CNNs, RNNs, GANs, and LSTMs, discussing their strengths and weaknesses in detecting deepfakes. It highlighted that ensemble-based models and attention mechanisms tend to perform better across various benchmarks. However, the work was largely theoretical with limited practical validation through experiments, making its real-world applicability somewhat restricted and indicating the need for larger scale empirical studies.

Explainable Deepfake Video Detection using CNN and CapsuleNet [2024] developed a detection framework that prioritized both accuracy and model explainability. By combining CNNs and Capsule Networks, the model achieved promising results in detecting manipulated videos while ensuring interpretability; however, the system required extensive training data and showed potential weaknesses when tested on unseen or real-time data. This pointed towards a trade-off between explainability and generalization ability.

Deepfake Detection: Analyzing Model Generation Across Architectures, Datasets, and Pre-Training Paradigms explored detection models based on CNNs and GANs, demonstrating strong results particularly on the DFDC dataset. However, it was observed that the models may overfit specific datasets and struggled to generalize effectively on degraded or unseen samples, suggesting a need for future work on improving cross-dataset performance and robustness against different manipulation techniques

**Table 1: Literature Survey**

| Year and Title | Algorithm | Result | Limitation/Drawbacks |
| --- | --- | --- | --- |
| DeepfakeDG: A Deep Learning Approach for Deep Fake Detection and Generation, 2023 | 1.CNNs (Convolutional Neural Networks)  2. VGG (Visual Geometry Group) Model | The proposed model demonstrated improved detection accuracy and generative capabilities for deepfakes. | The paper's limitations include limited dataset diversity, real-time detection challenges, and vulnerability to adversarial attacks. |
| Deep Fake Detection using Deep Learning Methods: A Systematic and Comprehensive Review, 2022 | 1.CNNs (Convolutional Neural Networks)  2.RNNs (Recurrent Neural Networks)  3.GANs(Generative Adversarial Networks).  4.LSTM (Long Short-Term Memory) | The paper reviews numerous algorithms, highlighting their effectiveness in detecting deepfakes. It suggests that ensemble methods and attention-based architectures yield better performance compared to others. | Limited in providing empirical results for specific algorithms; focuses more on theoretical insights and comparative analysis. |
| Explainable Deepfake Video Detection using Convolutional Neural Network and CapsuleNet, 2024 | 1.CNNs (Convolutional Neural Networks)  2.CapsuleNet | The proposed approach achieved high accuracy in detecting deepfake videos while maintaining explainability. | The method may struggle with unseen data and requires extensive training data. |
| Deepfake Detection: Analysing Model Generation Across Architectures, Datasets and Pre-Training Paradigms | 1.CNNs (Convolutional Neural Networks)  2.GANs(Generative Adversarial Networks). | The proposed approach achieved high accuracy in detecting deepfake videos on the DFDC dataset while maintaining robustness against adversarial attacks. | The detection methods struggle to generalize to unseen or degraded samples, leading to potential overfitting on specific datasets and manipulation patterns. |



**Chapter 4**

**Problem Definition and scope**

# Problem Statement

Convincing manipulations of digital images and videos have been demonstrated for several decades through the use of visual effects, recent advances in deep learn- ing have led to a dramatic increase in the realism of fake content and the accessibility in which it can be created. These so-called AI-synthesized media (popularly referred to as deep fakes).Creating the Deep Fakes using the Artificially intelligent tools are simple task. But, when it comes to detection of these Deep Fakes, it is major chal- lenge. Already in the history there are many examples where the deepfakes are used as powerful way to create political tension[14], fake terrorism events, revenge porn, blackmail peoples etc.So it becomes very important to detect these deepfake and avoid the percolation of deepfake through social media platforms. We have taken a step forward in detecting the deep fakes using LSTM based artificial Neural network.

## Goals and objectives

Goal and Objectives:

* + - * Our project aims at discovering the distorted truth of the deep fakes.
      * Our project will reduce the Abuses’ and misleading of the common people on the world wide web.
      * Our project will distinguish and classify the video as deepfake or pristine.
      * Provide an easy to use system to upload the video and distinguish whether the video is real or fake.





## Project scope

There are many tools available for creating the deep fakes, but for deep fake detection there is hardly any tool available. Our approach for detecting the deep fakes will be a great contribution in avoiding the percolation of the deep fakes over the world wide web. We will be providing a web-based platform for the user to upload the video and classify it as fake or real. This project can be scaled up from developing a web-based platform to a browser plugin for au- tomatic deep fake detection. Even big applications like WhatsApp, Facebook can integrate this project with their application for easy pre-detection of deep fakes before sending them to another user. A description of the software with Size of input, bounds on input, input validation, input dependency, i/o state diagram, Major inputs, and outputs are described without regard to implementation de- tail.

# Major Constraints

* **User:** Users of the application will be able detect whether the uploaded video is fake or real, Along with the model confidence of the prediction.
* **Prediction:** The user will be able to see the playing video with the output on the face along with the confidence of the model.
* **Easy and User-friendly User-Interface:** Users seem to prefer a more simplified process of Deep Fake video detection. Hence, a straightforward and user-friendly interface is implemented.The UI contains a browse tab to select the video for processing. It reduces the complications and at the same time enriches the user experience.
* **Cross-platform compatibility:** with an ever-increasing target market, acces- sibility should be your main priority. By enabling a cross-platform compatibility feature, you can increase your reach across different platforms. Being a server side application it will run on any device that has a web browser installed in it.





# Methodologies of Problem solving

## Analysis

* + - * Solution Requirement

We analysed the problem statement and found the feasibility of the solution of the problem. We read different research paper as mentioned in 3.3. After checking the feasibility of the problem statement. The next step is the data- set gathering and analysis. We analysed the data set in different approaches to training like negatively or positively trained i.e training the model with only fake or real video’s but found that it may lead to addition of extra bias in the model leading to inaccurate predictions. So after doing a lot of research we found that the balanced training of the algorithm is the best way to avoid the bias and variance in the algorithm and get a good accuracy.

* + - * Solution Constraints

We analysed the solution in terms of cost,speed of processing,requirements,level of expertise, availability of equipments.

* + - * Parameter Identified
        1. Blinking of eyes
        2. Teeth enchantment
        3. Bigger distance for eyes
        4. Moustaches
        5. Double edges, eyes, ears, nose
        6. Iris segmentation
        7. Wrinkles on face
        8. Inconsistent head pose
        9. Face angle
        10. Skin tone
        11. Facial Expressions
        12. Lighting
        13. Different Pose
        14. Double chins
        15. Hairstyle
        16. Higher cheekbones

## Design

After research and analysis we developed the system architecture of the solution as mentioned in Chapter 6. We decided the baseline architecture of the Model which includes the different layers and their numbers.

## Development

After analysis we decided to use the PyTorch framework along with python3 lan- guage for programming. PyTorch is chosen as it has good support for CUDA i.e Graphic Processing Unit (GPU) and it is customize-able. Google Cloud Platform for training the final model on a large number of data-set.

## Evaluation

We evaluated our model with a large number of real time dataset which include YouTube videos dataset. Confusion Matrix approach is used to evaluate the accuracy of the trained model.

# Outcome

The outcome of the solution is trained deepfake detection models that will help the users to check if the new video is deepfake or real.

# Applications

Web based applications will be used by the user to upload the video and submit the video for processing. The model will pre-process the video and predict whether the uploaded video is a deepfake or real video.

# Hardware Resources Required

In this project, a computer with sufficient processing power is needed. This project requires too much processing power, due to the image and video batch processing.

* **Client-side Requirements:** Browser: Any Compatible browser device





**Table 4.1:** Hardware Requirements

| Sr. No. | Parameter | Minimum Requirement |
| --- | --- | --- |
| 1 | Intel Xeon E5 2637 | 3.5 GHz |
| 2 | RAM | 16 GB |
| 3 | Hard Disk | 100 GB |
| 4 | Graphic card | NVIDIA GeForce GTX Titan (12 GB RAM) |

# Software Resources Required

Platform :

1. Operating System: Windows 7+
2. Programming Language : Python 3.0
3. Framework: PyTorch 1.4 , Django 3.0
4. Cloud platform: Google Cloud Platform
5. Libraries : OpenCV, Face-recognition





**Chapter 5**

**Design**

# Introduction

## Purpose and Scope of Document

This document lays out a project plan for the development of Deepfake video de- tection using neural network.The intended readers of this document are current and future developers working on Deepfake video detection using neural network and the sponsors of the project. The plan will include, but is not restricted to, a sum- mary of the system functionality, the scope of the project from the perspective of the “Deepfake video detection” team (me and my mentors), use case diagram, Data flow diagram,activity diagram, functional and non- functional requirements, project risks and how those risks will be mitigated, the process by which we will develop the project, and metrics and measurements that will be recorded throughout the project.

## Use Case View



**Figure 5.1: Use case diagram**



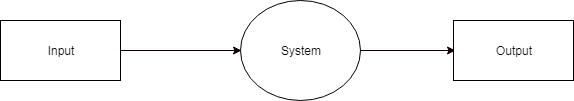


# Functional Model and Description

A description of each major software function, along with data flow (structured anal- ysis) or class hierarchy (Analysis Class diagram with class description for object oriented system) is presented.

## Data Flow Diagram

**DFD Level-0**



**Figure 5.2: DFD Level 0**

DFD level – 0 indicates the basic flow of data in the system. In this System Input is given equal importance as that for Output.

* + - 1. Input: Here input to the system is uploading video.
      2. System: In system it shows all the details of the Video.
      3. Output: Output of this system is it shows the fake video or not.

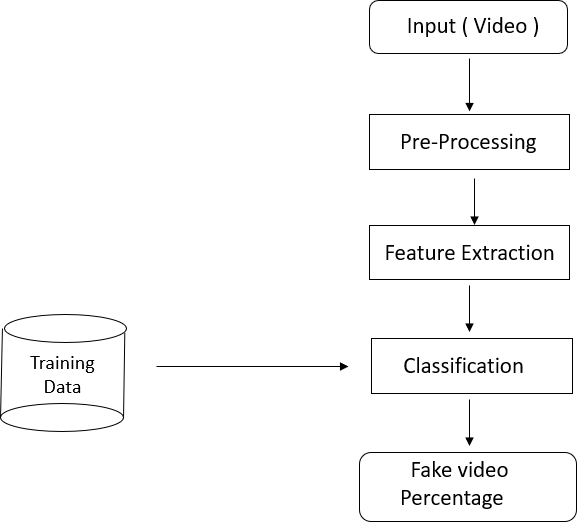
Hence, the data flow diagram indicates the visualization of system with its input and output flow.

**DFD Level-1**

1. DFD Level – 1 gives more in and out information of the system.
2. Where system gives detailed information of the procedure taking place.



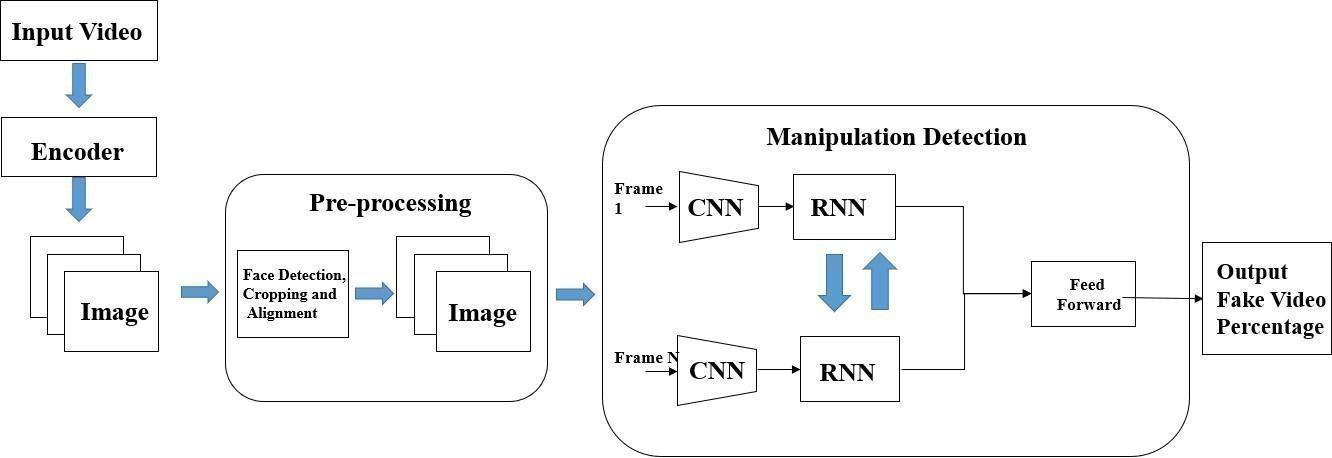




**Figure 5.3: DFD Level 1**

**DFD Level-2**

1. DFD level-2 enhances the functionality used by user etc.



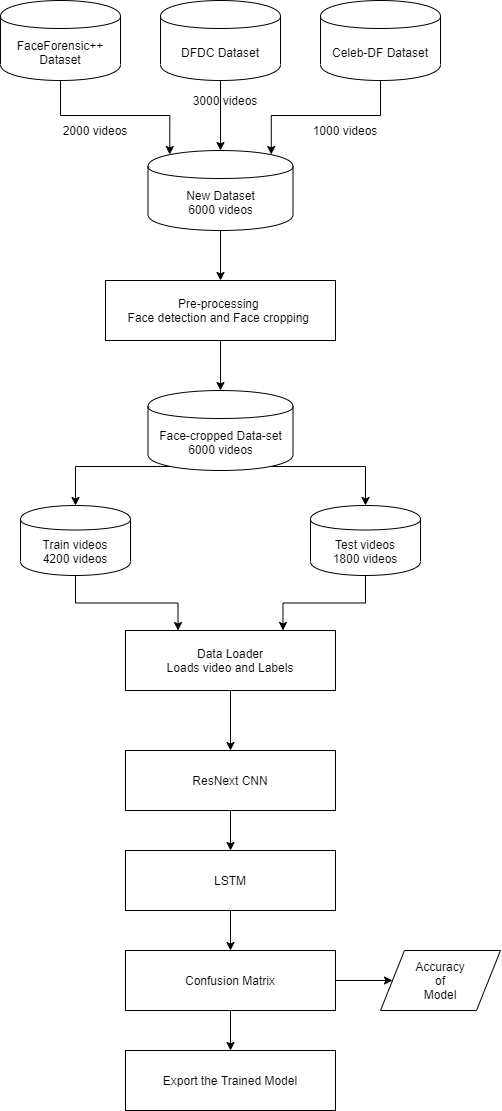
**Figure 5.4: DFD Level 2**





## Activity Diagram:

**Training Workflow:**

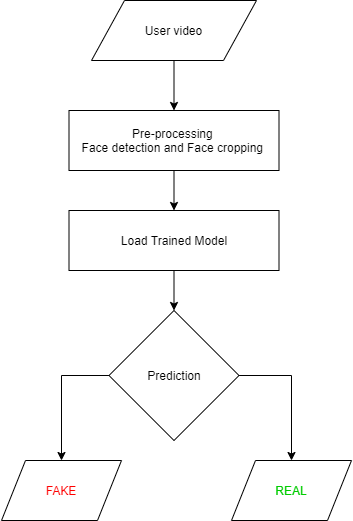


**Figure 5.5: Training Workflow**





**Testing Workflow:**



**Figure 5.6: Testing Workflow**

## Non Functional Requirements:

**Performance Requirement**

* + - 1. The software should be efficiently designed so as to give reliable recognition of fake videos and so that it can be used for more pragmatic purpose.
      2. The design is versatile and user friendly.
      3. The application is fast, reliable and time saving.
      4. The system have universal adaptations.
      5. The system is compatible with future upgradation and easy integration.





**Safety Requirement**

* + - 1. The Data integrity is preserved. Once the video is uploaded to the system. It is only processed by the algorithm. The videos are kept secured from the human interventions, as the uploaded video is not are not able for human manipulation.
      2. To extent the safety of the videos uploaded by the user will be deleted after 30 min from the server.

**Security Requirement**

* + - 1. While uploading the video, the video will be encrypted using a certain symmet- ric encryption algorithm. On server also the video is in encrypted format only. The video is only decrypted from preprocessing till we get the output. After getting the output the video is again encrypted.
      2. This cryptography will help in maintain the security and integrity of the video.
      3. SSL certification is made mandatory for Data security.

## Sequence Diagram



**Figure 5.7: Sequence Diagram**



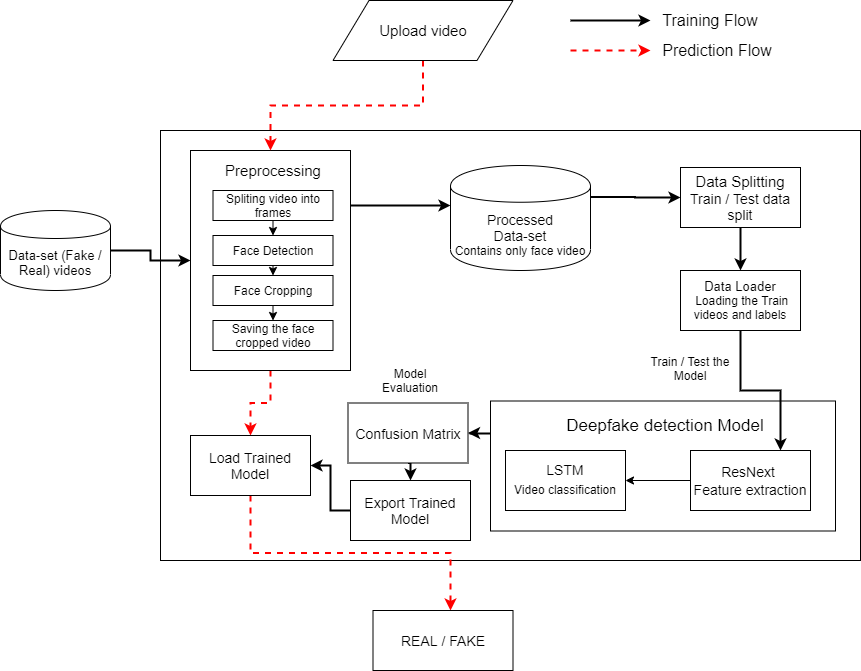


**Chapter 6**

**Detailed Design Document**

# Introduction

## System Architecture



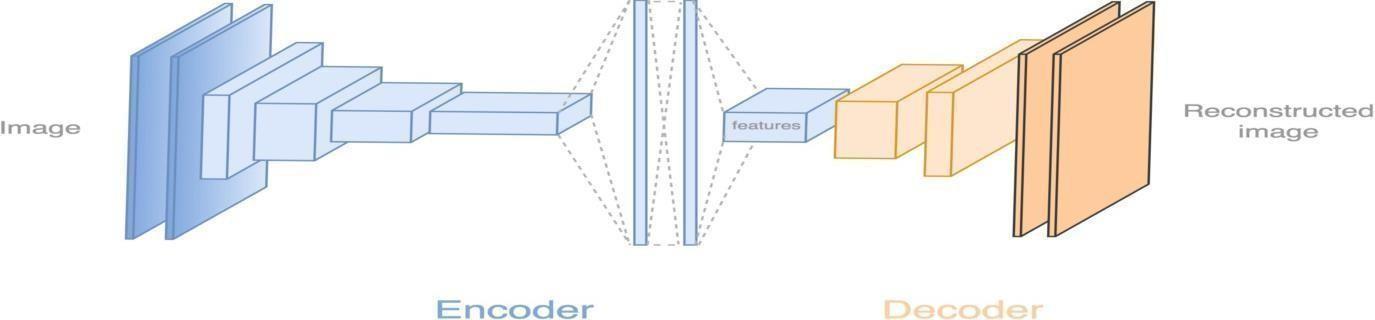




taken a dataset, preprocessed the dataset and created a new processed dataset which only includes the face cropped videos.

* + - 1. **Creating deepfake videos**

To detect the deepfake videos it is very important to understand the creation process of the deepfake. Majority of the tools including the GAN and autoencoders takes a source image and target video as input. These tools split the video into frames , detect the face in the video and replace the source face with target face on each frame. Then the replaced frames are then combined using different pre-trained models. These models also enhance the quality of video my removing the left-over traces by the deepfake creation model. Which result in creation of a deepfake looks realistic in nature. We have also used the same approach to detect the deepfakes. Deepfakes created using the pretrained neural networks models are very realistic that it is almost impossible to spot the difference by the naked eyes. But in reality, the deepfakes creation tools leaves some of the traces or artifacts in the video which may not be noticeable by the naked eyes. The motive of this paper to identify these unnoticeable traces and distinguishable artifacts of these videos and classified it as deepfake or real video.



**Figure 6.2: Deepfake generation**



**Figure 6.3: Face Swapped deepfake generation**





**Tools for deep fake creation.**

1. Faceswap
2. Faceit
3. Deep Face Lab
4. Deepfake Capsule GAN
5. Large resolution face masked

# Architectural Design

## Module 1 : Data-set Gathering

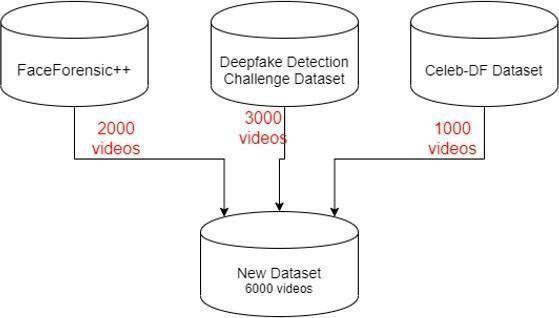
For making the model efficient for real time prediction. We have gathered the data from different available data-sets like FaceForensic++(FF)[1], Deepfake detection challenge(DFDC)[2], and Celeb-DF[3]. Futher we have mixed the dataset the col- lected datasets and created our own new dataset, to accurate and real time detection on different kind of videos. To avoid the training bias of the model we have consid- ered 50% Real and 50% fake videos.

Deep fake detection challenge (DFDC) dataset [3] consist of certain audio alerted video, as audio deepfake are out of scope for this paper. We preprocessed the DFDC dataset and removed the audio altered videos from the dataset by running a python script.

After preprocessing of the DFDC dataset, we have taken 1500 Real and 1500 Fake videos from the DFDC dataset. 1000 Real and 1000 Fake videos from the FaceForensic++(FF)[1] dataset and 500 Real and 500 Fake videos from the Celeb- DF[3] dataset. Which makes our total dataset consisting 3000 Real, 3000 fake videos and 6000 videos in total. Figure 2 depicts the distribution of the data-sets.







**Figure 6.4: Dataset**

## Module 2 : Pre-processing

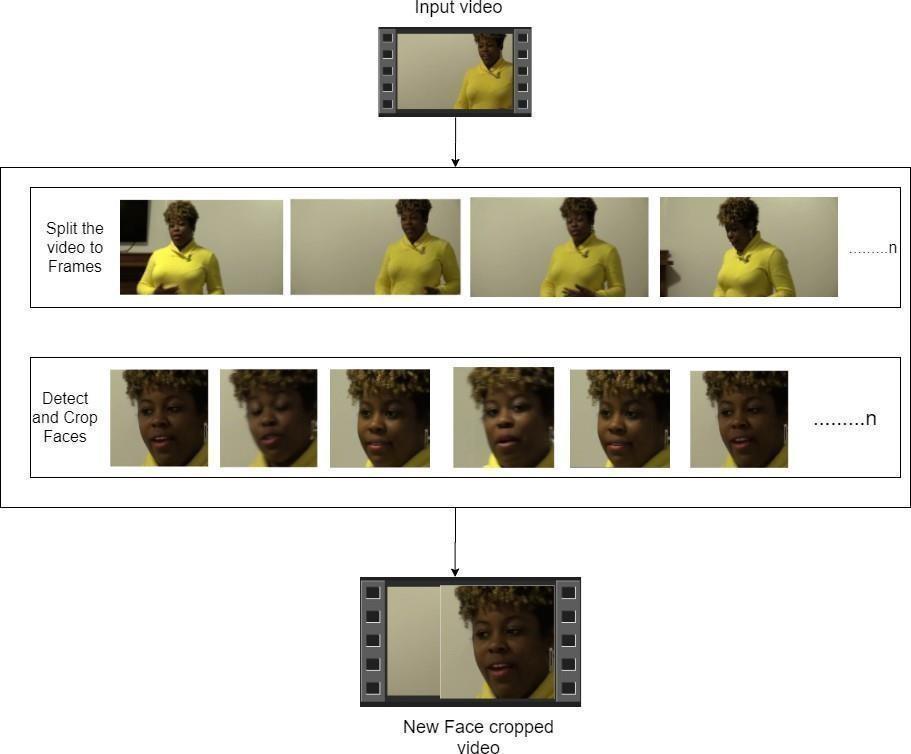
In this step, the videos are preprocessed and all the unrequired and noise is removed from videos. Only the required portion of the video i.e face is detected and cropped. The first steps in the preprocessing of the video is to split the video into frames.

After splitting the video into frames the face is detected in each of the frame and the frame is cropped along the face. Later the cropped frame is again converted to a new video by combining each frame of the video. The process is followed for each video which leads to creation of processed dataset containing face only videos. The frame that does not contain the face is ignored while preprocessing.

To maintain the uniformity of number of frames, we have selected a threshold value based on the mean of total frames count of each video. Another reason for selecting a threshold value is limited computation power. As a video of 10 second at 30 frames per second(fps) will have total 300 frames and it is computationally very difficult to process the 300 frames at a single time in the experimental envi- ronment. So, based on our Graphic Processing Unit (GPU) computational power in experimental environment we have selected 150 frames as the threshold value. While saving the frames to the new dataset we have only saved the first 150 frames of the video to the new video. To demonstrate the proper use of Long Short-Term Memory (LSTM) we have considered the frames in the sequential manner i.e. first 150 frames and not randomly. The newly created video is saved at frame rate of 30 fps and resolution of 112 x 112.







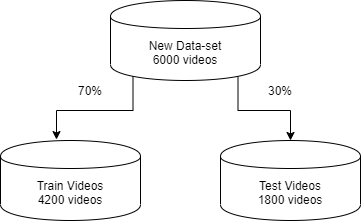
**Figure 6.5: Pre-processing of video**

## Module 3: Data-set split

The dataset is split into train and test dataset with a ratio of 70% train videos (4,200) and 30% (1,800) test videos. The train and test split is a balanced split i.e 50% of the real and 50% of fake videos in each split.







**Figure 6.6: Train test split**

## Module 4: Model Architecture

Our model is a combination of CNN and RNN. We have used the Pre- trained ResNext CNN model to extract the features at frame level and based on the extracted features a LSTM network is trained to classify the video as deepfake or pristine. Us- ing the Data Loader on training split of videos the labels of the videos are loaded and fitted into the model for training.

**ResNext :**

Instead of writing the code from scratch, we used the pre-trained model of ResNext for feature extraction. ResNext is Residual CNN network optimized for high per- formance on deeper neural networks. For the experimental purpose we have used resnext50\_32x4d model. We have used a ResNext of 50 layers and 32 x 4 dimen- sions.

Following, we will be fine-tuning the network by adding extra required layers and selecting a proper learning rate to properly converge the gradient descent of the model. The 2048-dimensional feature vectors after the last pooling layers of ResNext is used as the sequential LSTM input.

**LSTM for Sequence Processing:**

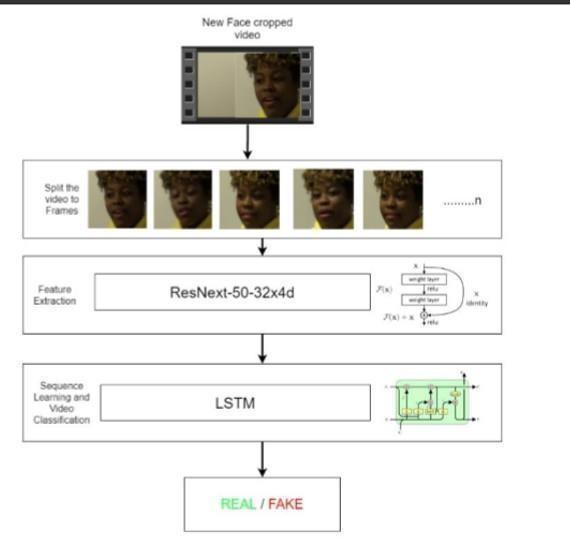
2048-dimensional feature vectors is fitted as the input to the LSTM. We are using 1 LSTM layer with 2048 latent dimensions and 2048 hidden layers along with 0.4 chance of dropout, which is capable to do achieve our objective. LSTM is used to





process the frames in a sequential manner so that the temporal analysis of the video can be made, by comparing the frame at ‘t’ second with the frame of ‘t-n’ seconds. Where n can be any number of frames before t.

The model also consists of Leaky Relu activation function. A linear layer of 2048 input features and 2 output features are used to make the model capable of learning the average rate of correlation between eh input and output. An adaptive average polling layer with the output parameter 1 is used in the model. Which gives the the target output size of the image of the form H x W. For sequential processing of the frames a Sequential Layer is used. The batch size of 4 is used to perform the batch training. A SoftMax layer is used to get the confidence of the model during predication.



**Figure 6 .7: Overview of our model**

## Module 5: Hyper-parameter tuning

It is the process of choosing the perfect hyper-parameters for achieving the maxi- mum accuracy. After reiterating many times on the model. The best hyper-parameters for our dataset are chosen. To enable the adaptive learning rate Adam[21] optimizer with the model parameters is used. The learning rate is tuned to 1e-5 (0.00001) to





achieve a better global minimum of gradient descent. The weight decay used is 1e-3.

As this is a classification problem so to calculate the loss cross entropy approach is used.To use the available computation power properly the batch training is used. The batch size is taken of 4. Batch size of 4 is tested to be ideal size for training in our development environment.

The User Interface for the application is developed using Django framework.

Django is used to enable the scalability of the application in the future.

The first page of the User interface i.e index.html contains a tab to browse and upload the video. The uploaded video is then passed to the model and prediction is made by the model. The model returns the output whether the video is real or fake along with the confidence of the model. The output is rendered in the predict.html on the face of the playing video.





**Chapter 7**

**Project Implementation**

# Introduction

There are many examples where deepfake creation technology is used to mis- lead the people on social media platform by sharing the false deepfake videos of the famous personalities like Mark Zuckerberg Eve of House A.I. Hearing, Don- ald Trump’s Breaking Bad series where he was introduces as James McGill, Barack Obama’s public service announcement and many more [5]. These types of deepfakes creates a huge panic among the normal people, which arises the need to spot these deepfakes accurately so that they can be distinguished from the real videos.

Latest advances in the technology have changed the field of video manipulation. The advances in the modern open source deep learning frameworks like TensorFlow, Keras, PyTorch along with cheap access to the high computation power has driven the paradigm shift. The Conventional autoencoders[10] and Generative Adversarial Network (GAN) pretrained models have made the tampering of the realistic videos and images very easy. Moreover, access to these pretrained models through the smartphones and desktop applications like FaceApp and Face Swap has made the deepfake creation a childish thing. These applications generate a highly realistic synthesized transformation of faces in real videos. These apps also provide the user with more functionalities like changing the face hair style, gender, age and other attributes. These apps also allow the user to create a very high quality and indistin- guishable deepfakes. Although some malignant deepfake videos exist, but till now they remain a minority. So far, the released tools [11,12] that generate deepfake videos are being extensively used to create fake celebrity pornographic videos or revenge porn [13]. Some of the examples are Brad Pitt, Angelina Jolie nude videos. The real looking nature of the deepfake videos makes the celebraties and other fa- mous personalities the target of pornographic material, fake surveillance videos, fake





news and malicious hoaxes. The Deepfakes are very much popular in creating the political tension [14]. Due to which it becomes very important to detect the deepfake videos and avoid the percolation of the deepfakes on the social media platforms.

# Tools and Technologies Used

## Planning

* + - 1. OpenProject

## UML Tools

* + - 1. draw.io

## Programming Languages

* + - 1. Python3
      2. JavaScript

## Programming Frameworks

* + - 1. PyTorch
      2. Django

## IDE

* + - 1. Google Colab
      2. Jupyter Notebook
      3. Visual Studio Code

## Versioning Control

* + - 1. Git

## Cloud Services

* + - 1. Google Cloud Platform





| **7.2.8** | **Application and web**  7.2.8.1 | **servers:**  Google Cloud Engine |
| --- | --- | --- |
| **7.2.9** | **Libraries**  7.2.9.1 |  |
|  | torch |
|  | 7.2.9.2 | torchvision |
|  | 7.2.9.3 | os |
|  | 7.2.9.4 | numpy |
|  | 7.2.9.5 | cv2 |
|  | 7.2.9.6 | matplotlib |
|  | 7.2.9.7 | face\_recognition |
|  | 7.2.9.8 | json |
|  | 7.2.9.9 | pandas |
|  | 7.2.9.10 | copy |
|  | 7.2.9.11 | glob |
|  | 7.2.9.12 | random |
|  | 7.2.9.13 | sklearn |

# Algorithm Details

## Dataset Details

Refer 7.2.1

## Preprocessing Details

* Using glob we imported all the videos in the directory in a python list.
* cv2.VideoCapture is used to read the videos and get the mean number of frames in each video.



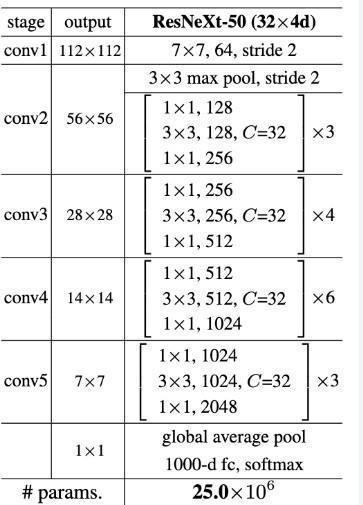


* To maintain uniformity, based on mean a value 150 is selected as idea value for creating the new dataset.
* The video is split into frames and the frames are cropped on face location.
* The face cropped frames are again written to new video using VideoWriter.
* The new video is written at 30 frames per second and with the resolution of 112 x 112 pixels in the mp4 format.
* Instead of selecting the random videos, to make the proper use of LSTM for temporal sequence analysis the first 150 frames are written to the new video.

## Model Details

The model consists of following layers:

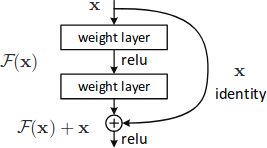
* **ResNext CNN :** The pre-trained model of Residual Convolution Neural Net- work is used. The model name is resnext50\_32x4d()[22]. This model consists of 50 layers and 32 x 4 dimensions. Figure shows the detailed implementation of model.



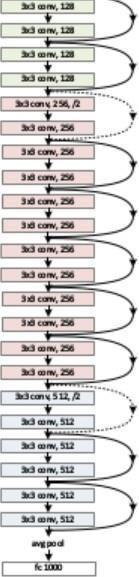
**Figure 7.1: ResNext Architecture**







**Figure 7.2: ResNext Working**

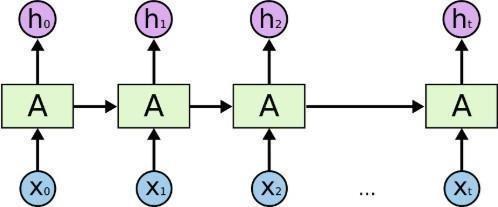


**Figure 7.3: Overview of ResNext Architecture**

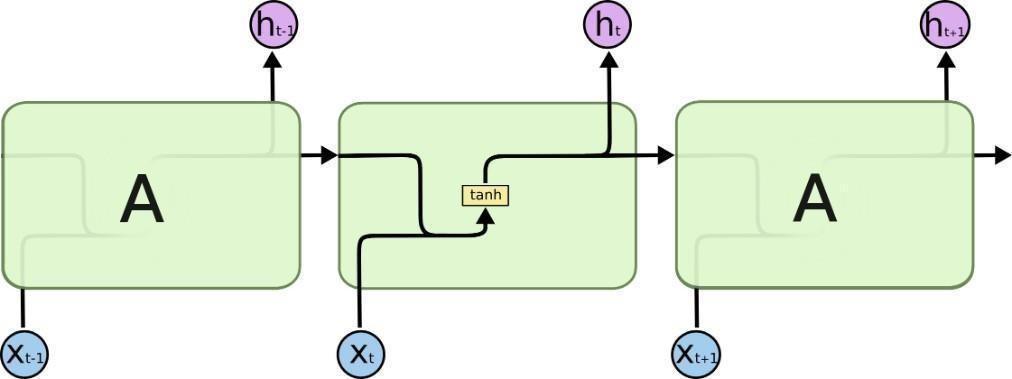
* **Sequential Layer :** Sequential is a container of Modules that can be stacked together and run at the same time. Sequential layer is used to store feature vector returned by the ResNext model in a ordered way. So that it can be passed to the LSTM sequentially.
* **LSTM Layer :** LSTM is used for sequence processing and spot the temporal change between the frames.2048-dimensional feature vectors is fitted as the input to the LSTM. We are using 1 LSTM layer with 2048 latent dimensions and 2048 hidden layers along with 0.4 chance of dropout, which is capable to do achieve our objective. LSTM is used to process the frames in a sequential manner so that the temporal analysis of the video can be made, by comparing the frame at ‘t’ second with the frame of ‘t-n’ seconds. Where n can be any number of frames before t.





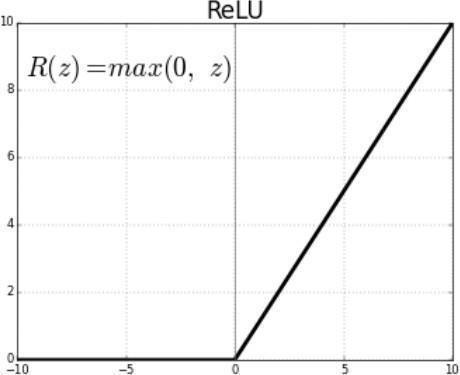


**Figure 7.4: Overview of LSTM Architecture**



**Figure 7.5: Internal LSTM Architecture**

* **ReLU:**A Rectified Linear Unit is activation function that has output 0 if the input is less than 0, and raw output otherwise. That is, if the input is greater than 0, the output is equal to the input. The operation of ReLU is closer to the way our biological neurons work. ReLU is non-linear and has the advantage of not having any backpropagation errors unlike the sigmoid function, also for larger Neural Networks, the speed of building models based off on ReLU is very fast.

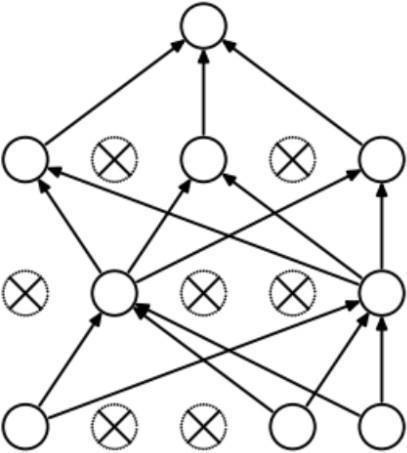


**Figure 7.6: Relu Activation function**





* **Dropout Layer :**Dropout layer with the value of 0.4 is used to avoid over- fitting in the model and it can help a model generalize by randomly setting the output for a given neuron to 0. In setting the output to 0, the cost function becomes more sensitive to neighbouring neurons changing the way the weights will be updated during the process of backpropagation.



**Figure 7.7: Dropout layer overview**

* **Adaptive Average Pooling Layer :** It is used To reduce variance, reduce com- putation complexity and extract low level features from neighbourhood.2 di- mensional Adaptive Average Pooling Layer is used in the model.

## Model Training Details

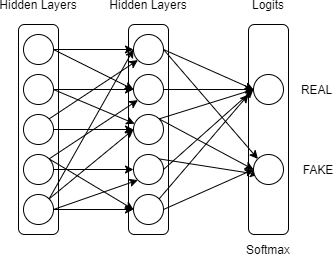
* **Train Test Split:**The dataset is split into train and test dataset with a ratio of 70% train videos (4,200) and 30% (1,800) test videos. The train and test split is a balanced split i.e 50% of the real and 50% of fake videos in each split. Refer figure 7.6
* **Data Loader:** It is used to load the videos and their labels with a batch size of 4.
* **Training:** The training is done for 20 epochs with a learning rate of 1e-5 (0.00001),weight decay of 1e-3 (0.001) using the Adam optimizer.
* **Adam optimizer[21]:** To enable the adaptive learning rate Adam optimizer with the model parameters is used.





* **Cross Entropy:** To calculate the loss function Cross Entropy approach is used because we are training a classification problem.
* **Softmax Layer:** A Softmax function is a type of squashing function. Squash- ing functions limit the output of the function into the range 0 to 1. This allows the output to be interpreted directly as a probability. Similarly, softmax func- tions are multi-class sigmoids, meaning they are used in determining probabil- ity of multiple classes at once. Since the outputs of a softmax function can be interpreted as a probability (i.e.they must sum to 1), a softmax layer is typically the final layer used in neural network functions. It is important to note that a softmax layer must have the same number of nodes as the output later.

In our case softmax layer has two output nodes i.e REAL or FAKE, also Soft- max layer provide us the confidence(probability) of prediction.



**Figure 7.8: Softmax Layer**

* **Confusion Matrix:** A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which your





classification model is confused when it makes predictions. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

Confusion matrix is used to evaluate our model and calculate the accuracy.

* **Export Model:** After the model is trained, we have exported the model. So that it can be used for prediction on real time data.

## Model Prediction Details

* + The model is loaded in the application
* The new video for prediction is preprocessed(refer 8.3.2, 7.2.2) and passed to the loaded model for prediction
* The trained model performs the prediction and return if the video is a real or fake along with the confidence of the prediction.





**Chapter 8 Software Testing**

# Type 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





# Test Cases and Test Results

**Test Cases**

**Table 8.1:** Test Case Report

| Case id | Test Case Description | Expected Result | Actual Result | Status |
| --- | --- | --- | --- | --- |
| 1 | Upload a word file in- stead of video | Error message: Only video files allowed | Error message: Only video files allowed | Pass |
| 2 | Upload a 200MB video file | Error message: Max limit 100MB | Error message: Max limit 100MB | Pass |
| 3 | Upload a file without any faces | Error message:No faces detected. Cannot pro- cess the video. | Error message:No faces detected. Cannot pro- cess the video. | Pass |
| 4 | Videos with many faces | Fake / Real | Fake | Pass |
| 5 | Deepfake video | Fake | Fake | Pass |
| 6 | Enter /predict in URL | Redirect to /upload | Redirect to /upload | Pass |
| 7 | Press upload button without selecting video | Alert message: Please select video | Alert message: Please select video | Pass |
| 8 | Upload a Real video | Real | Real | Pass |
| 9 | Upload a face cropped real video | Real | Real | Pass |
| 10 | Upload a face cropped fake video | Fake | Fake | Pass |

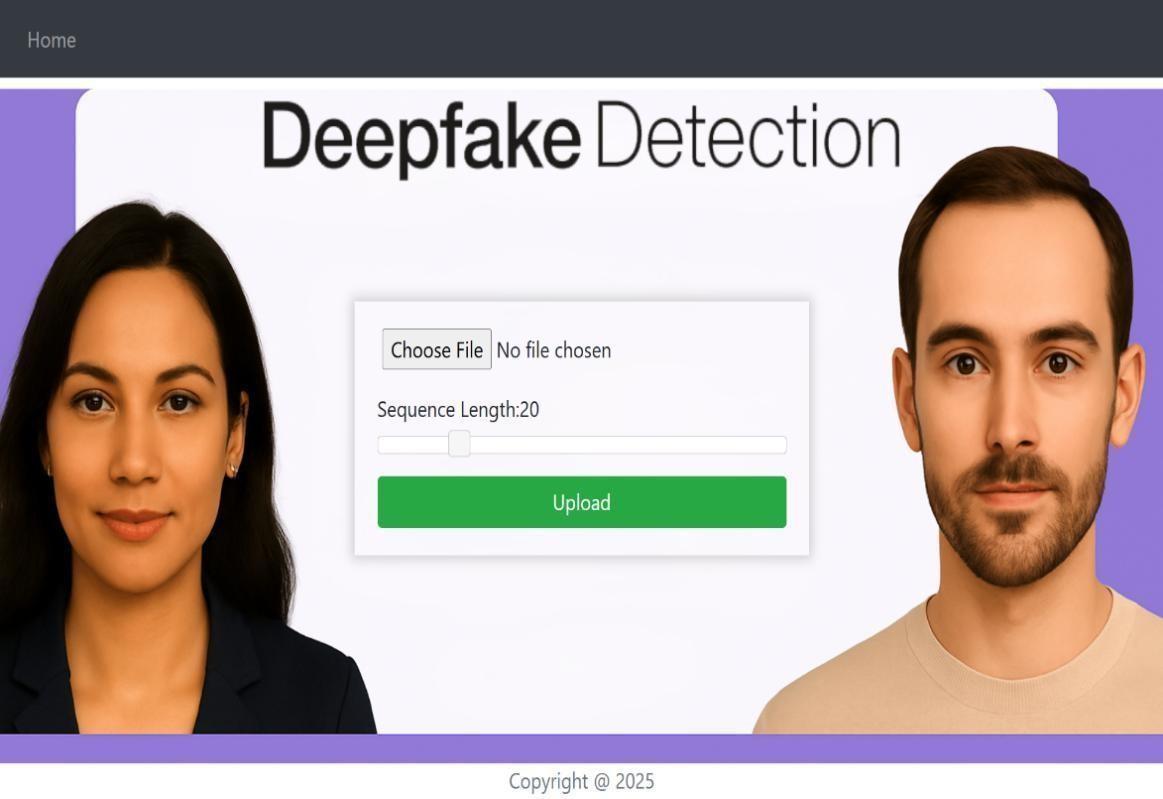




**Chapter 9**

**Results and Discussion**

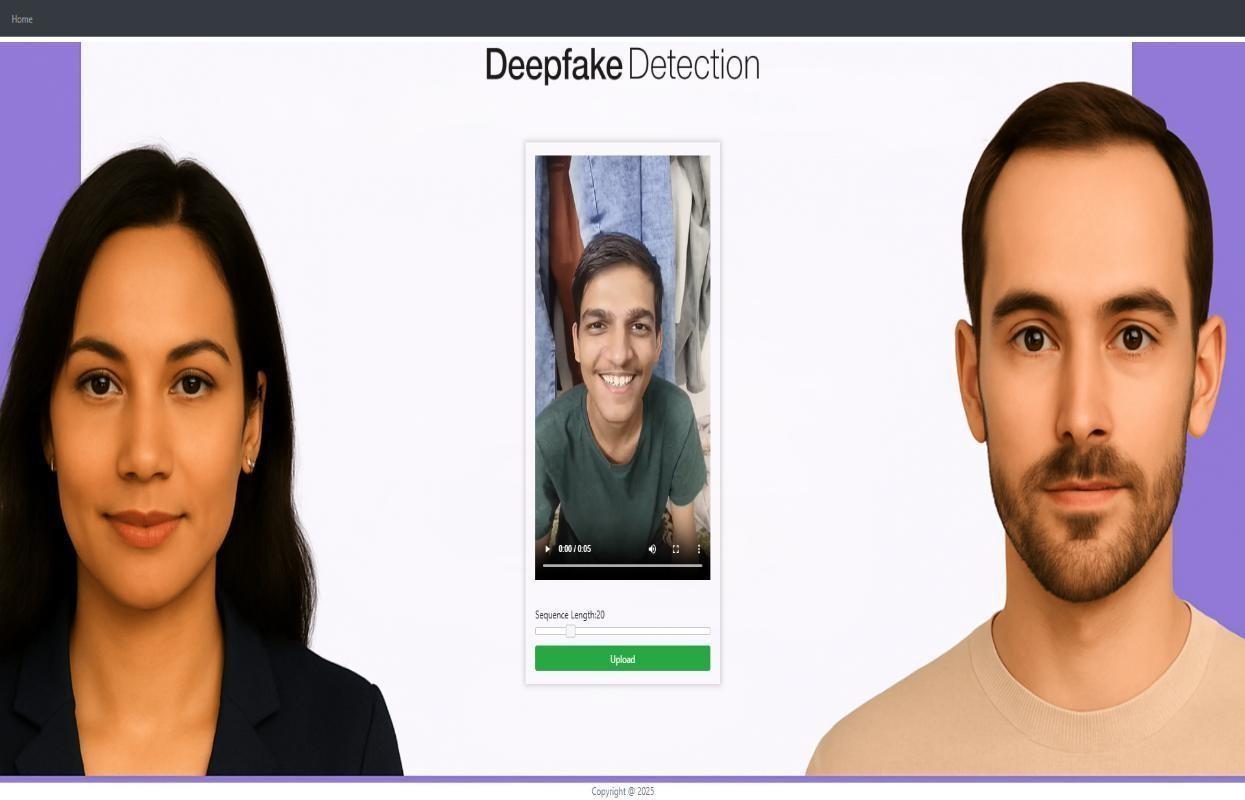
# Screen shots



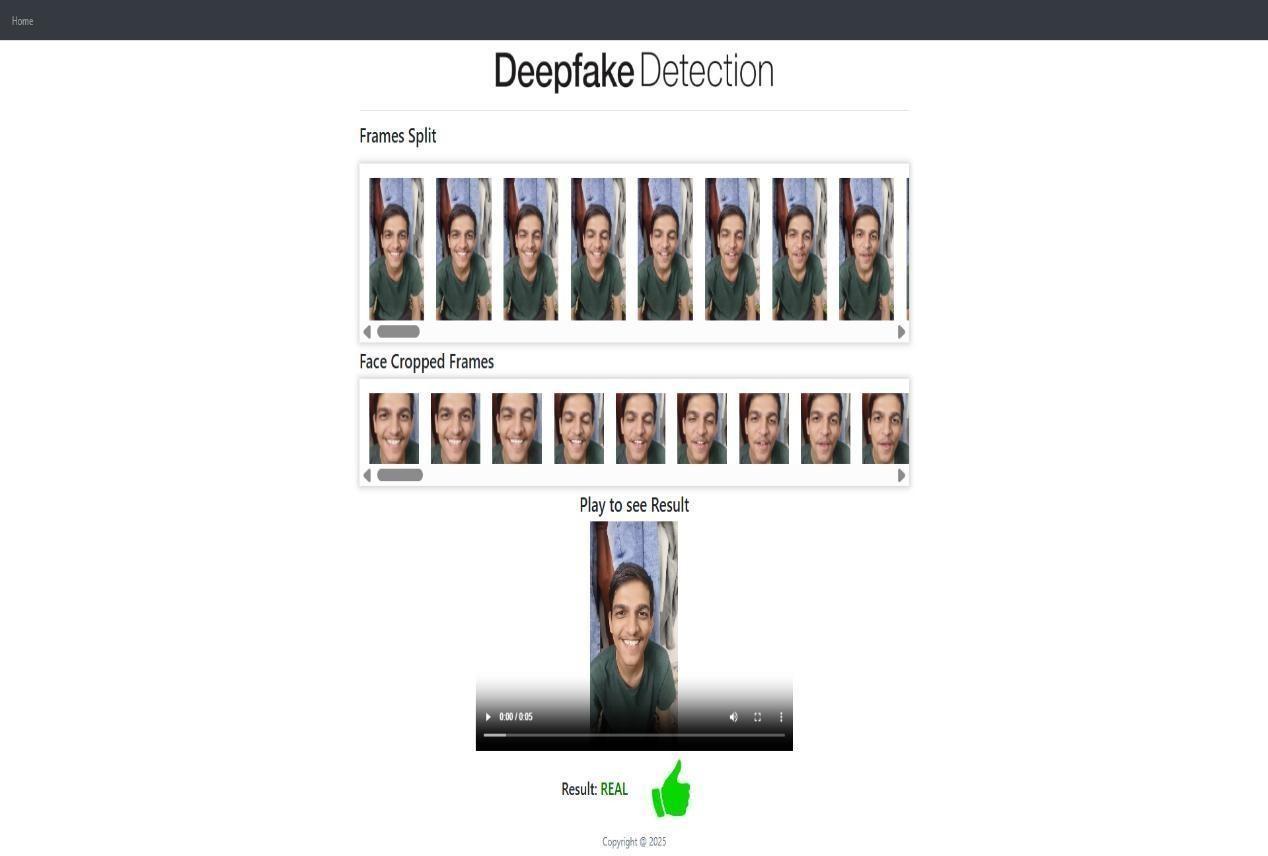
**Figure 9.1: Home Page**







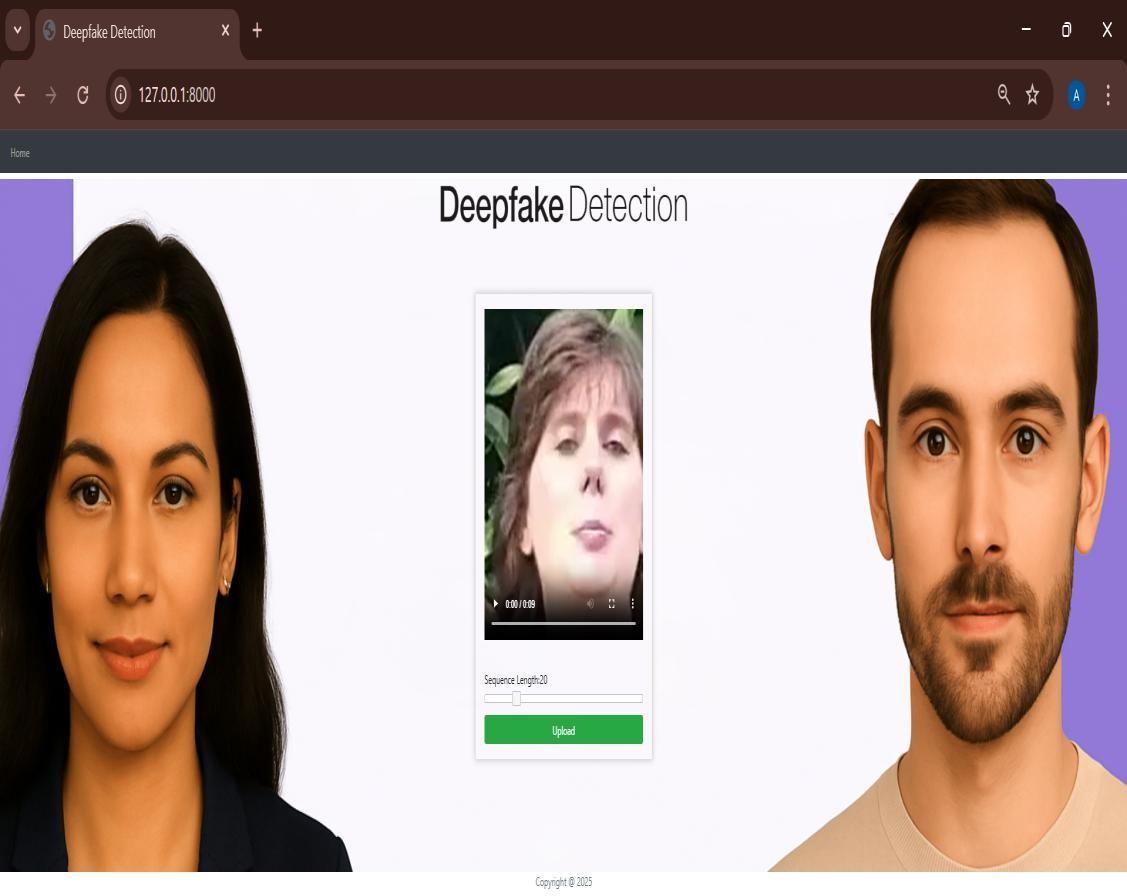
**Figure 9.2: Uploading Real Video**



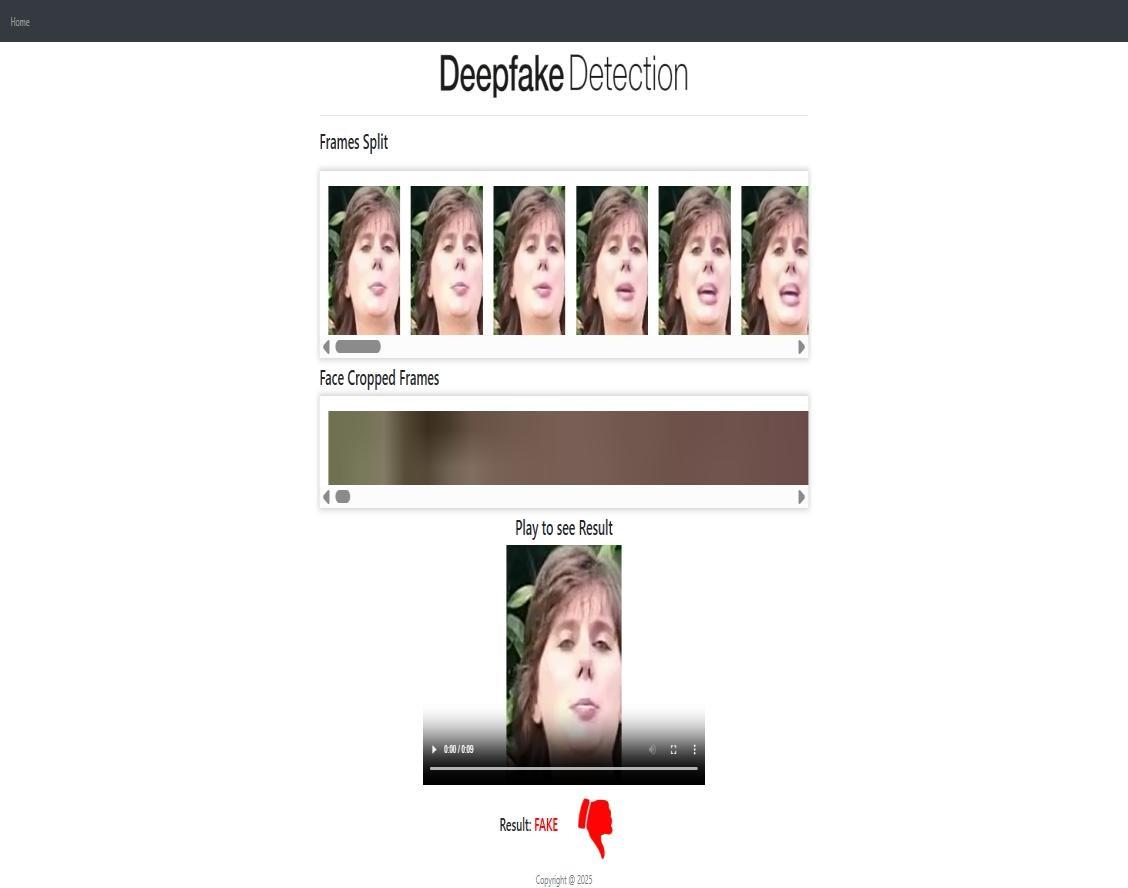
**Figure 9.3: Real Video Output**







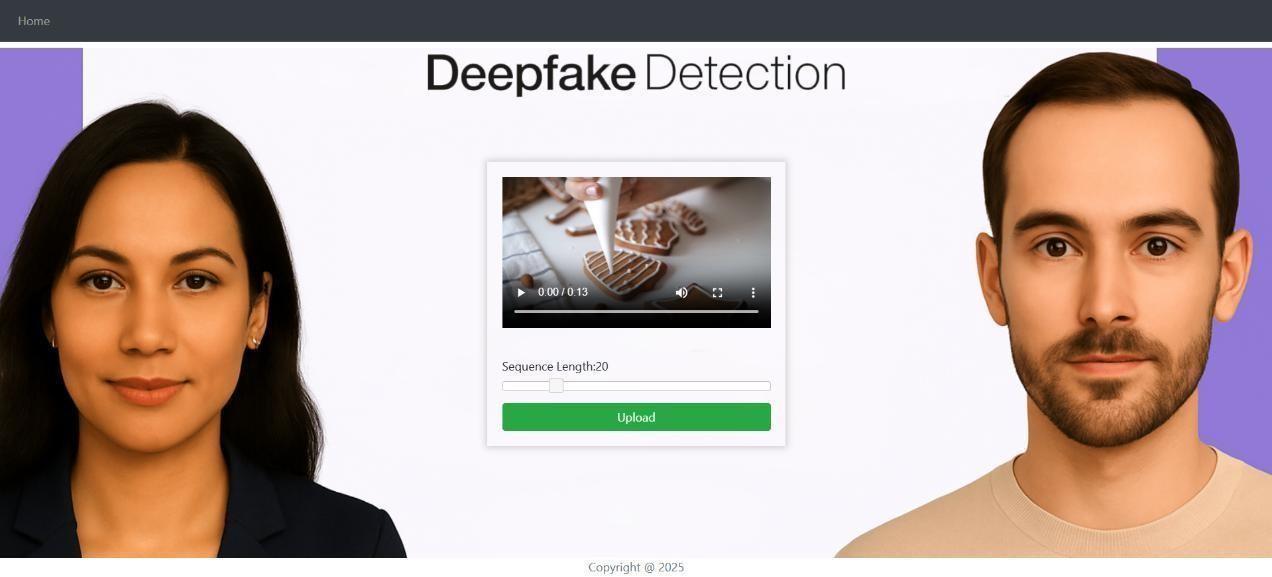
**Figure 9.4: Uploading Fake Video**



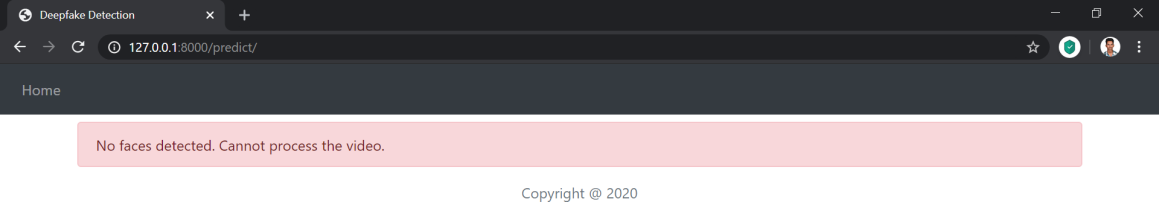
**Figure 9.5: Fake video Output**







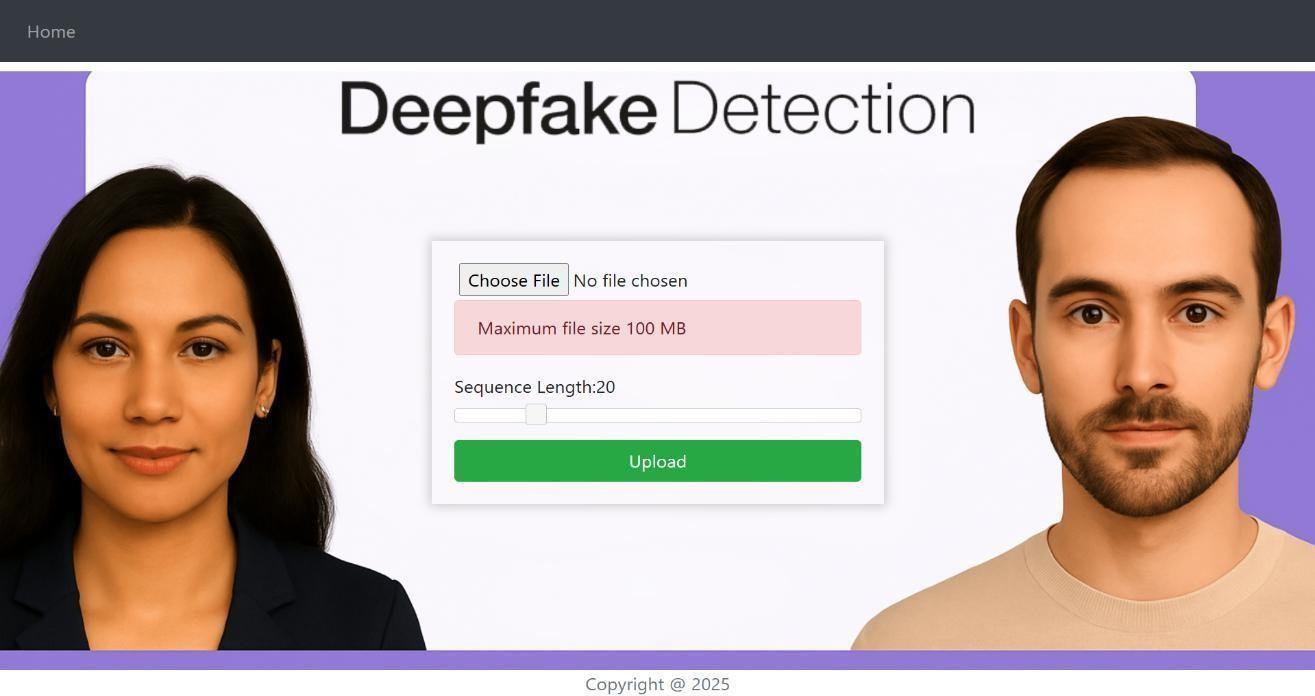
**Figure 9.6: Uploading Video with no faces**



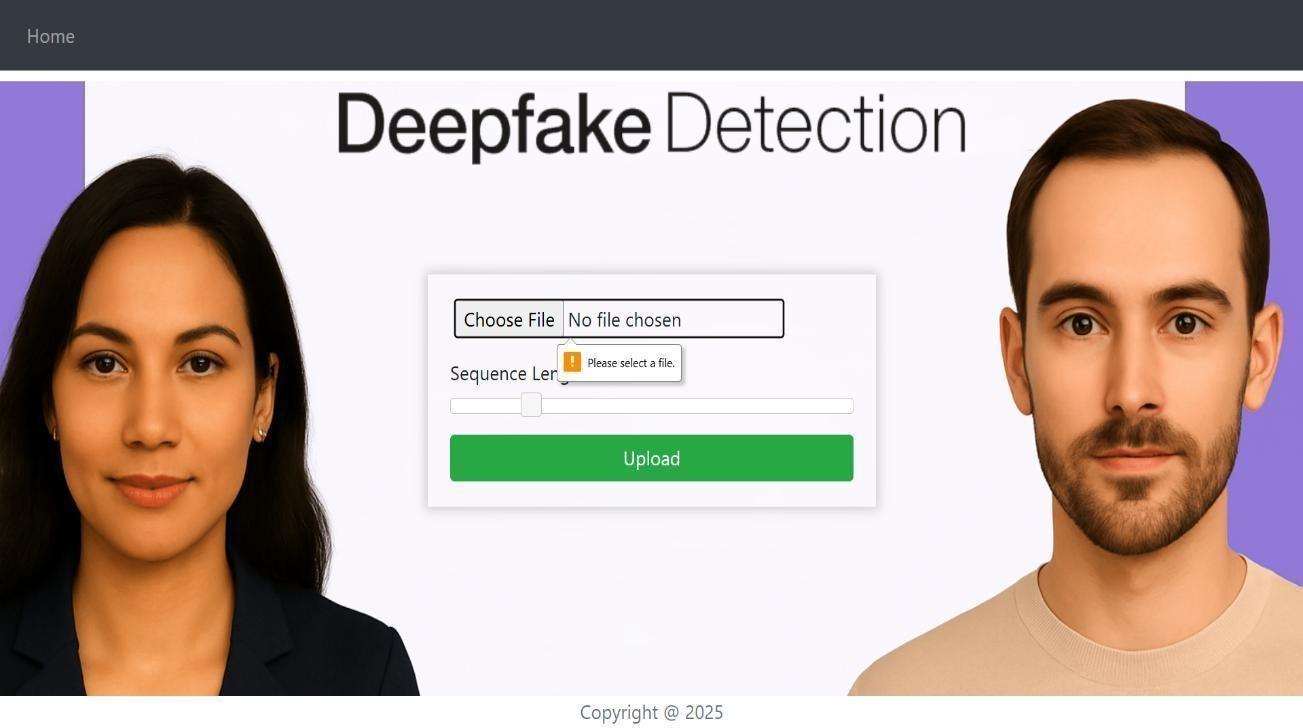
**Figure 9.7: Output of Uploaded video with no faces**







**Figure 9.8: Uploading file greater than 100MB**



**Figure 9.9: Pressing Upload button without selecting video**

# Outputs

## Model results





**Table 9.1:** Trained Model Results

| Model Name | Dataset | No. of videos | Sequence length | Accuracy |
| --- | --- | --- | --- | --- |
| model\_90\_acc  \_20\_frames\_ FF\_data | FaceForensic++ | 2000 | 20 | 90.95477 |
| model\_95\_acc  \_40\_frames\_ FF\_data | FaceForensic++ | 2000 | 40 | 95.22613 |
| model\_97\_acc  \_60\_frames\_ FF\_data | FaceForensic++ | 2000 | 60 | 97.48743 |
| model\_97\_acc  \_80\_frames\_ FF\_data | FaceForensic++ | 2000 | 80 | 97.73366 |
| model\_97\_acc  \_100\_frames\_ FF\_data | FaceForensic++ | 2000 | 100 | 97.76180 |
| model\_93\_acc  \_100\_frames\_ celeb\_FF\_data | Celeb-DF + FaceForen- sic++ | 3000 | 100 | 93.97781 |
| model\_87\_acc  \_20\_frames\_ final\_data | Our Dataset | 6000 | 20 | 87.79160 |
| model\_84\_acc  \_10\_frames\_ final\_data | Our Dataset | 6000 | 10 | 84.21461 |
| model\_89\_acc  \_40\_frames\_ final\_data | Our Dataset | 6000 | 40 | 89.34681 |





**Chapter 10**

**Conclusion and Future Scope**

# Conclusion

We presented a neural network-based approach to classify the video as deep fake or real, along with the confidence of proposed model. Our method is capable of predicting the output by processing 1 second of video (10 frames per second) with a good accuracy. We implemented the model by using pre-trained ResNext CNN model to extract the frame level features and LSTM for temporal sequence process- ing to spot the changes between the t and t-1 frame. Our model can process the video in the frame sequence of 10,20,40,60,80,100.

# Future Scope

There is always a scope for enhancements in any developed system, especially when the project build using latest trending technology and has a good scope in future.

* + 1. Web based platform can be upscaled to a browser plugin for ease of access to the user.
    2. Currently only Face Deep Fakes are being detected by the algorithm, but the algorithm can be enhanced in detecting full body deep fakes.





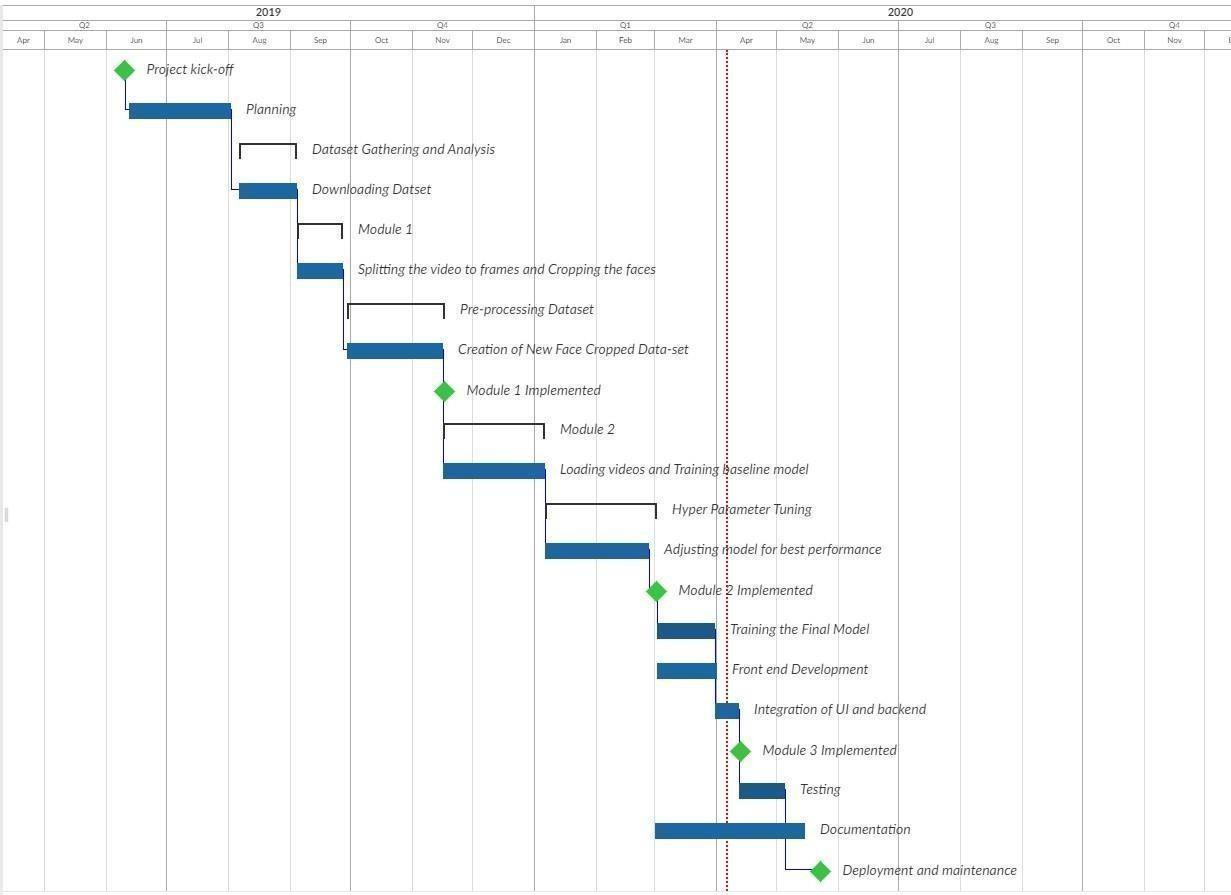
**Appendix A References**

* Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
* Kumar, M., Dutta, A., & Gupta, A. (2021). Deepfake Detection: A Comprehensive Review of Techniques and Approaches. Journal of Computer Science and Technology, 36(5), 932–963.
* Chesney, B., & Citron, D. K. (2019). Deep Fakes: A Looming Challenge for Privacy, Democracy, and National Security. California Law Review, 107(1), 175–203.
* Li, Y., Wu, Y., & Zhang, H. (2020). A Hybrid Deep Learning Framework for Deepfake Detection. International Journal of Information Technology, 12(3), 841–848.
* Aggregated Residual Transformations for Deep Neural Networks. Proceedings of CVPR, 1492–1500. Zhou, P., & Hu, W. (2020). A Survey on Deepfake Detection. Journal of Visual Communication and Image Representation, 71, 102850.
* Marra, F., et al. (2020). A New Era for Deepfake Detection: Transfer Learning, and Data Efficiency. Proceedings of CVPR, 1701–1710.
* Perez, L., & Wang, J. (2017). Effectiveness of Data Augmentation in Image Classification using Deep Learning. Convolutional Neural Networks Visions, 9(2), 181– 192.
* Sabour, S., Frosst, N., & Hinton, G. E. (2017). Dynamic Routing Between Capsules. Advances in Neural Information Processing Systems, 30. Xie, S., Girshick, R., Farhadi, A., &He, K. (2017).
* Gao, J., Xu, K., & Guo, Q. (2021). A Comparative Study on Deepfake Detection Techniques. Computers, Materials & Continua, 67(2), 1387–1404.





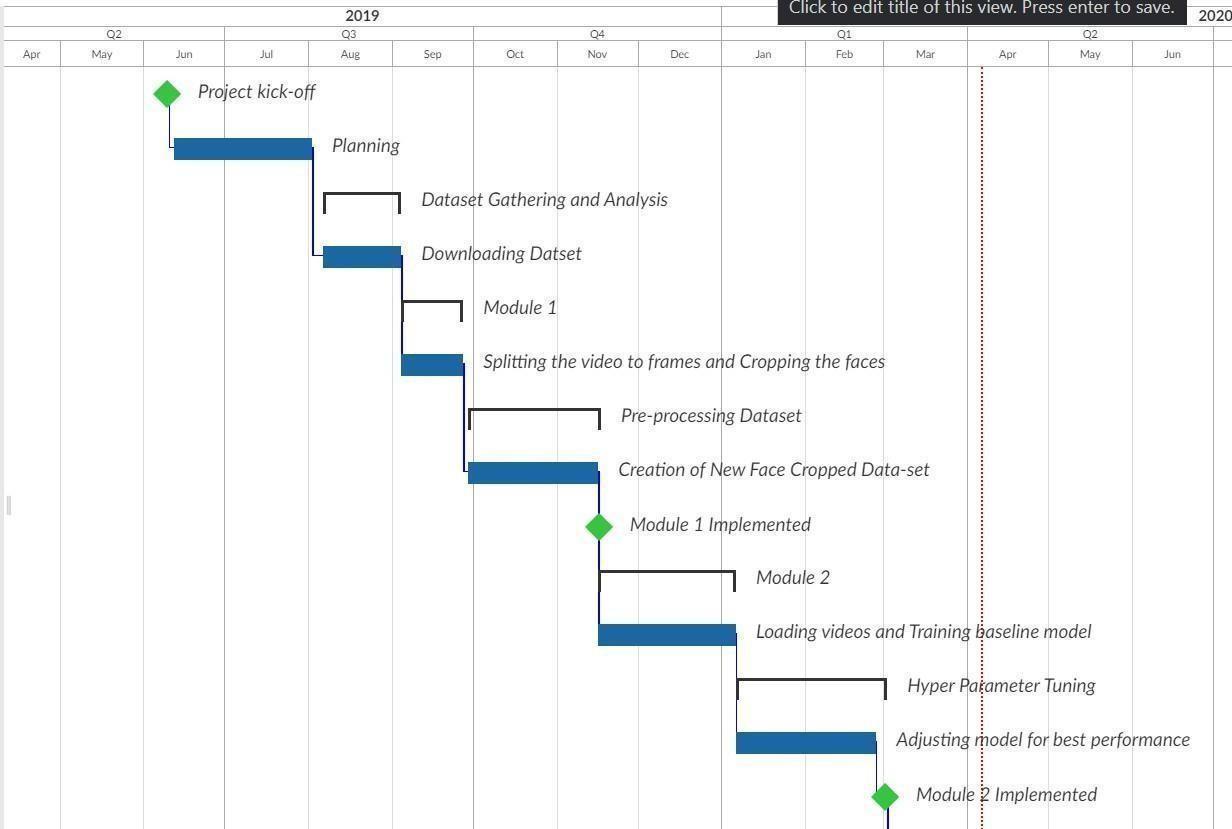
**Appendix B Project Planner**



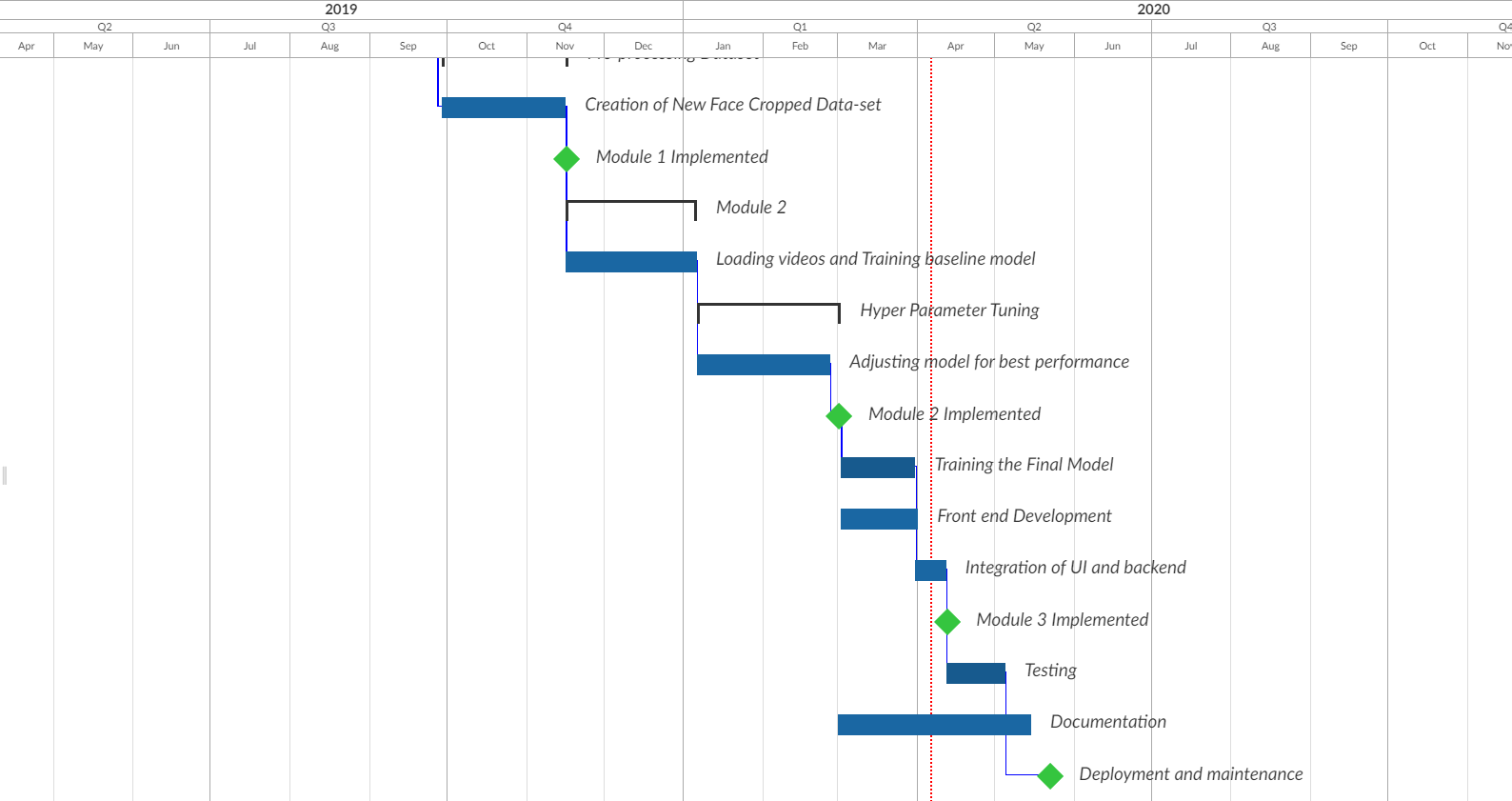
**Figure B.1: Complete Project Plan**







**Figure B.2: Project Plan 1**



**Figure B.3: Project Plan 2**





**Appendix C Timeline**

