A

Project Report

On

Deepfake Detection System

SUBMITTED TOWARDS THE
FULFILLMENT OF THE REQUIREMENTS OF

Bachelor Of Engineering (Computer Engineering)

BY

Avatade Rohan Ramdas	Exam No: B400190184
Bagal Yashkumar Ramchandra	Exam No: B400190185
Nigade Jaydeep Popatrao	Exam No: B400190240
Shinde Nikhil Kailas	Exam No: B400190261

Under The Guidance of

Prof. R.K.Taware

Department Of Computer Engineering
SVPM's College Of Engineering,Malegaon(Bk.),Baramati,Pune-413115.

SAVITRIBAI PHULE PUNE UNIVERSITY
2024-25

SVPM's COLLEGE OF ENGINEERING, DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the Project Entitled

Deepfake Detection System

Submitted by

Avatade Rohan Ramdas	Exam No: B400190184
Bagal Yashkumar Ramchandra	Exam No: B400190185
Nigade Jaydeep Popatrao	Exam No: B400190240
Shinde Nikhil Kailas	Exam No: B400190261

is a bonafide work carried out by Students under the supervision of Prof. R.K.Taware and it is submitted towards the fulfillment of the requirement of Bachelor of Engineering (Computer Engineering) Project.

Prof. R.K.Taware
Internal Guide

Prof. Khalate.Y.R
H.O.D

External Examiner

Dr. S.M.Mukane
Principal

Place : SVPM's COE Malegaon(Bk.)
Date :

PROJECT APPROVAL SHEET

A

Final Project

on

(Deepfake Detection System)

Is successfully completed by

Avatade Rohan Ramdas	Exam No: B400190184
Bagal Yashkumar Ramchandra	Exam No: B400190185
Nigade Jaydeep Popatrao	Exam No: B400190240
Shinde Nikhil Kailas	Exam No: B400190261

at

Department Of Computer Engineering
SVPM's College Of Engineering,Malegaon(Bk.), Baramati,Pune-413115.

SAVITRIBAI PHULE PUNE UNIVERSITY
2024-25

Prof. R.K.Taware
Project Guide

Prof. Khalate.Y.R
HOD

Abstract

The rise of deepfake technology, powered by advancements in artificial intelligence, poses significant risks in the realm of digital media, challenging the integrity and authenticity of visual and audio content. Detecting deepfakes—sophisticated manipulations of video, images, and audio—requires innovative solutions that can keep up with rapid developments in generative AI. This paper presents a novel hybrid model specifically designed to enhance the detection of deepfakes by combining Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) units within a Recurrent Neural Network (RNN) framework. CNNs are leveraged for spatial feature extraction, capturing subtle artifacts within individual frames, while LSTM units analyze temporal consistency across consecutive frames, addressing the unique characteristics of manipulated content over time.

The model's performance is rigorously evaluated using publicly available datasets, including FaceForensics++ and the Deepfake Detection Challenge (DFDC) dataset, covering a wide range of manipulation techniques and video resolutions. The experimental results indicate that our approach achieves an overall detection accuracy of 92

Keywords: Deepfake detection, Hybrid detection model, Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), Temporal consistency analysis, FaceForensics++ dataset, Deepfake Detection Challenge.

Acknowledgments

It gives us great pleasure in presenting the preliminary project report on ***Deepfake Detection System****.*

*I would like to take this opportunity to thank my internal guide***Prof. R.K.Taware**
Project Coordinator : Prof.R.K. Taware
HOD : Prof.Y.R.Khalate
PRINCIPAL : Dr.S.M.Mukane

Avatade Rohan Ramdas
Bagal Yashkumar Ramchandra
Nigade Jaydeep Popatrao
Shinde Nikhil Kailas
(B.E. Computer Engg.)

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Introduction

In the world of ever-growing social media platforms, Deepfakes are considered as the major threat of the AI. There are many Scenarios where these realistic face swapped deepfakes are used to create political distress, fake terrorism events, revenge porn, blackmail peoples are easily envisioned. Some of the examples are Brad Pitt, Angelina Jolie nude videos.
It becomes very important to spot the difference between the deepfake and pristine video. We are using AI to fight AI. Deep fakes are created using tools like Face App [11] and Face Swap [12], which using 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 Res-Next CNN to extract the frame level features. Res Next 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 large amount of balanced and combination of various available dataset like Face Forensic++ [1], Deepfake detection challenge [2], and Celeb-DF [3]. Further to make the ready to use for the customers, we have developed a front end application where the user 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.

1.2 Objectives

The primary goal of this project is to create an AI-driven system that can effectively detect and differentiate deepfake videos from genuine ones. The system utilizes a hybrid approach combining Res Next Convolutional Neural Networks for extracting detailed features from individual video frames and an LSTM-based Recurrent Neural Network to analyse the sequence of frames over time. This enhances both the accuracy and robustness of deepfake detection.
To ensure the system performs well in practical applications, it is trained on a diverse, well-balanced dataset that merges multiple reliable sources, including Face Forensics++, the Deepfake Detection Challenge (DFDC) dataset, and Celeb-DF. A simple and interactive front-end interface has also been developed, enabling users to upload videos and receive classification outputs along with a model confidence score. This project aims to serve as a proactive solution against the rising misuse of deepfake technology, which poses threats in areas like misinformation, political manipulation, cybercrime, and personal defamation.

1.3 Scope of the Project

While numerous tools exist for generating deepfakes, effective solutions for detecting them remain limited. Our proposed system offers a significant advancement in addressing the spread of manipulated media across the internet. The project includes a web-based platform where users can upload videos and receive a classification indicating whether the video is authentic or altered.
This solution not only aids in immediate detection but also holds potential for future expansion. The platform can be extended into a browser extension for real-time deepfake detection. Furthermore, popular social media and messaging applications like WhatsApp and Facebook could integrate this technology into their systems to automatically screen media before it is shared, enhancing user safety and curbing misinformation.
The software specification outlines key aspects such as input size limitations, validation rules, input dependencies, and an overview of major inputs and outputs. These are detailed with a focus on system behaviour and user interaction, without delving into the underlying implementation.

1.4 Motivation of the Project

With the rapid advancement of smartphone cameras and the widespread influence of social media platforms, creating and distributing digital videos has become easier than ever. Alongside these developments, deep learning technologies have introduced groundbreaking capabilities, such as generating highly realistic images, voices, music, and videos through advanced generative models. These tools have been leveraged for beneficial purposes like improving accessibility via text-to-speech systems and generating synthetic data for medical research and diagnostics.

However, like any powerful technology, these innovations come with significant challenges. One of the most concerning developments is the rise of deepfake AI-generated content that manipulates visuals and audio in a realistic yet deceptive way. Since their emergence in 2017, a growing number of open-source tools have made deepfake creation more accessible, resulting in an increase in both quantity and quality of synthetic media. While some of these creations may be harmless or comedic, others pose serious threats to individuals and society at large.

The widespread circulation of deepfakes on platforms like Facebook, Instagram, and WhatsApp has become a serious issue. These fake videos can easily mislead the public, spread misinformation, and even incite panic. Imagine a fabricated video of a nation's leader announcing war, or a forged clip of a respected public figure making offensive statements—such scenarios could have disastrous consequences.

To tackle this emerging threat, it is crucial to develop robust systems for identifying and stopping deepfakes before they spread. In response to this need, we propose a deep learning-based detection method designed to reliably differentiate between real and AI-manipulated videos. This technology plays a vital role in safeguarding digital trust and preventing the harmful impact of synthetic media.

Chapter 2

LITERATURE SURVEY

2.1 Literature Survey

The literature survey provides an overview of various methods proposed for deepfake detection. These techniques include artifact detection, biological signal extraction, eye blinking analysis, and the use of neural network models such as CNNs, RNNs, and Capsule Networks. A comparison of these methods is presented in the table below.

2.2 Proposed System

To address the limitations present in current deepfake detection techniques, we introduce a reliable and intelligent system that combines both spatial and temporal analysis for improved video classification. Our solution leverages a pre-trained ResNeXt Convolutional Neural Network (CNN) to extract meaningful features from each video frame. These extracted features are then fed into a Long Short-Term Memory (LSTM) based Recurrent Neural Network (RNN), which analyses the temporal sequence of frames to detect inconsistencies in facial expressions and motion—subtleties that often go unnoticed in static frame analysis.

Unlike existing approaches that depend solely on visual clues such as eye blinking, face blending errors, or physiological signals, our method integrates both spatial context and temporal patterns to enhance the generalization and accuracy of detection across various video types.

To ensure robustness and high performance in real-world scenarios, the model is trained on a comprehensive and balanced dataset compiled from leading sources like FaceForensics++, the Deepfake Detection Challenge (DFDC), and Celeb-DF, while avoiding artificial noise and minimizing the risk of over-

Sr. No.	Paper Title and Author(s)	Description	Approach Used
1	**Face Warping Artifacts** Yuezun Li, Siwei Lyu	Detects warping artifacts using a CNN by comparing generated face areas with surroundings. Ignores temporal frame analysis.	Focus on frame-level spatial inconsistencies.
2	**Detection by Eye Blinking** Yuezun Li, Ming-Ching Chang, Siwei Lyu	Uses LRCN for blinking pattern detection. Alone not reliable; suggests using other facial cues too.	Combines multiple facial features including blinking and wrinkles.
3	**Capsule Networks to Detect Forged Images and Videos** Afchar Darius et al.	Applies capsule networks with random noise in training. Limited performance in clean data.	Uses stable training on clean data with enhanced feature extraction.
4	**RNN for Deepfake Detection** Tariq Sohail et al.	Uses RNN on HOHO dataset with low diversity. Real-world generalization is limited.	Uses larger, diverse datasets for temporal modeling.
5	**Biological Signals in Portrait Videos** Ciftci Umur et al.	Extracts facial biological signals (PPG maps) and classifies using CNN and SVM.	Integrates both biological and visual features.

Table 2.1: Summary of Literature Survey

fitting. The end product includes a simple, intuitive front-end platform that enables users to upload videos and instantly view classification outcomes along with a confidence score, making the system practical, scalable, and deployment-ready for real-world use.

Chapter 3

Software Requirements Specifications

3.1 Introduction

This Software Requirements Specification (SRS) document serves to define the functional and non-functional requirements of an AI-driven deepfake detection system. As deep learning technologies continue to evolve, the creation of synthetic media—commonly referred to as deepfakes—has become alarmingly accessible. These AI-manipulated videos pose a serious threat, enabling the rapid spread of false information, political manipulation, reputational harm, and cybercrimes such as blackmail and revenge porn. Despite the ease of generating deepfakes using tools powered by generative models like GANs and autoencoders, detecting such fabricated content remains a complex and unresolved issue.

To address this challenge, the proposed solution utilizes a combination of ResNeXt Convolutional Neural Networks (CNNs) for spatial feature extraction and Long Short-Term Memory (LSTM) networks for temporal sequence analysis, enabling the system to effectively differentiate between authentic and manipulated videos. This document outlines the detailed software requirements for the system, covering its purpose, scope, functionalities, performance criteria, user interface design, and technical specifications. It is intended for use by developers, researchers, QA testers, and project stakeholders involved in building and refining the deepfake detection platform.

3.1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to deliver a clear, structured, and detailed overview of the deepfake detection system powered by artificial intelligence. This document defines the core functionalities, system constraints, design principles, and interface requirements essential for the successful development and deployment of the application. The system's primary objective is to identify and limit the circulation of deepfake videos—synthetically manipulated content often used for harmful purposes such as spreading misinformation, political sabotage, blackmail, and reputational damage.

This SRS acts as a foundational guide for developers, project leads, testers, and other stakeholders, ensuring consistency and clarity throughout all stages of the software development process. It supports alignment between technical implementation and user expectations, emphasizing real-time performance, scalability, and ease of use in real-world conditions.

3.1.2 Scope

While numerous tools exist for generating deepfake content, effective and accessible solutions for detecting such manipulated media are still scarce. Our proposed system addresses this critical gap by offering a web-based platform that enables users to upload videos and receive instant classification results identifying whether the video is authentic or artificially generated. By providing an intuitive interface and robust AI-powered backend, this project contributes significantly to curbing the spread of deceptive media across the internet.

Looking ahead, this solution can be extended beyond a web platform into a browser extension or plugin, allowing for automatic deepfake detection directly within users' browsing environments. Furthermore, integration into popular communication platforms such as WhatsApp and Facebook could provide pre-screening capabilities, helping to identify and flag manipulated videos before they are shared.

The system design includes comprehensive specifications, including input size, boundaries, validation mechanisms, input-output dependencies, and state diagrams to detail the flow and processing of data. These design elements are described at a high level, without focusing on the implementation details, providing a clear understanding of the system's behaviour and interaction.

3.1.3 Problem Statement

For years, digital manipulation of images and videos has been possible through visual effects. However, the emergence of deep learning has significantly advanced the realism and ease with which such synthetic content commonly known as deepfakes can be created. With the rise of AI-generated media, producing deepfakes has become relatively simple, making their misuse a growing concern.
While generating deepfakes is now more accessible than ever, accurately detecting them remains a complex challenge. Numerous incidents have shown how deepfakes can be weaponized to fuel political unrest, simulate acts of terrorism, create non-consensual explicit content, and carry out blackmail. Given the severe societal implications, it is crucial to develop effective systems to identify and prevent the spread of such manipulated content across digital platforms.
In response to this need, we propose a deepfake detection system that leverages an LSTM-based artificial neural network, capable of analysing video data to identify signs of tampering. This initiative represents a proactive step toward countering the misuse of AI-generated media and protecting the integrity of online information.

3.1.4 Overview

This Software Requirements Specification (SRS) outlines the architecture and key functionalities of an AI-driven deepfake detection system. The primary objective of this system is to accurately differentiate between genuine and manipulated videos using advanced deep learning techniques. To achieve this, the system employs a pre-trained ResNeXt Convolutional Neural Network (CNN) for extracting spatial features from individual video frames, along with a Long Short-Term Memory (LSTM) based Recurrent Neural Network (RNN) to capture and analyze temporal relationships across frames. This combination ensures a more reliable and robust detection of deepfakes.

3.2 Overall Description

The deepfake detection system is a specialized software application built to examine digital video content and determine its authenticity using advanced artificial intelligence methods. At its core, the system incorporates deep learning models—specifically, a ResNeXt-based Convolutional Neural Network (CNN) to extract spatial features from each frame, and a Long Short-

Term Memory (LSTM) Recurrent Neural Network (RNN) to evaluate the temporal flow and coherence between frames. This dual-model approach enhances the system's ability to spot subtle irregularities and unnatural movements that are typical in deepfake videos generated by AI.
Designed with usability in mind, the platform features an intuitive web interface, enabling users regardless of technical background to upload videos for analysis. After submission, the backend AI engine processes the input and delivers a result indicating whether the content is genuine or manipulated, accompanied by a confidence level score.

3.2.1 Product Perspective

The deepfake detection system functions as an independent, web-based solution that enables users to verify the authenticity of video content using advanced deep learning methodologies. It employs a modular client-server architecture, where the front-end handles user interactions—such as video uploads and result displays—while the backend is responsible for intensive processing and executing the AI model.
The system incorporates a pre-trained ResNeXt Convolutional Neural Network (CNN) for extracting spatial features from individual video frames and a Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) to assess the temporal continuity and detect manipulations commonly found in deepfake content.
The front-end is built using modern web technologies like ReactJS, ensuring a responsive and user-friendly interface accessible via standard web browsers without requiring any installation. Data exchange between the client and server is facilitated through RESTful APIs using JSON format. On the backend, frameworks like Flask or Django in Python are used, along with powerful machine learning libraries such as PyTorch or TensorFlow for model inference. OpenCV supports the extraction and preprocessing of video frames. While the front-end can run on typical user hardware, the backend requires GPU-enabled machines for real-time performance and high-accuracy predictions.
Though the system is self-contained, its architecture allows easy integration with broader applications such as social media monitoring systems, digital forensic tools, or misinformation detection frameworks. It is compatible with both on-premise servers and cloud environments like AWS or Google Cloud, offering scalability and deployment flexibility to suit different use cases and organizational needs.

3.2.2 User Characteristics

The deepfake detection system is intended for a wide range of users, including everyday individuals, journalists, cybersecurity experts, law enforcement personnel, and academic researchers. Most end-users are expected to possess basic digital skills, such as uploading video files, navigating a simple web interface, and understanding basic classification results. The front-end is designed with ease of use in mind, requiring no prior knowledge of artificial intelligence or machine learning concepts.
On the other hand, users managing the backend such as system developers, maintainers, or AI professionals may need more advanced knowledge in areas like neural network design, video data processing, and system infrastructure. These users are typically responsible for maintaining system performance, updating models with new datasets, and ensuring overall reliability and security. The system is thus built to be inclusive and accessible, providing simplicity for general users while allowing in-depth control for technical stakeholders.

3.3 Specific Requirements

The deepfake detection system is designed to meet a range of specific functional and non-functional requirements that ensure accurate, efficient, and secure operation. The main functionality allows users to upload video files through a web interface. After the upload, the system extracts individual frames from the video using video processing tools. These frames are analysed by a pre-trained ResNeXt-based Convolutional Neural Network (CNN) to extract spatial features, which are then passed to a Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) for temporal sequence analysis. Based on this dual analysis, the system classifies the video as authentic or deepfake, displaying the result along with a confidence score.
The application supports common video formats such as MP4 and AVI and includes validation steps to ensure that input videos meet minimum quality standards, such as resolution and duration, for effective analysis. A progress bar or status indicator will provide real-time feedback during file upload and analysis. Once the process is completed, users receive a straightforward result showing the classification, confidence level, and, in future updates, visual cues or highlighted areas indicating signs of manipulation.
From a non-functional standpoint, the system must perform efficiently, processing standard videos (e.g., up to 1 minute in length) in less than 2 minutes on a GPU-enabled server. The user interface should be clean, responsive, and accessible across devices, including desktops and smartphones. Uploaded files

must be securely managed, with measures in place to protect user data and maintain privacy. The system should also be scalable, handling multiple users simultaneously without significant performance issues. Optional logging features may be enabled based on privacy regulations to record user submissions and results for auditing and continuous improvement.
For robustness and maintainability, the backend will include error-handling capabilities to detect and manage invalid inputs, unsupported video formats, or processing failures. The system architecture will be modular, enabling easy updates to the AI models and seamless integration of new features or datasets as deepfake generation techniques evolve.

3.3.1 External Interface Requirements

The deepfake detection system interfaces with various external components to deliver a smooth user experience, accurate processing, and efficient communication between its modules. At the forefront, the user interface is a browser-based front-end built using technologies like HTML, CSS, and JavaScript, with frameworks such as ReactJS for dynamic interaction. It enables users to upload video files, track the status of the analysis process, and view the final results. The interface is designed to be responsive and accessible on both desktop and mobile browsers, ensuring ease of use across different devices.
To bridge the front-end and back-end, the system employs RESTful APIs that manage the transmission of data including video uploads, feature extraction outputs, and classification results. These APIs communicate using secure HTTPS protocols and structured JSON format, ensuring data integrity and efficiency. Dedicated endpoints are responsible for handling specific operations like file uploads, processing progress checks, and result retrieval.
On the hardware side, the front-end demands only basic client devices with internet access and a browser. In contrast, the backend system requires more robust hardware featuring GPU support to efficiently perform deep learning model inference. The recommended setup includes an NVIDIA GPU with CUDA support, a multi-core CPU, and a minimum of 16 GB RAM for optimal performance.
In terms of software dependencies, the system integrates several external libraries and tools. OpenCV is used for breaking down videos into individual frames, while deep learning frameworks like PyTorch or TensorFlow are responsible for running the AI models. Backend logic is managed using Python-based web frameworks such as Flask or Django. Additionally, a database system like PostgreSQL or SQLite can be incorporated to store

logs, user uploads, and system activity records.
All external interfaces are carefully structured to maintain security, flexibility, and compatibility with industry-standard technologies. They enable cohesive interaction between different parts of the application and contribute to the system's reliability and scalability.

3.3.2 Performance Requirements

The deepfake detection system is engineered to provide fast and reliable results while maintaining a seamless user experience. It is optimized to analyse short to medium-duration videos (typically less than one minute) and produce classification results within 1 to 2 minutes when operating on a GPU-equipped server. For videos with higher resolutions or longer durations, the system should still process them efficiently, aiming to complete analysis within a maximum of 5 minutes, depending on hardware specifications.
To support multiple users simultaneously, the system is designed to manage concurrent video processing tasks without compromising performance or accuracy. Under standard operating conditions, it should efficiently serve at least 10 users at once. For larger-scale deployments, the architecture allows for scalability through horizontal scaling or integration with cloud platforms to ensure consistent responsiveness and throughput.
Moreover, the detection model is expected to deliver high accuracy— preferably exceeding 90% on established benchmark datasets such as Face Forensics++, Celeb-DF, and the Deepfake Detection Challenge dataset. The system should also achieve minimal false positive and false negative rates, reducing the chances of real videos being misclassified as fake or vice versa, thereby enhancing the reliability of the results.

3.3.3 Safety Requirements

Although the deepfake detection system does not directly interact with any physical components that could pose a safety risk, it must still comply with essential safety and security protocols to protect user data and ensure ethical usage. Uploaded video content should be managed securely, stored only temporarily if necessary, and automatically deleted after analysis to safeguard privacy and prevent unauthorized access or misuse.
Role-based access control must be implemented, particularly for administrative functions such as retraining models or managing datasets. To maintain the system's integrity, robust authentication and authorization mechanisms are necessary to prevent abuse, including the upload of malicious files. Input

validation should also be enforced to filter out corrupted or harmful content that could potentially disrupt system operations or expose vulnerabilities.
In the event of operational issues like failed uploads, model errors, or hardware failures, the system should respond gracefully—providing clear error messages to users while logging these events for administrator review. Mechanisms for recovery and continuity should be in place to ensure that ongoing processes or previous data are not lost or compromised.
Moreover, since incorrect classifications could lead to serious consequences such as reputational harm or misinformation, all analysis results should be accompanied by a confidence score. Users must also be informed through disclaimers that the detection results are based on AI predictions and should not be interpreted as conclusive proof.

3.3.4 Security Requirements

The deepfake detection system is required to implement strong security protocols to protect user data and ensure the reliability and confidentiality of system operations. All data exchanges between the client interface and the backend must be encrypted using HTTPS to safeguard against potential interception or tampering during transmission. Any video files temporarily stored for processing must reside in secure, access-controlled locations and should be automatically removed from storage once the analysis is complete to uphold user privacy.
Secure authentication methods must be in place for all administrative users, including the use of hashed passwords and secure session management techniques such as token-based authentication. If the platform is expanded to include user registration or logging of activity history, it must comply with applicable data protection standards like the GDPR or other regional privacy regulations to safeguard personally identifiable information.
To maintain application security, the system should rigorously validate and sanitize all incoming data to protect against injection-based threats, including SQL injection, command injection, and cross-site scripting (XSS). Additionally, safeguards must be in place to defend against denial-of-service (DoS) attacks by enforcing limits on request frequency and restricting upload file sizes.

3.3.5 Design Constraints

The design and deployment of the deepfake detection system are influenced by several key constraints that shape its overall structure and functionality. One significant limitation is the reliance on specific deep learning li-

braries such as PyTorch and OpenCV. These tools define the development environment and introduce dependencies that must be carefully managed. Additionally, the selected architecture which incorporates a ResNeXt CNN for extracting spatial features and an LSTM network for analysing temporal dynamics—demands substantial computational power, particularly GPU acceleration. As a result, the system is not well-suited for deployment on low-powered or edge devices and is better optimized for cloud-based or high-performance computing setups.

Real-time responsiveness is another critical constraint. To maintain quick processing and deliver prompt results to users, restrictions must be applied to video length, resolution, and format. Supported formats are limited to standards like `.mp4`, and input size limits are enforced to maintain processing efficiency and ensure system reliability.

Furthermore, the system's effectiveness is closely tied to the datasets used during training such as Face Forensics++, Celeb-DF, and the Deepfake Detection Challenge (DFDC). The model's accuracy may decrease when analysing content that significantly differs from these datasets unless it undergoes further fine-tuning or retraining with more diverse data sources.

3.3.6 Software Quality Attributes

The deepfake detection system is designed to deliver core functionalities by leveraging AI models to accurately differentiate between genuine and manipulated videos. It ensures consistent and dependable performance, reinforcing its reliability across different usage scenarios. The application features a web-based interface that prioritizes ease of use, making it accessible to users regardless of their technical background. High processing efficiency allows for quick analysis and minimal wait times, contributing to strong performance. Security is a key focus, with measures like encrypted data transmission, strict input validation, and safe handling of uploaded content. The system's modular and clearly documented codebase supports easy maintenance and future updates. Additionally, its portable architecture enables smooth deployment across different platforms with minimal configuration.

Chapter 4

SYSTEM DESIGN

4.1 Architectural Design

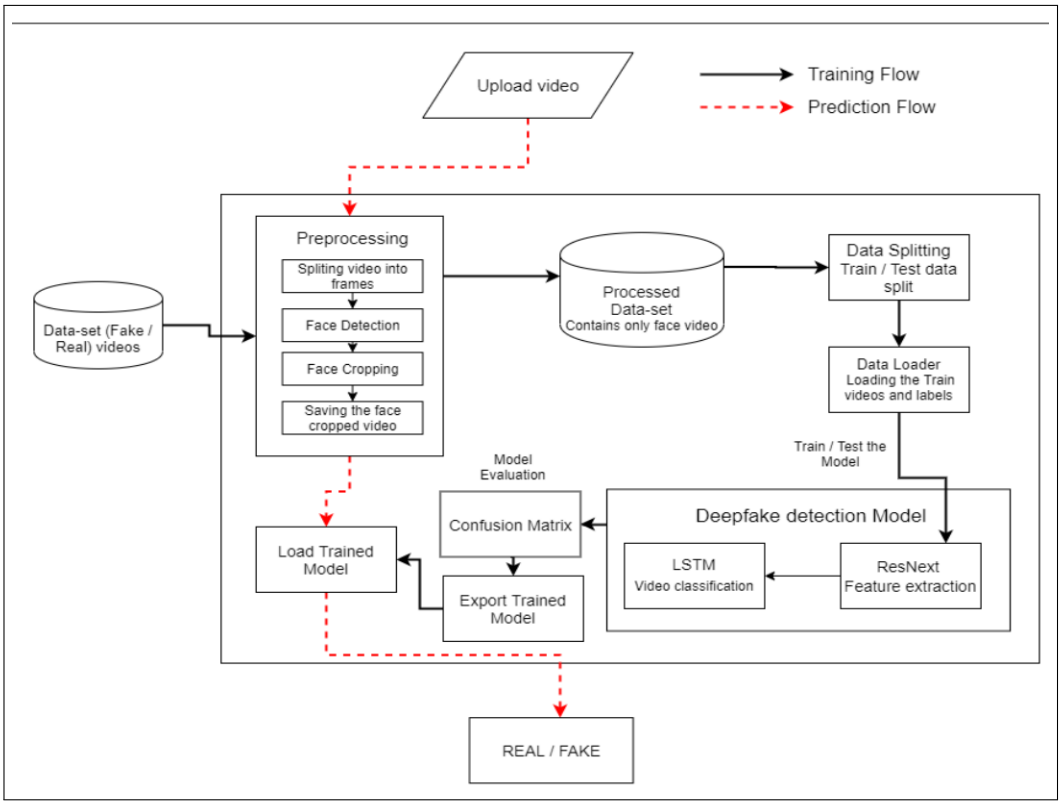


Figure 4.1: Architecture diagram

4.1.1 Class Diagram

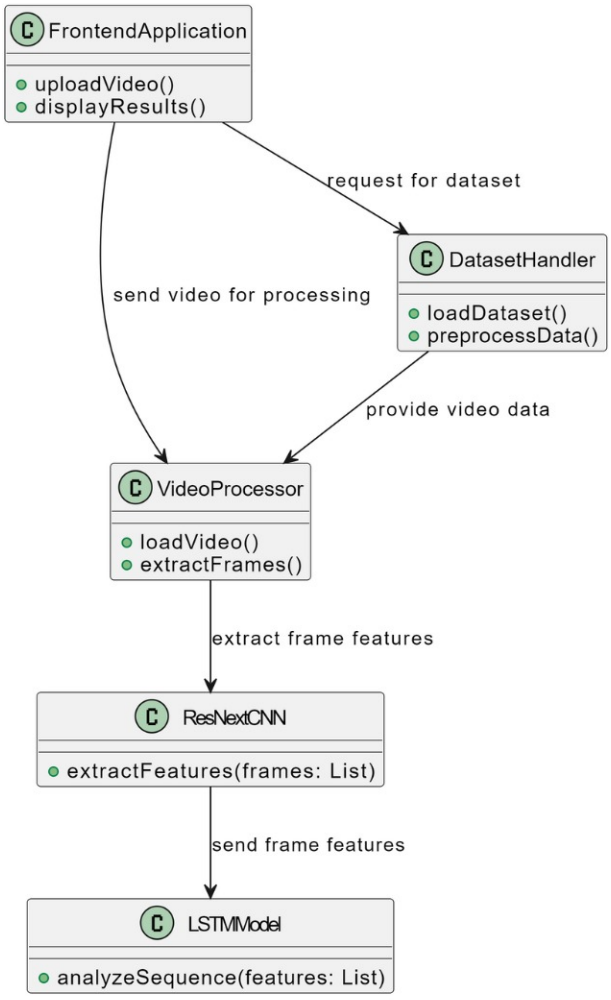


Figure 4.2: Class Diagram

4.1.2 Use Case Diagram

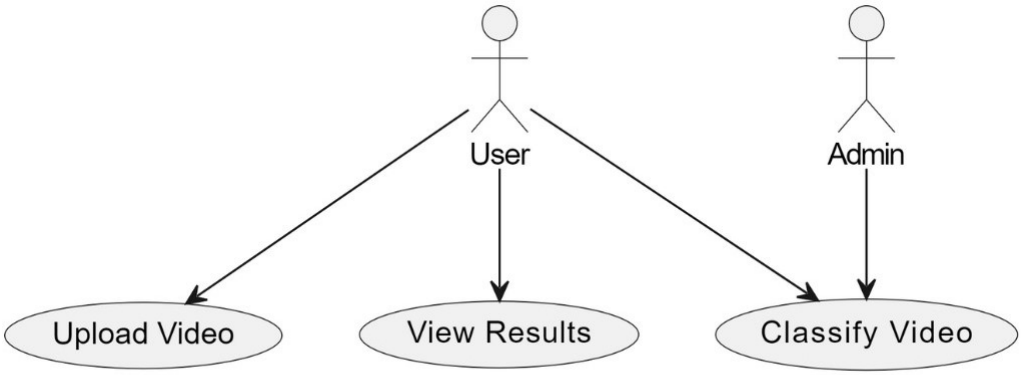


Figure 4.3: Use Case Diagram

4.1.3 Sequence Diagram

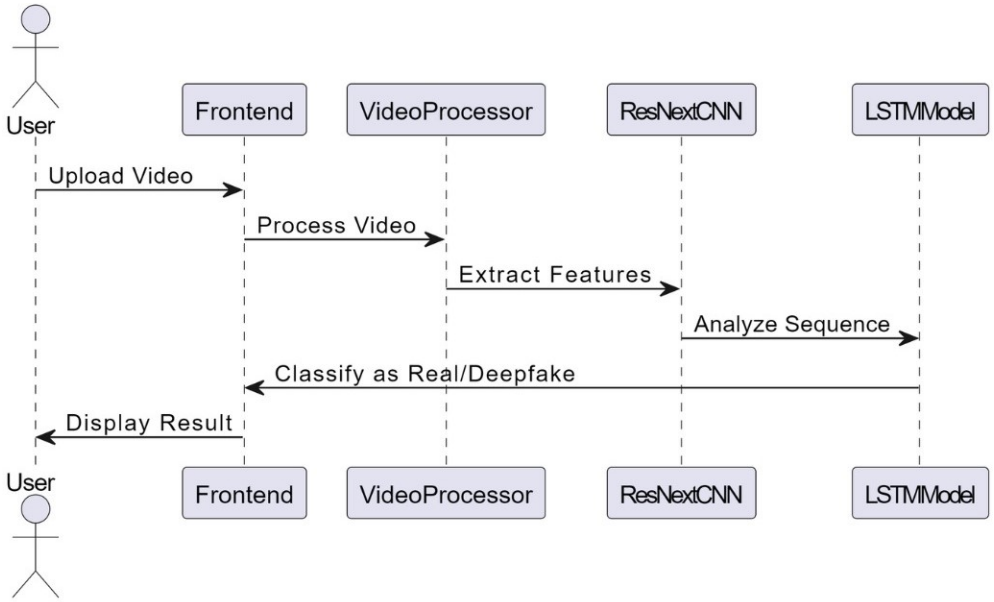


Figure 4.4: Sequence Diagram

4.1.4 Activity Diagram

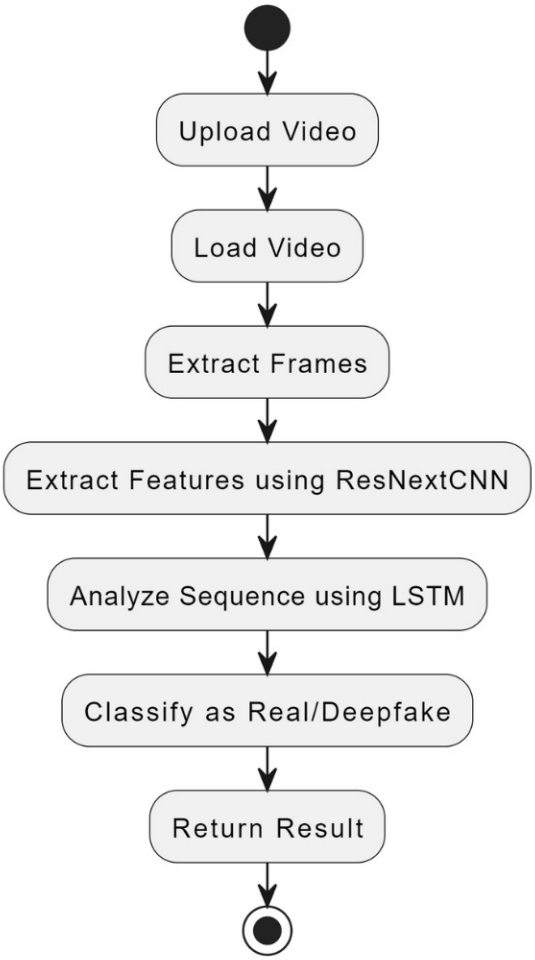


Figure 4.5: Activity Diagram

4.1.5 Component Diagram

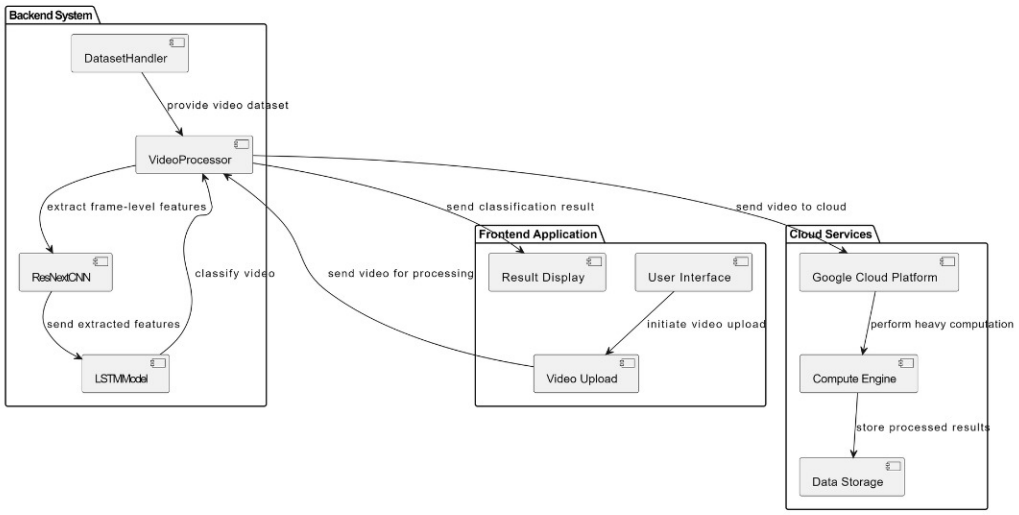


Figure 4.6: Component Diagram

4.1.6 Data Flow Diagram

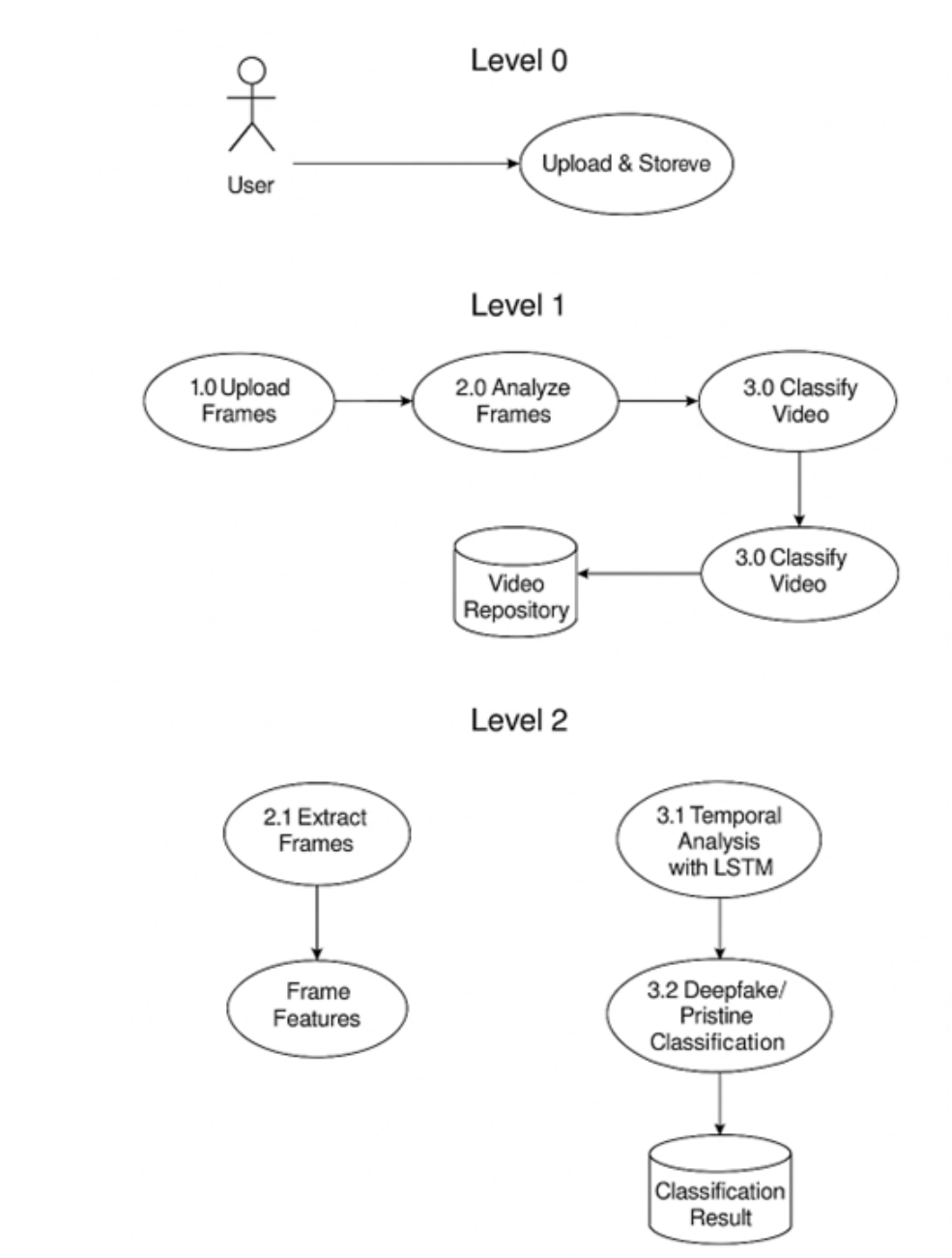


Figure 4.7: Data Flow Diagram

Chapter 5

PROJECT PLANNING

5.1 Project Estimate

There are many examples where deepfake creation technology is used to mislead the people on social media platforms by sharing the false deepfake videos of the famous personalities like Mark Zuckerberg Eve of House A.I. Hearing, Donald Trump's Breaking Bad series where he was introduced as James McGill, Barack Obama's public service announcement and many more [**?**]. These types of deepfakes create 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 [**?**] 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 indistinguishable deepfakes.
Although some malignant deepfake videos exist, but till now they remain a minority. So far, the released tools [**?**, **?**] that generate deepfake videos are being extensively used to create fake celebrity pornographic videos or revenge porn [**?**]. Some of the examples are Brad Pitt, Angelina Jolie nude

videos. The real-looking nature of the deepfake videos makes the celebrities and other famous 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 [**?**]. Due to which it becomes very important to detect the deepfake videos and avoid the percolation of the deepfakes on the social media platforms.

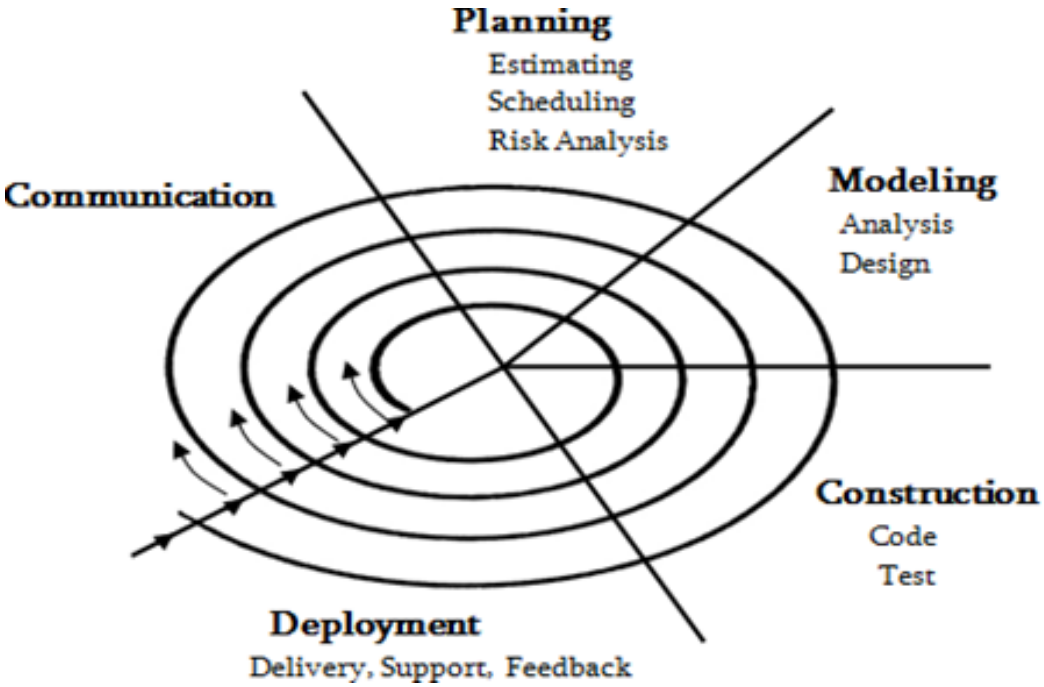


Figure 5.1: Spiral Methodology SDLC

5.1.1 Reconciled Estimates

- **Cost Estimate:** Rs 11,600

- **Time Estimates:** 12 Months (refer Appendix B)

5.1.2 Cost Estimation using COCOMO (Constructive Cost) Model

Since we have a small team, less-rigid requirements, long deadline we are using the organic COCOMO [**?**] model.

Cost (in Rs)	Description
5260	Pre-processing the dataset on GCP
2578	Training models on GCP
761	Google Colab Pro subscription
3000	Deploying project to GCP using Cloud engine

Table 5.1: Cost Estimation

1. **Efforts Applied:** It defines the amount of labor that will be required to complete a task. It is measured in person-months units.

$$EffortApplied(E) = a_b \times (KLOC)^{b_b}$$

$$E = 2.4 \times (20.5)^{1.05} = 57.2206PM$$

2. **Development Time:** Simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put. It is measured in the units of time such as weeks, months.

$$DevelopmentTime(D) = c_b \times (E)^{d_b}$$

$$D = 2.5 \times (57.2206)^{0.38} = 11.6M$$

3. **People Required:** The number of developers needed to complete the project.

$$PeopleRequired(P) = \frac{E}{D} = \frac{57.2206}{11.6} = 4.93$$

5.1.3 Risk Identification

Before the training, we need to prepare thousands of images for both persons. We can take a shortcut and use a face detection library to scrape facial pictures from their videos. Spend significant time to improve the quality of your facial pictures. It impacts your final result significantly.

- Remove any picture frames that contain more than one person.
- Make sure you have an abundance of video footage. Extract facial pictures containing different pose, face angle, and facial expressions.
- Some resemblance of both persons may help, like similar face shape.

ID	Risk Description	Probability	Impact (Schedule)	Impact (Quality)
1	Does it over blur compared to other non-facial areas of the video?	Low	High	High
2	Does it flick?	High	High	High
3	Is there a change of skin tone near the edge of the face?	Low	High	Low
4	Are there double chin, double eyebrows, or double edges on the face?	High	High	Low
5	When the face is partially blocked (e.g., by hands), does it flick or blur?	High	High	High

Table 5.2: Risk Description

Probability	Value Description
High	Probability of occurrence is greater than 75%
Medium	Probability of occurrence is between 26% and 75%
Low	Probability of occurrence is less than 25%

Table 5.3: Risk Probability Definitions

5.1.4 Risk Analysis

In Deepfakes, it creates a mask on the created face so it can blend in with the target video. To further eliminate the artifacts:

- Apply a Gaussian filter to further diffuse the mask boundary area.
- Configure the application to expand or contract the mask further.
- Control the shape of the mask.

5.1.5 Overview of Risk Mitigation, Monitoring, and Management

Risk Mitigation:
Risk mitigation refers to the proactive strategies used to reduce or eliminate the likelihood and impact of potential risks. In deepfake projects, this includes:

- **Improving Data Quality:** Ensuring high-resolution, clear facial images from varied angles and expressions to prevent poor model performance.
- **Avoiding Multi-Face Frames:** Filtering out images with more than one face to prevent data confusion and inaccurate results.
- **Balanced Dataset:** Using similar facial structures and consistent lighting conditions for both subjects to help the model generalize better.
- **Technical Enhancements:** Applying techniques like Gaussian blur and mask shaping to reduce visible artifacts and improve face blending.

Risk Monitoring:
Risk monitoring is the continuous observation of potential threats and issues during the project lifecycle. This includes:

- Tracking Model Behaviour: Observing training outputs to identify anomalies such as model bias, overfitting, or blending errors.
- Performance Metrics: Regularly measuring quality indicators (e.g., identity loss, reconstruction error) to ensure the model is learning correctly.
- Automated Alerts: Setting up systems to flag unexpected changes or performance drops during training or deployment.
- Data Updates: Keeping the dataset under review to ensure it remains consistent and relevant as the project progresses.

Risk Management:
Risk management is the broader process of identifying, assessing, prioritizing, and addressing risks throughout the project. It includes:

- Risk Assessment: Listing potential risks—technical, ethical, or operational—and evaluating their impact and probability.
- Prioritization: Classifying risks based on severity and urgency to determine which need immediate action.
- Contingency Planning: Developing backup plans, such as retraining with different data or switching to alternate models if issues arise.
- Documentation & Review: Maintaining a detailed log of all identified risks, actions taken, and results, which supports future audits and improvements.

Chapter 6

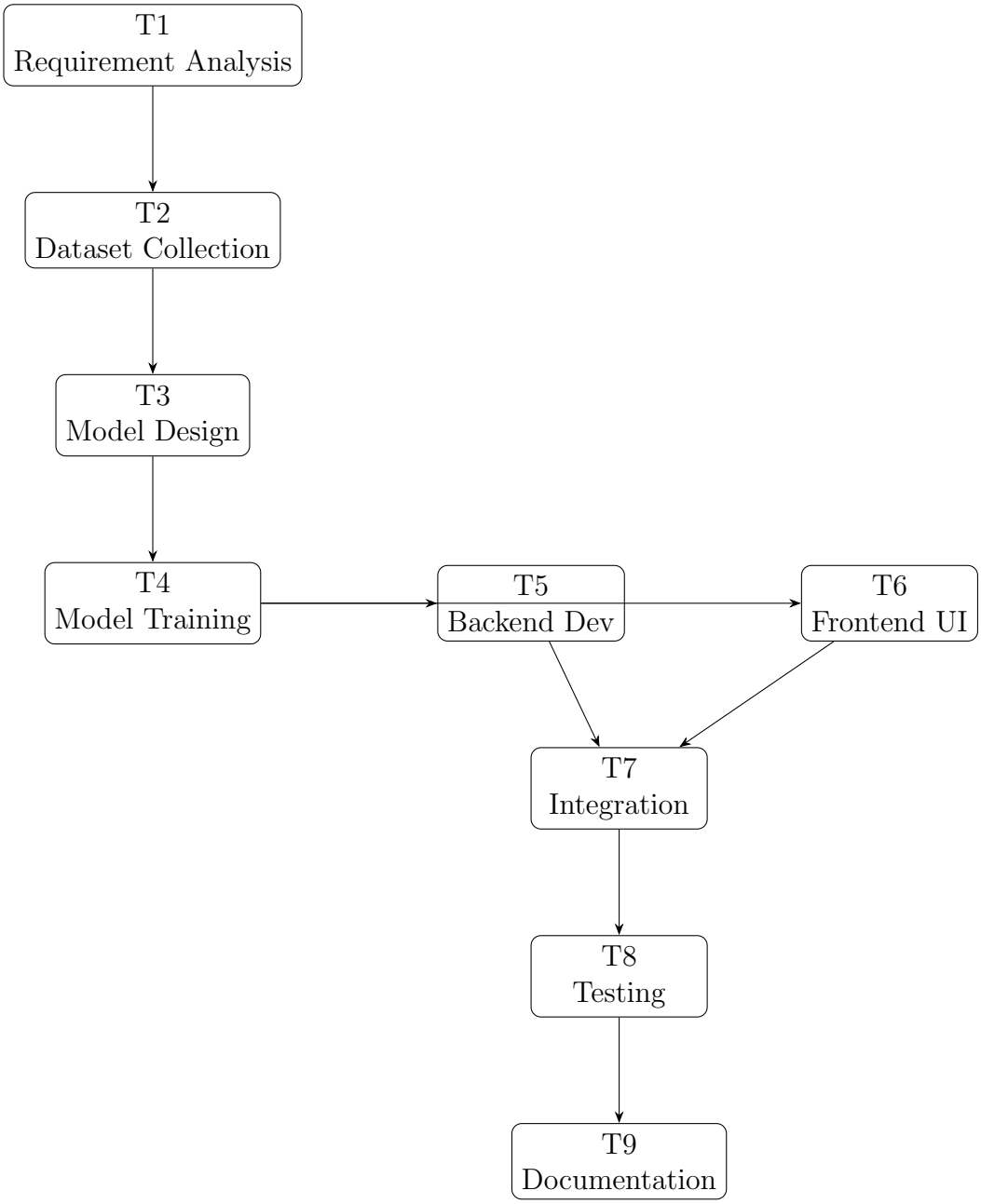
PROJECT SCHEDULE

6.1 Project Task Set

ID	Task	Description	Deliverable
T1	Requirement Analysis	Define goals, study papers, tools to be used	Requirement Document
T2	Dataset Collection & Preprocessing	Collect videos, extract frames, normalize data	Preprocessed Dataset
T3	Model Design	Design CNN + LSTM model	Model Architecture Diagram
T4	Model Training & Validation	Train and validate model using evaluation metrics	Trained Model, Metrics
T5	Backend Development	Create Flask/Django API for predictions	Backend Script
T6	Frontend & UI Design	UI for upload and output display (HTML/CSS/JS)	Web Interface
T7	Integration & Deployment	Connect frontend-backend, deploy project	Working Web App
T8	Testing	Unit, integration, and UI testing	Test Scripts, Reports
T9	Documentation & Demo	Report, presentation, and demo video	Report, PPT, Demo

Table 6.1: Project Task Set

6.2 Task Network



6.3 Timeline Chart

Week	T1	T2	T3	T4	T5	T6	T7	T8	T9
Week 1	✓								
Week 2		✓							
Week 3			✓						
Week 4				✓					
Week 5					✓				
Week 6						✓			
Week 7							✓	✓	
Week 8									✓

Table 6.2: Project Timeline Chart

6.4 Gantt Chart

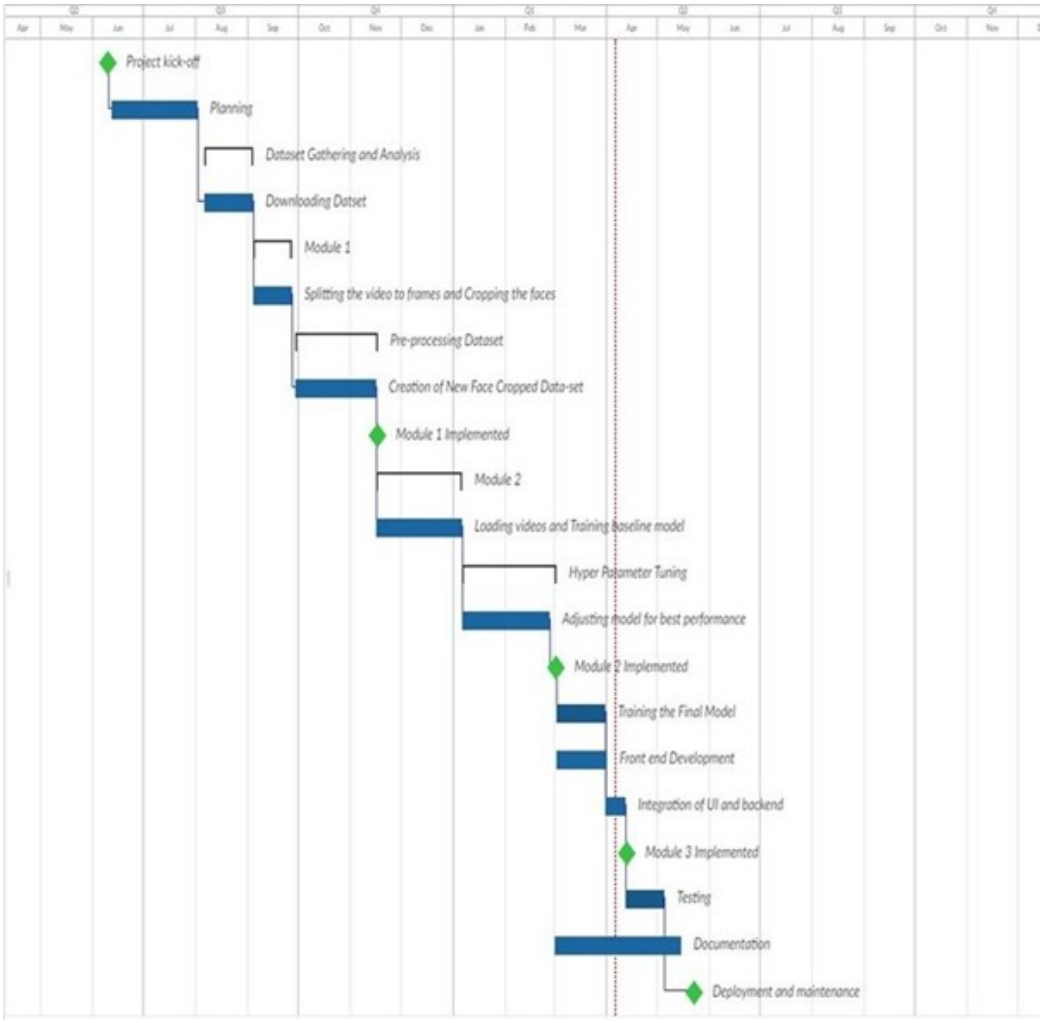


Figure 6.1: Fig 6.4.1 :- Complete Project Plan

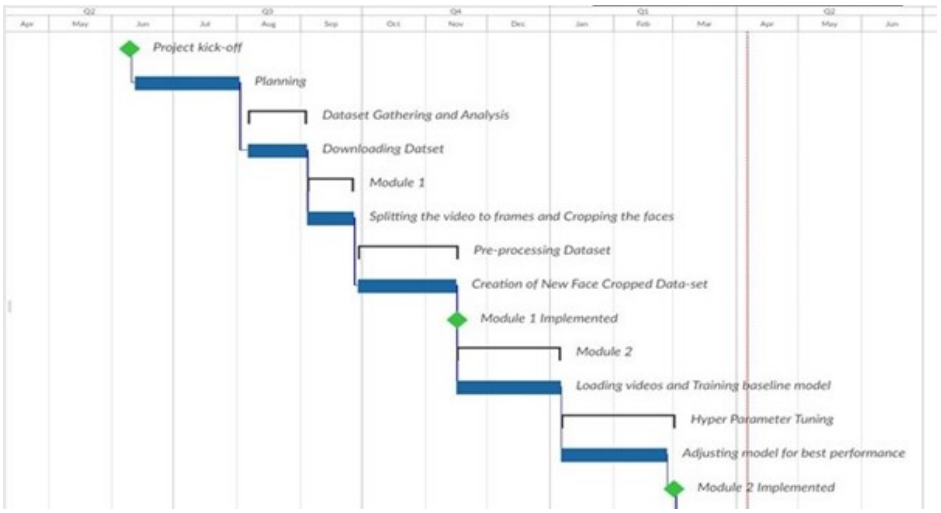


Figure 6.2: Fig 6.4.2 :- Project Plan 1

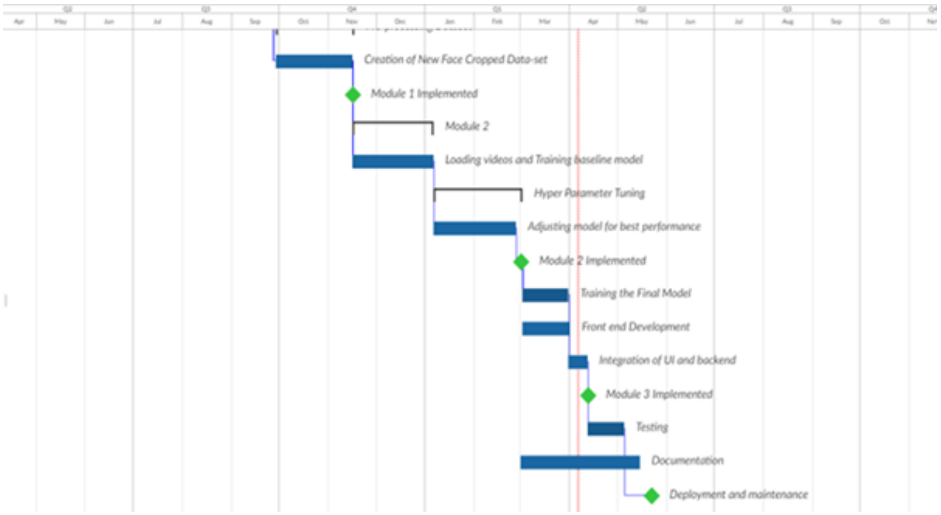


Figure 6.3: Fig 6.4.3 :- Project Plan 2

Chapter 7

ALGORITHM ANALYSIS AND MATHEMATICAL MODELING

7.1 Algorithm Details

7.1.1 Algorithm with Pseudo Code

Algorithm ExtractFramesAndPreprocess(*VideoPath*)
Input: VideoPath - path to the uploaded video
Output: List of preprocessed frames

1. Open video file using OpenCV
2. Initialize empty FrameList
3. While video has more frames:
    - Read frame
    - Resize frame to (224, 224)
    - Normalize pixel values
    - Append to FrameList
4. End While
5. Return FrameList

End Algorithm

7.1.2 GAPS

- **Dataset Diversity:** Performance drops on unseen datasets with different compression or lighting.
- **Real-Time Detection:** Processing video frame-by-frame is slow for real-time systems.
- **Model Interpretability:** The CNN+LSTM model is a black box; difficult to explain predictions.
- **Scalability:** Difficult to scale detection for large video files on low-end hardware.
- **Adaptive Deepfakes:** Newer GANs may bypass traditional detection methods.

This project focuses on detecting deepfake videos using a hybrid neural network approach. It involves:

- Extracting frames from the uploaded video.
- Preprocessing frames using OpenCV and resizing them.
- Feeding the sequence of frames to a CNN-LSTM model.
- Outputting the classification as "Real" or "Fake".
- Implementing a Django-based web app where users can upload videos and see results.

7.1.3 Mathematical Model

Let's represent the system using a Mathematical Model (5-tuple):

$$S = \{I, P, F, O, DD\}$$

Symbol	Description
I	Input: Video file uploaded by user
P	Preprocessing: Frame extraction, resizing, normalization
F	Functions: Neural Network functions (CNN + LSTM)
O	Output: Label = {Real, Fake}
DD	Deterministic Data: Pre-trained model weights, thresholds, image size

Table 7.1: Mathematical Model Description

Step-by-step Model:

- $I = video.mp4$
- $P = \{x_i = Resize(Normalize(Frame_i))\}$
- $F = Model(x_1, x_2, ..., x_n)$
- $O = \{ \text{"}Fake\text{"}, if y \geq \theta$
  $\text{"}Real\text{"}, if y < \theta$

Chapter 8

CODE IMPLEMENTATION

8.1 Django Code Implementation

views.py

```
from django.shortcuts import render
from django.http import JsonResponse
from .deepfake_detector import detect_deepfake

def home(request):
    return render(request, 'index.html')

def upload_video(request):
    if request.method == 'POST' and request.FILES['video']:
        video = request.FILES['video']
        result = detect_deepfake(video)
        return JsonResponse({'result': result})
    return JsonResponse({'error': 'Invalid Request'})
```

urls.py

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home, name='home'),
    path('upload/', views.upload_video, name='upload_video'),
]
```

deepfake$_d$etector.py

```
import cv2
import numpy as np
from keras.models import load_model

model = load_model('deepfake_cnn_lstm_model.h5')

def detect_deepfake(video_file):
    frames = extract_and_preprocess(video_file)
    prediction = model.predict(np.array([frames]))
    label = 'Fake' if prediction >= 0.5 else 'Real'
    return label

def extract_and_preprocess(video_file):
    cap = cv2.VideoCapture(video_file.temporary_file_path())
    frames = []
    while True:
        ret, frame = cap.read()
        if not ret:
            break
        resized = cv2.resize(frame, (224, 224))
        normalized = resized / 255.0
        frames.append(normalized)
        if len(frames) >= 30:
            break
    cap.release()
    return frames
```

index.html (Template)

```
<!DOCTYPE html>
<html>
<head>
    <title>Deepfake Detection</title>
</head>
<body>
    <h1>Upload Video to Detect Deepfake</h1>
    <form action="/upload/" method="post" enctype="multipart/form-data">
        {% csrf_token %}
        <input type="file" name="video">
```

```
        <input type="submit" value="Upload">
    </form>
</body>
</html>
```

Chapter 9

SOFTWARE TESTING

9.1 Manual Testing

Manual testing plays a crucial role in validating the accuracy, usability, and robustness of the system. Unlike automated testing, manual testing allows testers to simulate real-world scenarios and evaluate system behavior based on human judgment and interaction. In this project, manual testing was carried out to assess the system's ability to detect deepfake videos, ensure proper handling of video upload functionality, validate size constraints, and handle various user inputs (correct and incorrect). These tests are essential to ensure the end-user has a smooth experience and that the system does not behave unpredictably when exposed to unforeseen situations.
The key objectives of manual testing included:

- Verifying that only valid video formats (e.g., .mp4) are accepted.
- Ensuring the system rejects oversized video files that exceed the defined limit.
- Checking if the system accurately detects real and fake videos.
- Testing user actions such as submitting forms without selecting a file.
- Validating URL redirection behavior in cases of manual endpoint access.
- Identifying issues related to multiple faces, cropped faces, or no faces in video inputs.

The following table outlines 20 key test cases performed during the manual testing phase:

Case ID	Test Case Description	Test Data	Expected Result	Actual Result	Status
1	Upload a word file instead of video	.docx file	Error: Only video files allowed	Error: Only video files allowed	Pass
2	Upload a word file, but system accepts it	.docx file	Error: Only video files allowed	File uploaded successfully	Fail
3	Upload a 200MB video file	.mp4 ¿ 100MB	Error: Max limit 100MB	Error: Max limit 100MB	Pass
4	Large file bypasses size check	.mp4 ¿ 100MB	Error: Max limit 100MB	File uploaded successfully	Fail
5	Upload file without any faces	No faces	Error: No faces detected	Error: No faces detected	Pass
6	File with no faces is processed	No faces	Error: No faces detected	Fake/Real label returned	Fail
7	Videos with many faces	Group video	Fake/Real classification	Fake	Pass
8	System fails on multiple faces	Group video	Fake/Real classification	Error/No result	Fail
9	Deepfake video	Known deepfake	Fake	Fake	Pass
10	Deepfake misclassified	Known deepfake	Fake	Real	Fail
11	Enter /predict URL manually	/predict in browser	Redirect to /upload	Redirect to /upload	Pass
12	Wrong redirection for /predict	/predict in browser	Redirect to /upload	404 or wrong page	Fail
13	Upload clicked without file	No file selected	Alert: Please select video	Alert: Please select video	Pass
14	No file but processed	No file selected	Alert: Please select video	System crashes or uploads	Fail
15	Upload a real video	Real video	Real	Real	Pass
16	Real video misclassified	Real video	Real	Fake	Fail
17	Cropped real video	Cropped face (real)	Real	Real	Pass
18	Cropped real video misclassified	Cropped face (real)	Real	Fake	Fail
19	Cropped fake video	Cropped face (deepfake)	Fake	Fake	Pass
20	Cropped fake misclassified	Cropped face (deepfake)	Fake	Real	Fail

Table 9.1: Manual Test Cases and Results

9.2 Automation Testing

9.2.1 Unit Testing

```
import unittest
from ml_app import views

class UnitTestModelFunctions(unittest.TestCase):
    def test_model_load(self):
        model = views.load_model()
        self.assertIsNotNone(model)

    def test_preprocess_video(self):
        test_video_path = "test_videos/sample.mp4"
        try:
            output = views.preprocess_video(test_video_path)
            self.assertTrue(isinstance(output, list))
        except FileNotFoundError:
            self.skipTest("Test video not available")

if __name__ == '__main__':
    unittest.main()
```

9.2.2 Integration Testing

```
import unittest
from ml_app.views import predict_video

class IntegrationTestPipeline(unittest.TestCase):
    def test_predict_video_pipeline(self):
        test_video_path = "test_videos/sample.mp4"
        try:
            result = predict_video(test_video_path)
            self.assertIn(result, ["real", "fake"])
        except FileNotFoundError:
            self.skipTest("Test video not found")

if __name__ == '__main__':
    unittest.main()
```

9.2.3 System Testing

```
from django.test import TestCase, Client
from django.core.files.uploadedfile import SimpleUploadedFile
import os

class SystemTest(TestCase):
    def setUp(self):
        self.client = Client()

    def test_full_prediction_flow(self):
        video_path = "test_videos/sample.mp4"
        if not os.path.exists(video_path):
            self.skipTest("Sample test video not available")
        with open(video_path, 'rb') as video:
            response = self.client.post('/predict/', {
                'video_file': SimpleUploadedFile("sample.mp4", video.read())})
        self.assertEqual(response.status_code, 200)
        self.assertIn(b'Prediction Result', response.content)
```

9.2.4 White Box Testing

```
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
import unittest

class UITest(unittest.TestCase):
    def setUp(self):
        options = Options()
        options.headless = True
        self.driver = webdriver.Chrome(options=options)
        self.driver.get("http://localhost:8000")

    def test_home_page_title(self):
        self.assertIn("DeepFake", self.driver.title)

    def test_upload_button_present(self):
        upload = self.driver.find_element("id", "videoUpload")
        self.assertIsNotNone(upload)

    def tearDown(self):
```

```
        self.driver.quit()

if __name__ == '__main__':
    unittest.main()
```

Chapter 10

RESULT

10.1 Screenshots of All Pages

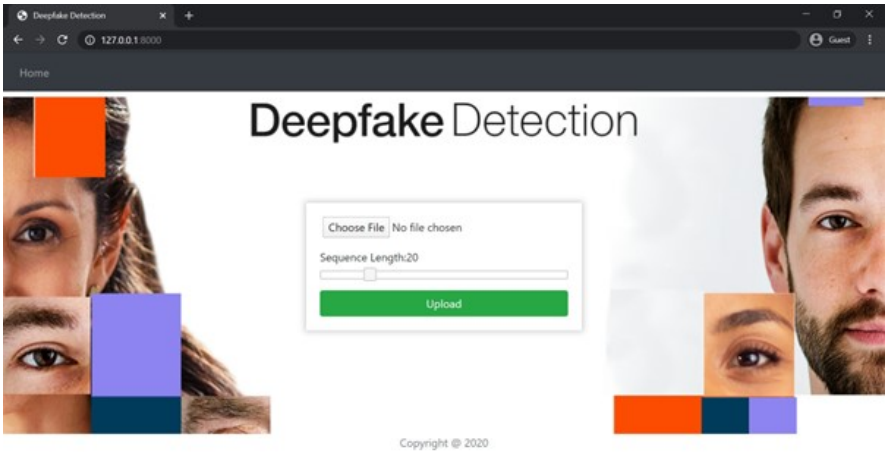


Figure 10.1: Home Page

Figure 10.2: Uploading Real Video

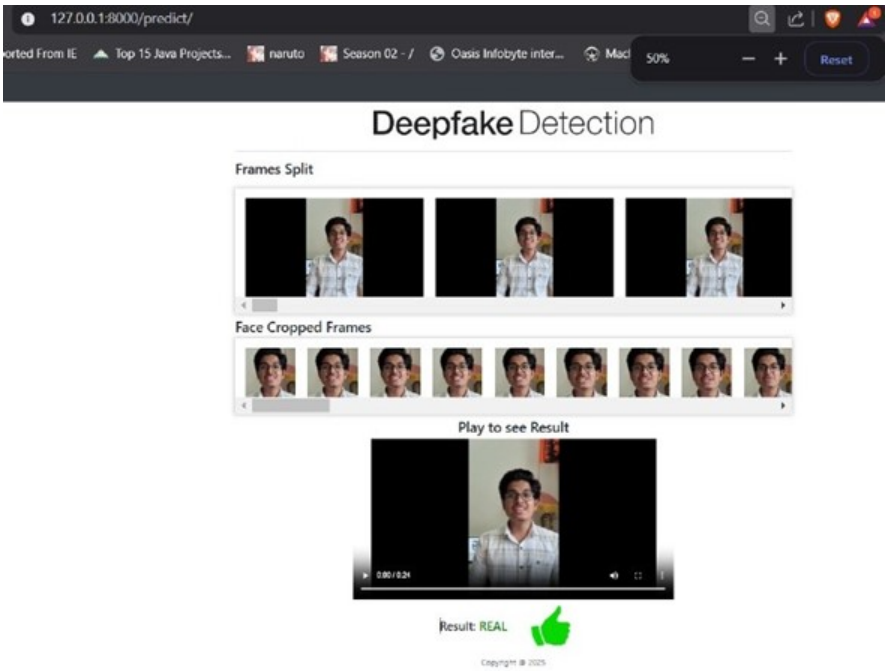


Figure 10.3: Real Video Output

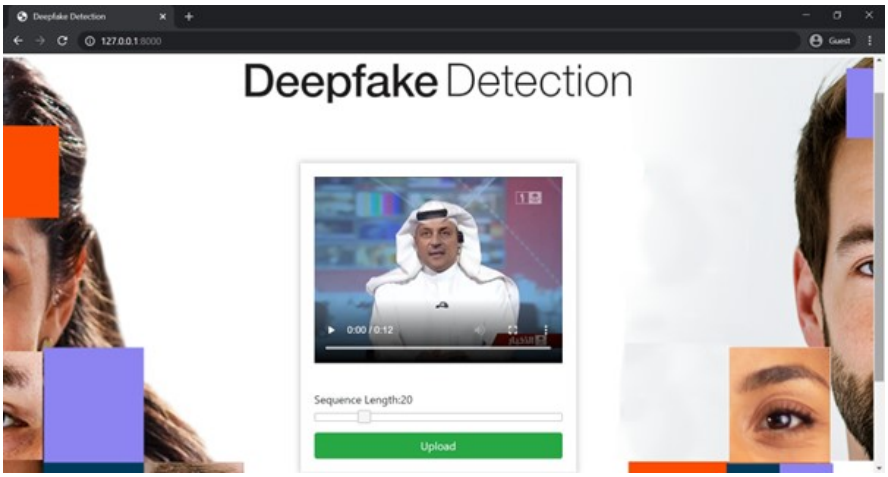


Figure 10.4: Uploading Fake Video

Figure 10.5: Fake Video Output

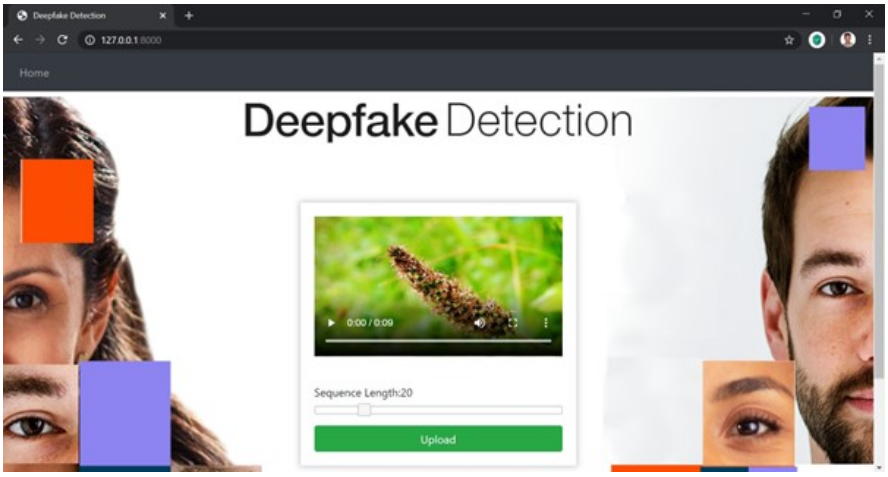


Figure 10.6: Uploading Video with No Faces

Figure 10.7: Output of Video with No Faces

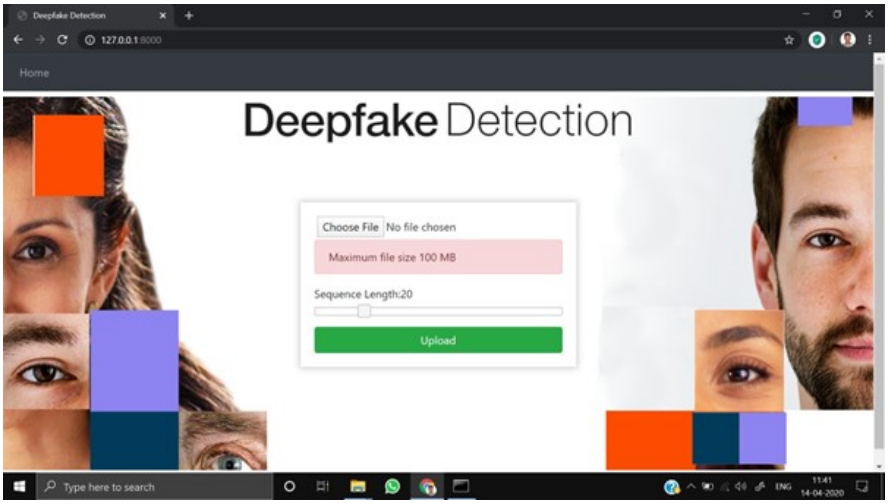


Figure 10.8: Uploading File Greater Than 100MB

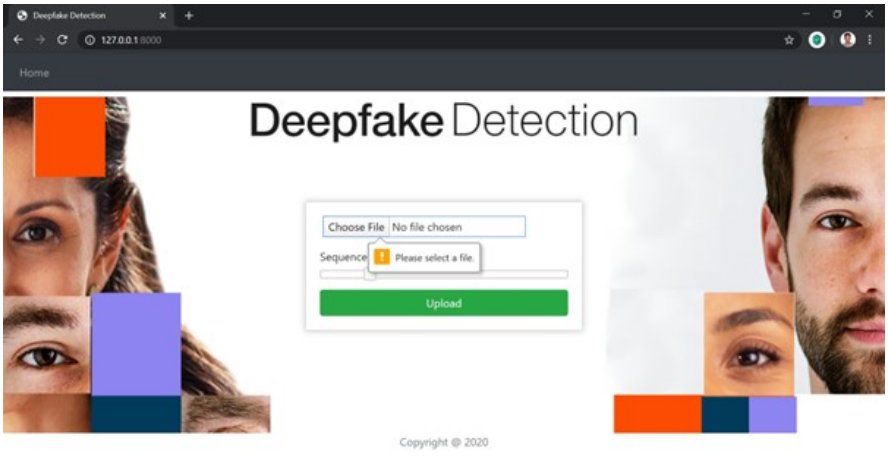


Figure 10.9: Pressing Upload Button Without Selecting Video

10.2 Result

In this study, multiple deepfake detection models were evaluated using different datasets and sequence lengths, showing significant improvements in accuracy compared to previous research.
Using the FaceForensics++ dataset, models trained with sequence lengths of 20, 40, 60, 80, and 100 frames achieved accuracies of 90.95%, 95.23%, 97.49%, 97.73%, and 97.76% respectively. These results indicate a clear trend: increasing the number of frames per sequence leads to improved model performance, with the highest accuracy observed at 100 frames.
A model trained on a combined dataset of Celeb-DF and FaceForensics++ with 100-frame sequences achieved an accuracy of 93.98%, demonstrating that combining datasets contributes to enhanced generalization and reliability.
On a separate custom dataset, models trained with 10, 20, and 40-frame sequences reached accuracies of 84.21%, 87.79%, and 89.35% respectively. While these figures are slightly lower than those achieved using FaceForensics++, they still surpass typical accuracies reported in earlier works, which often range between 75% and 85%.
Overall, the models presented here outperform prior published results, with the best-performing model reaching 97.76% accuracy, emphasizing the importance of sequence length and dataset quality in deepfake detection tasks.

Chapter 11

CONCLUSION

We presented a neural network-based approach to classify a video as deepfake or real, along with the model's confidence score. Our method is capable of predicting the output by processing just 1 second of video (10 frames per second) with good accuracy.
The model implementation involves using a pre-trained ResNeXt CNN to extract frame-level features, and an LSTM network to perform temporal sequence processing. This setup captures differences between consecutive frames (t and t-1) effectively.
The model can process videos in frame sequences of 10, 20, 40, 60, 80, and 100, enabling flexibility for real-time or batch processing use cases. Overall, the system shows promising results in detecting deepfake videos accurately and can be enhanced further with more robust datasets and training refinements.

Chapter 12

REFERENCES

1. Andreas Rossler et al., "FaceForensics++: Learning to Detect Manipulated Facial Images," arXiv:1901.08971.
2. Deepfake Detection Challenge Dataset: `https://www.kaggle.com/c/deepfake-detection-challenge/data` (Accessed: March 26, 2020)
3. Yuezun Li et al., "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics," arXiv:1909.12962
4. Deepfake Video of Mark Zuckerberg Goes Viral: `https://fortune.com/2019/06/12/deepfake-mark-zuckerberg/`
5. 10 Deepfake Examples: `https://www.creativebloq.com/features/deepfake-examples`
6. TensorFlow Framework: `https://www.tensorflow.org/`
7. Keras Library: `https://keras.io/`
8. PyTorch Framework: `https://pytorch.org/`
9. G. Antipov et al., "Face Aging with Conditional GANs," arXiv:1702.01983
10. J. Thies et al., "Face2Face: Real-time Face Capture and Reenactment," CVPR, 2016
11. FaceApp: `https://www.faceapp.com/`
12. FaceSwap Online: `https://faceswaponline.com/`
13. Deepfakes Impact on Women: `https://www.forbes.com/sites/chenxiwang/2019/11/01/deepfakes-revenge-porn-and-the-impact-on-women/`

14. Deepfakes Democracy: https://www.theguardian.com/technology/ng-interactive/2019/jun/22/the-rise-of-the-deepfake-and-the-threat-to-democ

15. Yuezun Li, Siwei Lyu, "Exposing DF Videos by Detecting Face Warping Artifacts," arXiv:1811.00656v3

16. Yuezun Li et al., "Detecting Eye Blinking in Deepfake Videos," arXiv:1806.02877v2

17. Huy H. Nguyen et al., "Using Capsule Networks to Detect Forged Videos," arXiv:1810.11215

18. D. Guera and E.J. Delp, "Deepfake Detection Using RNNs," IEEE AVSS 2018

19. I. Laptev et al., "Learning Human Actions from Movies," CVPR 2008

20. Umur Ciftci et al., "Detection Using Biological Signals," arXiv:1901.02212v2

21. D. P. Kingma and J. Ba, "Adam Optimizer," arXiv:1412.6980

22. ResNeXt Model: https://pytorch.org/hub/pytorch_vision_resnext/

23. COCOMO Model: https://www.geeksforgeeks.org/software-engineering-cocomo-mo

24. Deepfake Video Detection using Neural Networks: http://www.ijsrd.com/articles/IJSRDV8I10860.pdf

25. IJSRD Journal: http://ijsrd.com/

Annexure A

Base Paper

Deepfake Video Detection Using Recurrent Neural Networks

David Güera Edward J. Delp

Video and Image Processing Laboratory (VIPER), Purdue University

Abstract

In recent months a machine learning based free software tool has made it easy to create believable face swaps in videos that leaves few traces of manipulation, in what are known as "deepfake" videos. Scenarios where these realistic fake videos are used to create political distress, blackmail someone or fake terrorism events are easily envisioned. This paper proposes a temporal-aware pipeline to automatically detect deepfake videos. Our system uses a convolutional neural network (CNN) to extract frame-level features. These features are then used to train a recurrent neural network (RNN) that learns to classify if a video has been subject to manipulation or not. We evaluate our method against a large set of deepfake videos collected from multiple video websites. We show how our system can achieve competitive results in this task while using a simple architecture.

Figure 1. Face swapping is not new. Examples such as the swap of U.S. President Lincoln's head with politician John Calhoun's body were produced in mid-19th century (left). Modern tools like FakeApp [2] have made it easy for anyone to produce "deepfakes", such as the one swapping the heads of late-night TV hosts Jimmy Fallon and John Oliver (right).

1. Introduction

The first known attempt at trying to swap someone's face, circa 1865, can be found in one of the iconic portraits of U.S. President Abraham Lincoln. The lithography, as seen in Figure 1, mixes Lincoln's head with the body of Southern politician John Calhoun. After Lincoln's assassination, demand for lithographies of him was so great that engravings of his head on other bodies appeared almost overnight [27].

Recent advances [21, 42] have radically changed the playing field of image and video manipulation. The democratization of modern tools such as Tensorflow [6] or Keras [12] coupled with the open accessibility of the recent technical literature and cheap access to compute infrastructure have propelled this paradigm shift. Convolutional autoencoders [38, 37] and generative adversarial network (GAN) [17, 7] models have made tampering images and videos, which used to be reserved to highly-trained professionals, a broadly accessible operation within reach of almost any individual with a computer. Smartphone and desktop applications like FaceApp [1] and FakeApp [2] are built upon this progress.

FaceApp automatically generates highly realistic transformations of faces in photographs. It allows one to change face hair style, gender, age and other attributes using a smartphone. FakeApp is a desktop application that allows one to create what are now known as "deepfakes" videos. Deepfake videos are manipulated videoclips which were first created by a Reddit user, deepfake, who used TensorFlow, image search engines, social media websites and public video footage to insert someone else's face onto pre-existing videos frame by frame.

Although some benign deepfake videos exist, they remain a minority. So far, the released tools [2] that generate deepfake videos have been broadly used to create fake celebrity pornographic videos or revenge porn [5]. This kind of pornography has already been banned by sites including Reddit, Twitter, and Pornhub. The realistic nature of deepfake videos also makes them a target for generation of pedopornographic material, fake news, fake surveillance videos, and malicious hoaxes. These fake videos have already been used to create political tensions and they are being taken into account by governmental entities [4].

As presented in the Malicious AI report [11], researchers in artificial intelligence should always reflect on the dual-use nature of their work, allowing misuse considerations to influence research priorities and norms. Given the severity of the malicious attack vectors that deepfakes have caused, in this paper we present a novel solution for the detection of this kind of video.

The main contributions of this work are summarized as follows. First, we propose a two-stage analysis composed of a CNN to extract features at the frame level followed by a temporally-aware RNN network to capture temporal inconsistencies between frames introduced by the face-swapping process. Second, we have used a collection of 600 videos to evaluate the proposed method, with half of the videos being deepfakes collected from multiple video hosting websites. Third, we show experimentally the effectiveness of the described approach, which allows use to detect if a suspect video is a deepfake manipulation with 94% more accuracy than a random detector baseline in a balanced setting.

2. Related Work

Digital Media Forensics. The field of digital media forensics aims to develop technologies for the automated assessment of the integrity of an image or video. Both feature-based [35, 16] and CNN-based [18, 19] integrity analysis methods have been explored in the literature. For video-based digital forensics, the majority of the proposed solutions try to detect computationally cheap manipulations, such as dropped or duplicated frames [40] or copy-move manipulations [9]. Techniques that detect face-based manipulations include methods that distinguish computer generated faces from natural ones such as Conotter et al. [13] or Rahmouni et al. [33]. In biometry, Raghavendra et al. [32] recently proposed to detect morphed faces with two pre-trained deep CNNs and Zhou et al. [41] proposed detection of two different face swapping manipulations using a two-stream network. Of special interest to practitioners is a new dataset by Rössler et al. [34], which has about half a million edited images that have been generated with feature-based face editing [38].

Face-based Video Manipulation Methods. Multiple approaches that target face manipulations in video sequences have been proposed since the 1990s [10, 14]. Thies et al. demonstrated the first real-time expression transfer for faces and later proposed Face2Face [38], a real-time facial reenactment system, capable of altering facial movements in different types of video streams. Alternatives to Face2Face have also been proposed [8].

Several face image synthesis techniques using deep learning have also been explored as surveyed by Lu et al. [29]. Generative adversarial networks (GANs) are used for aging alterations to faces [7], or to alter face attributes such as skin color [28]. Deep feature interpolation [39] shows remarkable results in altering face attributes such as age, facial hair or mouth expressions. Similar results of attribute interpolations are achieved by Lample et al. [24]. Most of these deep learning based image synthesis techniques suffer from low image resolution. Karras et al. [22] show high-quality synthesis of faces, improving the image quality using progressive GANs.

Recurrent Neural Networks. – Long Short Term Memory (LSTM) networks are a particular type of Recurrent Neural Network (RNN), first introduced by Hochreiter and Schmidhuber [20] to learn long-term dependencies in data sequences. When a deep learning architecture is equipped with a LSTM combined with a CNN, it is typically considered as "deep in space" and "deep in time" respectively, which can be seen as two distinct system modalities. CNNs have achieved massive success in visual recognition tasks, while LSTMs are widely used for long sequence processing problems. Because of the inherent properties (rich visual description, long-term temporal memory and end-to-end training) of a convolutional LSTM architecture, it has been thoroughly studied for other computer vision tasks involving sequences (e.g. activity recognition [15] or human re-identification in videos [30]) and has lead to significant improvements.

3. Deepfake Videos Exposed

Due to the way that FakeApp [2] generates the manipulated deepfake video, intra-frame inconsistencies and temporal inconsistencies between frames are created. These video anomalies can be exploited to detect if a video under analysis is a deepfake manipulation or not. Let us briefly explain how a deepfake video is generated to understand why these anomalies are introduced in the videos and how we can exploit them.

3.1. Creating Deepfake Videos

It is well known that deep learning techniques have been successfully used to enhance the performance of image compression. Especially, the autoencoder has been applied for dimensionality reduction, compact representations of images, and generative models learning [26]. Thus, autoencoders are able to extract more compressed representations of images with a minimized loss function and are expected to achieve better compression performance than existing image compression standards. The compressed representations or latent vectors that current convolutional autoencoders learn are the first cornerstone behind the faceswapping capabilities of [2]. The second insight is the use of two sets of encoder-decoders with shared weights for the encoder networks. Figure 2 shows how these ideas are used in the training and generation phases that happen during the creation of a deepfake video.

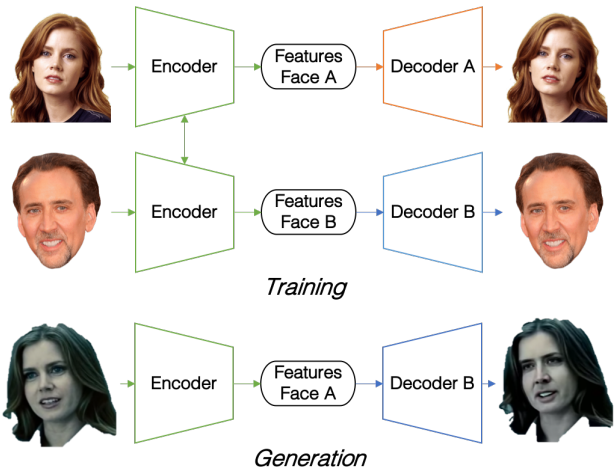


Figure 2. What makes deepfakes possible is finding a way to force both latent faces to be encoded on the same features. This is solved by having two networks sharing the same encoder, yet using two different decoders (top). When we want to do a new faceswapp, we encode the input face and decode it using the target face decoder (bottom).

3.1.1 Training

Two sets of training images are required. The first set only has samples of the original face that will be replaced, which can be extracted from the target video that will be manipulated. This first set of images can be further extended with images from other sources for more realistic results. The second set of images contains the desired face that will be swapped in the target video. To ease the training process of the autoencoders, the easiest face swap would have both the original face and target face under similar viewing and illumination conditions. However, this is usually not the case. Multiple camera views, differences in lightning conditions or simply the use of different video codecs makes it difficult for autencoders to produce realistic faces under all conditions. This usually leads to swapped faces that are visually inconsistent with the rest of the scene. This frame-level scene inconsistency will be the first feature that we will exploit with our approach.

It is also important to note that if we train two autoencoders separately, they will be incompatible with each other. If two autoencoders are trained separately on different sets of faces, their latent spaces and representations will be different. This means that each decoder is only able to decode a single kind of latent representations which it has learnt during the training phase. This can be overcome by forcing the two set of autoencoders to share the weights for the encoder networks, yet using two different decoders. In this fashion, during the training phase these two networks are treated separately and each decoder is only trained with faces from one of the subjects. However, all latent faces are produced by the same encoder which forces the encoder itself to identify common features in both faces. This can be easily accomplished due to the natural set of shared traits of all human faces (e.g. number and position of eyes, nose, ...).

3.1.2 Video Generation

When the training process is complete, we can pass a latent representation of a face generated from the original subject present in the video to the decoder network trained on faces of the subject we want to insert in the video. As shown in Figure 2, the decoder will try to reconstruct a face from the new subject, from the information relative to the original subject face present in the video. This process is repeated for every frame in the video where we want to do a faceswapping operation. It is important to point out that for doing this frame-level operation, first a face detector is used to extract only the face region that will be passed to the trained autoencoder. This is usually a second source of scene inconsistency between the swapped face and the reset of the scene. Because the encoder is not aware of the skin or other scene information it is very common to have boundary effects due to a seamed fusion between the new face and the rest of the frame.

The third major weakness that we exploit is inherent to the generation process of the final video itself. Because the autoencoder is used frame-by-frame, it is completely unaware of any previous generated face that it may have created. This lack of temporal awareness is the source of multiple anomalies. The most prominent is an inconsistent choice of illuminants between scenes with frames, with leads to a flickering phenomenon in the face region common to the majority of fake videos. Although this phenomenon can be hard to appreciate to the naked eye in the best manually-tuned deepfake manipulations, it is easily captured by a pixel-level CNN feature extractor. The phenomenon of incorrect color constancy in CNN-generated videos is a well known and still open research problem in the computer vision field [31]. Hence, it is not surprising that an autoencoder trained with very constrained data fails to render illuminants correctly.

4. Recurrent Network for Deepfake Detection

In this section, we present our end-to-end trainable recurrent deepfake video detection system (Figure 3). The proposed system is composed by a convolutional LSTM structure for processing frame sequences. There are two essential components in a convolutional LSTM:

1. CNN for frame feature extraction.

2. LSTM for temporal sequence analysis.

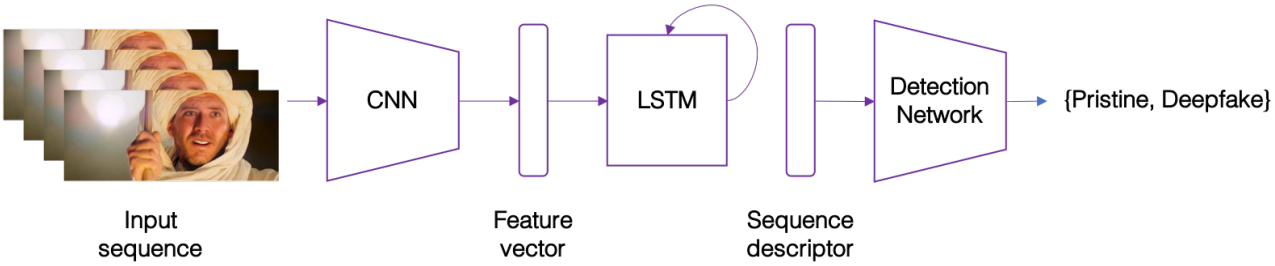


Figure 3. Overview of our detection system. The system learns and infers in an end-to-end manner and, given a video sequence, outputs a probability of it being a deepfake or a pristine video. It has a convolutional LSTM subnetwork, for processing the input temporal sequence.

Given an unseen test sequence, we obtain a set of features for each frame that are generated by the CNN. Afterwards, we concatenate the features of multiple consecutive frames and pass them to the LSTM for analysis. We finally produce an estimate of the likelihood of the sequence being either a deepfake or a nonmaninpulated video.

4.1. Convolutional LSTM

Given an image sequence (see Figure 3), a convolutional LSTM is employed to produce a temporal sequence descriptor for image manipulation of the shot frame. Aiming at end-to-end learning, an integration of fully-connected layers is used to map the high-dimensional LSTM descriptor to a final detection probability. Specifically, our shallow network consists of two fully-connected layers and one dropout layer to minimize training over-fitting. The convolutional LSTM can be divided into a CNN and a LSTM, which we will describe separately in the following paragraphs.

CNN for Feature Extraction. Inspired by its success in the IEEE Signal Processing Society Camera Model Identification Challenge, we adopt the InceptionV3 [36] with the fully-connected layer at the top of the network removed to directly output a deep representation of each frame using the ImageNet pre-trained model. Following [3], we do not fine-tune the network. The 2048-dimensional feature vectors after the last pooling layers are then used as the sequential LSTM input.

LSTM for Sequence Processing. Let us assume a sequence of CNN feature vectors of input frames as input and a 2-node neural network with the probabilities of the sequence being part of a deepfake video or an untampered video. The key challenge that we need to address is the design of a model to recursively process a sequence in a meaningful manner. For this problem, we resort to the use of a 2048-wide LSTM unit with 0.5 chance of dropout, which is capable to do exactly what we need. More particularly, during training, our LSTM model takes a sequence of 2048-dimensional ImageNet feature vectors. The LSTM is followed by a 512 fully-connected layer with 0.5 chance of dropout. Finally, we use a softmax layer to compute the probabilities of the frame sequence being either pristine or deepfake. Note that the LSTM module is an intermediate unit in our pipeline, which is trained entirely end-to-end without the need of auxiliary loss functions.

5. Experiments

In this section we report the details about our experiments. First, we describe our dataset. Then, we provide details of the experimental settings to ensure reproducibility and end up by analyzing the reported results.

5.1. Dataset

For this work, we have collected 300 deepfake videos from multiple video-hosting websites. We further incorporate 300 more videos randomly selected from the HOHA dataset [25], which leads to a final dataset with 600 videos. We selected the HOHA dataset as our source of pristine videos since it contains a realistic set of sequence samples from famous movies with an emphasis on human actions. Given that a considerable number of the deepfake videos are generated using clips from major films, using videos from the HOHA dataset further ensures that the overall system learns to spot manipulation features present in the deepfake videos, instead of memorizing semantic content from the two classes of videos present in the final dataset.

5.2. Parameter Settings

First, we have used a random 70/15/15 split to generate three disjoints sets, used for training, validation and test respectively. We do a balanced splitting, i.e., we do the splitting first for the 300 deepfake videos and then we repeat the process for the 300 nonmanipulated videos. This guarantees that each final set has exactly 50% videos of each class, which allows use to report our results in terms of accuracy without having to take into account biases due to the appearance frequency of each class or the need of using regularizing terms during the training phase. In terms of data preprocessing of the video sequences, we do:

- Subtracting channel mean from each channel.
- Resizing of every frame to 299×299.
- Sub-sequence sampling of length N controlling the length of input sequence – N = 20, 40, 80 frames. This allows use to see how many frames are necessary per video to have an accurate detection.
- The optimizer is set to Adam [23] for end-to-end training of the complete model with a learning rate of $1e{-}5$ and decay of $1e{-}6$.

5.3. Results

It is not unusual to find deepfake videos where the manipulation is only present in a small portion of the video (i.e. the target face only appears briefly on the video, hence the deepfake manipulation is short in time). To account for this, for every video in the training, validation and test splits, we extract continuous subsequences of fixed frame length that serve as the input of our system.

In Table 1 we present the performance of our system in terms of detection accuracy using sub-sequences of length $N = 20, 40, 80$ frames. These frame sequences are extracted sequentially (without frame skips) from each video. The entire pipeline is trained end-to-end until we reach a 10-epoch loss plateau in the validation set.

Model	Training acc. (%)	Validation acc. (%)	Test acc. (%)
Conv-LSTM, 20 frames	99.5	96.9	96.7
Conv-LSTM, 40 frames	99.3	97.1	97.1
Conv-LSTM, 80 frames	99.7	97.2	97.1

Table 1. Classification results of our dataset splits using video subsequences with different lengths.

As we can observe in our results, with less than 2 seconds of video (40 frames for videos sampled at 24 frames per second) our system can accurately predict if the fragment being analyzed comes from a deepfake video or not with an accuracy greater than 97%.

6. Conclusion

In this paper we have presented a temporal-aware system to automatically detect deepfake videos. Our experimental results using a large collection of manipulated videos have shown that using a simple convolutional LSTM structure we can accurately predict if a video has been subject to manipulation or not with as few as 2 seconds of video data.

We believe that our work offers a powerful first line of defense to spot fake media created using the tools described in the paper. We show how our system can achieve competitive results in this task while using a simple pipeline architecture. In future work, we plan to explore how to increase the robustness of our system against manipulated videos using unseen techniques during training.

Acknowledgments: This material is based on research sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under agreement number FA8750-16-2-0173. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, AFRL or the U.S. Government.

References

[1] Faceapp. https://www.faceapp.com/. (Accessed on 05/29/2018). 1

[2] Fakeapp. https://www.fakeapp.org/. (Accessed on 05/29/2018). 1, 2

[3] IEEE's Signal Processing Society - Camera Model Identification — Kaggle. https://www.kaggle.com/c/sp-society-camera-model-identification/discussion/49299. (Accessed on 05/29/2018). 4

[4] The Outline: Experts fear face swapping tech could start an international showdown. https://theoutline.com/post/3179/deepfake-videos-are-freaking-experts-out?zd=1&zi=hbmf4svs. (Accessed on 05/29/2018). 1

[5] What are deepfakes & why the future of porn is terrifying. https://www.highsnobiety.com/p/what-are-deepfakes-ai-porn/. (Accessed on 05/29/2018). 1

[6] M. Abadi et al. Tensorflow: A system for large-scale machine learning. *Proceedings of the USENIX Conference on Operating Systems Design and Implementation*, 16:265–283, Nov. 2016. Savannah, GA. 1

[7] G. Antipov, M. Baccouche, and J.-L. Dugelay. Face aging with conditional generative adversarial networks. *arXiv:1702.01983*, Feb. 2017. 1, 2

[8] H. Averbuch-Elor et al. Bringing portraits to life. *ACM Transactions on Graphics*, 36(6):196:1–196:13, Nov. 2017. 2

[9] P. Bestagini et al. Local tampering detection in video sequences. *Proceedings of the IEEE International Workshop on Multimedia Signal Processing*, pages 488–493, Sept. 2013. Pula, Italy. 2

[10] C. Bregler, M. Covell, and M. Slaney. Video rewrite: Driving visual speech with audio. *Proceedings of the ACM Annual*

Conference on Computer Graphics And Interactive Techniques, pages 353–360, Aug. 1997. Los Angeles, CA. 2

[11] M. Brundage et al. The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv:1802.07228*, Feb. 2018. 2

[12] F. Chollet et al. Keras. https://keras.io, 2015. 1

[13] V. Conotter, E. Bodnari, G. Boato, and H. Farid. Physiologically-based detection of computer generated faces in video. *Proceedings of the IEEE International Conference on Image Processing*, pages 248–252, Oct. 2014. Paris, France. 2

[14] K. Dale, K. Sunkavalli, M. K. Johnson, D. Vlasic, W. Matusik, and H. Pfister. Video face replacement. *ACM Transactions on Graphics*, 30(6):1–130, Dec. 2011. 2

[15] J. Donahue et al. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):677–691, Apr. 2017. 2

[16] H. Farid. *Photo Forensics*. MIT Press Ltd, 2016. 2

[17] I. Goodfellow et al. Generative adversarial nets. *Advances in Neural Information Processing Systems*, pages 2672–2680, Dec. 2014. Montréal, Canada. 1

[18] D. Güera, Y. Wang, L. Bondi, P. Bestagini, S. Tubaro, and E. J. Delp. A counter-forensic method for CNN-based camera model identification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1840–1847, July 2017. Honolulu, HI. 2

[19] D. Güera, S. K. Yarlagadda, P. Bestagini, F. Zhu, S. Tubaro, and E. J. Delp. Reliability map estimation for cnn-based camera model attribution. *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, Mar. 2018. Lake Tahoe, NV. 2

[20] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, Nov. 1997. 2

[21] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5967–5976, July 2017. Honolulu, HI. 1

[22] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv:1710.10196*, Oct. 2017. 2

[23] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, Dec. 2014. 5

[24] G. Lample et al. Fader networks: Manipulating images by sliding attributes. *Advances in Neural Information Processing Systems*, pages 5967–5976, Dec. 2017. Long Beach, CA. 2

[25] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008. Anchorage, AK. 4

[26] Y. Liao, Y. Wang, and Y. Liu. Graph regularized auto-encoders for image representation. *IEEE Transactions on Image Processing*, 26(6):2839–2852, June 2017. 2

[27] S. Lorant. *Lincoln; a picture story of his life*. Norton, 1969. 1

[28] Y. Lu, Y.-W. Tai, and C.-K. Tang. Conditional cyclegan for attribute guided face image generation. *arXiv:1705.09966*, May 2017. 2

[29] Z. Lu, Z. Li, J. Cao, R. He, and Z. Sun. Recent progress of face image synthesis. *arXiv:1706.04717*, June 2017. 2

[30] N. McLaughlin, J. M. d. Rincon, and P. Miller. Recurrent convolutional network for video-based person re-identification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1325–1334, June 2016. Las Vegas, NV. 2

[31] Y. Qian et al. Recurrent color constancy. *Proceedings of the IEEE International Conference on Computer Vision*, pages 5459–5467, Oct. 2017. Venice, Italy. 3

[32] R. Raghavendra, K. B. Raja, S. Venkatesh, and C. Busch. Transferable deep-cnn features for detecting digital and print-scanned morphed face images. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1822–1830, July 2017. Honolulu, HI. 2

[33] N. Rahmouni, V. Nozick, J. Yamagishi, and I. Echizen. Distinguishing computer graphics from natural images using convolution neural networks. *Proceedings of the IEEE Workshop on Information Forensics and Security*, pages 1–6, Dec. 2017. Rennes, France. 2

[34] A. Rössler et al. Faceforensics: A large-scale video dataset for forgery detection in human faces. *arXiv:1803.09179*, Mar. 2018. 2

[35] H. T. Sencar and N. Memon, editors. *Digital Image Forensics*. Springer New York, 2013. 2

[36] C. Szegedy et al. Rethinking the inception architecture for computer vision. *Proeedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, June 2016. Las Vegas, NV. 4

[37] A. Tewari et al. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1274–1283, Oct. 2017. Venice, Italy. 1

[38] J. Thies et al. Face2Face: Real-time face capture and reenactment of rgb videos. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2387–2395, June 2016. Las Vegas, NV. 1, 2

[39] P. Upchurch et al. Deep feature interpolation for image content changes. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6090–6099, July 2017. Honolulu, HI. 2

[40] W. Wang and H. Farid. Exposing digital forgeries in interlaced and deinterlaced video. *IEEE Transactions on Information Forensics and Security*, 2(3), 2007. 2

[41] P. Zhou et al. Two-stream neural networks for tampered face detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1831–1839, July 2017. Honolulu, HI. 2

[42] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2242–2251, Oct. 2017. Venice, Italy. 1

Annexure B

Review Paper

Annexure C

Papers Published & Certificates

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:07/Issue:05/May-2025 Impact Factor- 8.187 www.irjmets.com

DEEPFAKE VIDEO DETECTION USING NEURAL NETWORKS

Nikhil Shinde[*1], Jaydeep Nigade[*2], Yashkumar Bagal[*3], Rohan Avatade[*4], Rutuja Taware[*5]

[*1,2,3,4]Student, Department Of Computer Engineering, SVPM's College Of Engineering, Malegaon (Bk), Maharashtra, India.

[*5]Professor, Department Of Computer Engineering, SVPM's College Of Engineering, Malegaon (Bk), Maharashtra, India.

DOI : https://www.doi.org/10.56726/IRJMETS75444

ABSTRACT

In recent times, the emergence of free deep learning-based software tools has made it easier to generate highly realistic face-swapped videos, commonly known as "DeepFakes" (DF). While video manipulation has been possible for decades using traditional visual effects, recent breakthroughs in deep learning have significantly enhanced the realism of such content and lowered the barrier to creating it. These AI-generated videos, also known as AI-synthesized media, have become increasingly convincing and difficult to detect.

Although generating DeepFakes has become relatively straightforward with the help of artificial intelligence, detecting them remains a complex and challenging task. Training models to reliably identify manipulated content requires sophisticated techniques. In this work, we propose a DeepFake detection approach that leverages both Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). The CNN is used to extract spatial features from individual video frames, which are then passed to an RNN that captures temporal dependencies and inconsistencies across frames caused by DeepFake generation tools.

Our system is evaluated on a large, publicly available dataset of manipulated videos. The experimental results demonstrate that our approach, despite using a relatively simple architecture, achieves competitive performance in detecting DeepFakes.

Keywords: Deepfake Video Detection, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN).

I. INTRODUCTION

With the rapid advancement of smartphone camera technology and widespread access to high-speed internet, the creation and sharing of digital videos has become easier than ever. This, combined with the rise of social media platforms, has significantly amplified the spread of video content across the globe. At the same time, increasing computational power has fueled the progress of deep learning, enabling applications that were once considered impossible. However, like many emerging technologies, these advancements bring new challenges—one of the most pressing being the rise of "DeepFakes."

DeepFakes are synthetic videos generated using deep learning techniques, particularly Generative Adversarial Networks (GANs), which can manipulate both video and audio to produce highly realistic but fake media. The ease with which DeepFakes can now be created and shared has led to a surge in misinformation, harassment, and deception on digital platforms. This makes the detection of DeepFakes critical to preserving the integrity of information online.

To address this issue, we propose a novel deep learning-based approach for detecting AI-generated fake videos. Identifying DeepFakes requires a thorough understanding of how GANs generate these videos. Typically, GANs take a video and an image of a 'target' person and produce a new video where the target's face is replaced with that of a 'source' individual. This process is powered by deep neural networks trained on facial images and target videos to accurately map the expressions and movements of the source onto the target. The video is processed frame by frame, and the modified frames are then stitched back together, often using autoencoders to achieve a high degree of realism.

However, DeepFake generation is not without limitations. Due to constraints in computational resources and time, the GAN-generated face images are usually of a fixed resolution and must be warped to align with the

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:07/Issue:05/May-2025 Impact Factor- 8.187 www.irjmets.com

target face's configuration. This affine transformation introduces noticeable artifacts—specifically, inconsistencies between the resolution of the synthesized face area and the surrounding regions.

Our detection method is designed to exploit these artifacts. We analyze each frame of the video, using a ResNeXt-based Convolutional Neural Network (CNN) to extract visual features, focusing on the contrast between the manipulated face and the background. To further enhance accuracy, we employ a Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) units to identify temporal inconsistencies between frames—irregularities often introduced during DeepFake video reconstruction.

To train the ResNeXt model, we simulate the resolution mismatches typically found in DeepFakes by applying affine transformations to facial images, helping the model learn to recognize these subtle discrepancies. This approach allows our system to effectively distinguish between real and AI-generated content, providing a promising solution for combating the spread of DeepFakes online.

II. LITERATURE SURVEY

The rapid rise in the creation and misuse of DeepFake videos poses a significant threat to democracy, justice, and public trust. This growing concern has led to an increasing demand for advanced methods in fake video analysis, detection, and intervention. Several existing approaches in DeepFake detection are highlighted below:

1. Face Warping Artifact Detection

The method presented in "Exposing DeepFake Videos by Detecting Face Warping Artifacts" \[1] focuses on identifying inconsistencies by analyzing differences between synthetically generated facial areas and their surrounding regions using a dedicated Convolutional Neural Network (CNN). Their approach is grounded in the observation that current DeepFake generation techniques typically produce facial images at limited resolutions, which must then be scaled and adjusted to fit the source video, often introducing detectable artifacts.

2. Eye Blinking Analysis

The work titled "Exposing AI-Created Fake Videos by Detecting Eye Blinking" \[2] introduces a novel approach to identify fake videos based on the absence of natural eye blinking—a physiological cue often missing in AI-generated videos. The method is validated using eye-blinking detection benchmark datasets and has shown promising results. However, relying solely on blinking may not be sufficient. Additional facial attributes like teeth detail, skin texture, and wrinkle patterns are crucial, and our proposed method incorporates such features for a more comprehensive analysis.

3. Capsule Network-Based Detection

In "Using Capsule Networks to Detect Forged Images and Videos" \[3], the authors employ capsule networks to distinguish manipulated media across various scenarios, including replay attacks and AI-generated videos. Although their model performs well on controlled datasets, the use of random noise during training could hinder its effectiveness on real-world data. Our proposed method addresses this limitation by training on clean, real-time datasets for improved generalization.

4. Biological Signal-Based Detection

The study "Detection of Synthetic Portrait Videos Using Biological Signals" \[5] explores the use of biological signals—such as those captured from facial blood flow patterns (e.g., via PPG signals)—to detect fake videos. The approach involves extracting features related to spatial and temporal consistency from real and synthetic video pairs. These features are then used to train a CNN and probabilistic SVM for classification. The "FakeCatcher" system from this research achieves high detection accuracy across various conditions, independent of the video's source, content, or resolution. However, the absence of a discriminator component and the complexity in designing a differentiable loss function to preserve biological signals pose limitations.

III. PROPOSED SYSTEM

While numerous tools exist for creating DeepFakes (DF), there is a significant lack of reliable and accessible solutions for detecting them. Our proposed approach aims to fill this gap by offering an effective method for identifying DeepFake content, thereby helping to prevent its spread across the internet. A key component of our solution is a user-friendly web-based platform where users can upload videos to determine whether they are authentic or manipulated.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:07/Issue:05/May-2025 Impact Factor- 8.187 www.irjmets.com

This project has strong potential for scalability—from a basic web application to a browser extension that can automatically detect DeepFakes in real time. Furthermore, popular messaging and social media platforms such as WhatsApp and Facebook could integrate this solution to enable automatic detection of DeepFake content before it is shared with others.

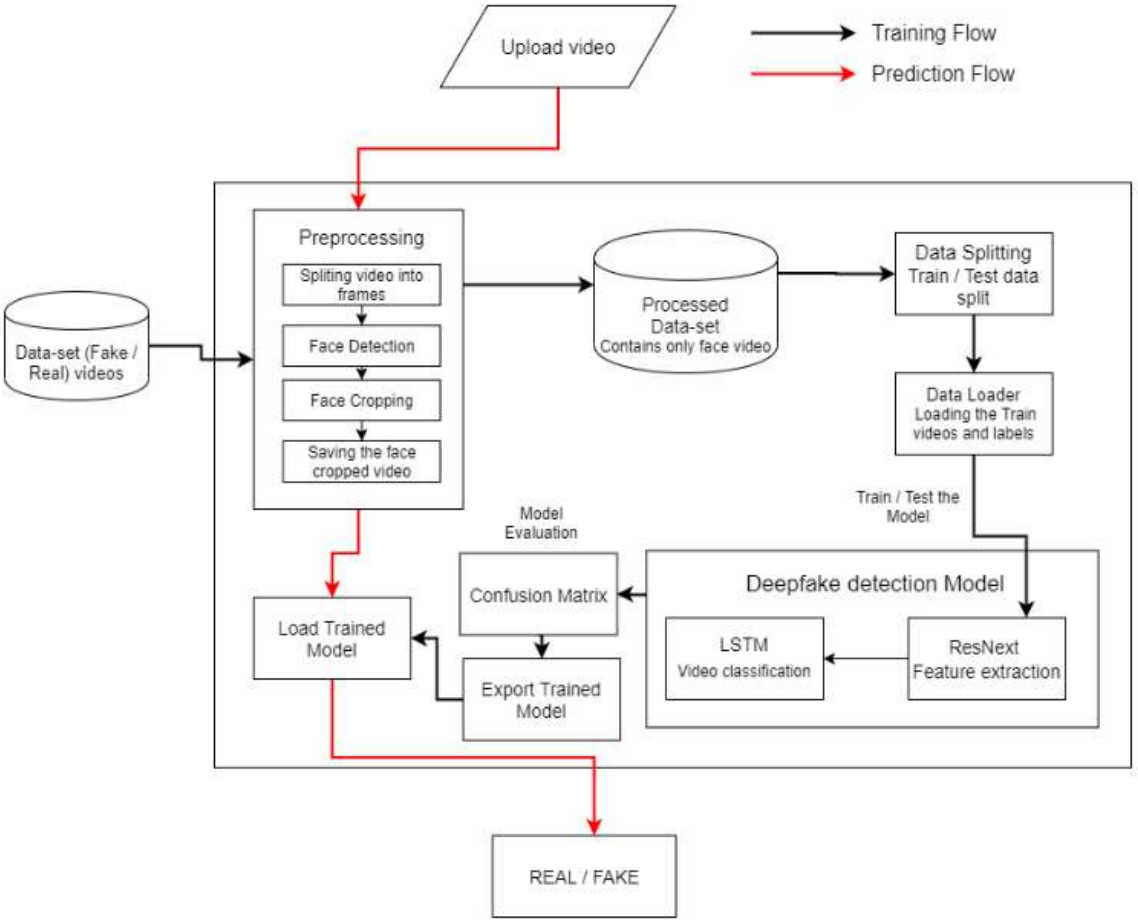


Fig. 1: System Architecture

A. Dataset:

We utilize a combined dataset that includes an equal distribution of videos sourced from various publicly available datasets such as YouTube, FaceForensics++ \[14], and the DeepFake Detection Challenge dataset \[13]. The curated dataset is composed of 50% authentic (real) videos and 50% manipulated (DeepFake) videos. For model training and evaluation, the dataset is divided into two subsets: 70% for training and 30% for testing.

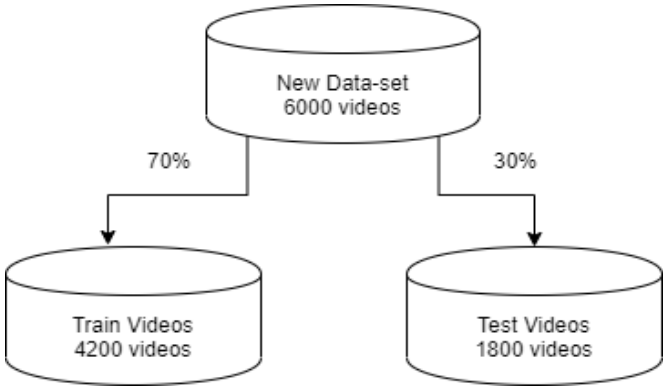


Fig. 2: Dataset

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:07/Issue:05/May-2025 Impact Factor- 8.187 www.irjmets.com

B. Preprocessing:

The dataset preprocessing involves several steps, starting with splitting each video into individual frames. Next, face detection is performed on these frames, and only the regions containing faces are cropped and retained for further processing. To ensure consistency in the number of frames across all videos, the mean number of frames per video in the dataset is calculated. A new, standardized dataset is then generated, where each video sample contains a number of face-cropped frames equal to this mean. Any frames that do not contain a detectable face are excluded during preprocessing.

Processing an entire 10-second video at 30 frames per second results in approximately 300 frames, which can be computationally intensive. To manage resource constraints during experimentation, we limit the input to the first 100 frames per video for training the model.

C. Model:

The proposed model architecture comprises a ResNeXt-50 (32x4d) network followed by a single Long Short-Term Memory (LSTM) layer. The data pipeline begins with a Data Loader that loads the preprocessed, face-cropped video samples and splits them into training and testing sets. During model training and evaluation, frames from these videos are processed in mini-batches and passed through the model to learn spatial and temporal features for DeepFake detection.

D. ResNext CNN for Feature Extraction:

Rather than developing a classifier from scratch, we propose utilizing the ResNeXt CNN architecture for effective feature extraction and accurate detection of frame-level characteristics. To enhance performance, the network will be fine-tuned by adding necessary layers and selecting an appropriate learning rate to ensure stable and efficient convergence during gradient descent. The output from the final pooling layer of ResNeXt, which consists of 2048-dimensional feature vectors, serves as the input sequence for the subsequent LSTM layer.

E. LSTM for Sequence Processing:

Consider a sequence of feature vectors extracted from video frames using the ResNeXt CNN. These vectors are fed into a neural network with two output nodes representing the probabilities of the sequence belonging to either a DeepFake video or an authentic (untampered) video. A central challenge in this approach is designing a model that can effectively interpret the temporal sequence of frames. To address this, we propose using an LSTM layer with 2048 units and a dropout rate of 0.4. This configuration enables the model to perform sequential analysis, capturing temporal dependencies across frames. The LSTM processes each frame in order, allowing the model to compare the state of a frame at time `t` with one at time `t-n`, where `n` refers to the number of frames prior to `t`, thus enabling meaningful temporal context analysis.

IV. RESULT

The model's output will indicate whether the video is a DeepFake or authentic, along with the associated confidence level of the prediction.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:07/Issue:05/May-2025 Impact Factor- 8.187 www.irjmets.com

V. LIMITATIONS

Our method currently does not account for audio, meaning it is unable to detect audio-based DeepFakes. However, we plan to incorporate audio DeepFake detection in future iterations of the system.

VI. CONCLUSION

We have presented a neural network-based approach for classifying videos as either DeepFake or authentic, along with the confidence level of the model's prediction. The proposed method draws inspiration from the generation of DeepFakes using GANs and Autoencoders. Our approach employs ResNeXt CNN for frame-level feature extraction and utilizes an RNN with LSTM for video classification. The system is designed to accurately determine whether a video is a DeepFake or real, based on the parameters outlined in this paper. We believe that this approach will achieve high accuracy when applied to real-time data.

VII. REFERENCES

[1] Yuezun Li, Siwei Lyu. “Exposing DeepFake Videos by Detecting Face Warping Artifacts,” arXiv:1811.00656v3.

[2] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. “Exposing AI-Generated Fake Videos by Detecting Eye Blinking,” arXiv.

[3] Huy H. Nguyen, Junichi Yamagishi, and Isao Echizen. “Using Capsule Networks to Detect Forged Images and Videos.”

[4] Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, and Weipeng Xu. “Deep Video Portraits,” arXiv:1901.02212v2.

[5] Umur Aybars Ciftci, İlke Demir, Lijun Yin. “Detection of Synthetic Portrait Videos Using Biological Signals,” arXiv:1901.02212v2.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. “Generative Adversarial Nets,” NIPS, 2014.

[7] David Güera, Edward J. Delp. “DeepFake Video Detection Using Recurrent Neural Networks,” AVSS, 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition,” CVPR, 2016.

[9] An Overview of ResNet and Its Variants: https://towardsdatascience.com/an-overview-of-resnetand-its-variants-5281e2f56035

[10] Long Short-Term Memory: From Zero to Hero with Pytorch: https://blog.floydhub.com/long-short-term-memory-from-zero-to-hero-with-pytorch/

[11] Sequence Models and LSTM Networks:
https://pytorch.org/tutorials/beginner/nlp/sequence_models_tutorial.html

[12] https://discuss.pytorch.org/t/confused-about-the-image-preprocessing-in-classification/3965

[13] https://www.kaggle.com/c/deepfake-detection-challenge/data

[14] https://github.com/ondyari/FaceForensics

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500029158

DOI : https://www.doi.org/10.56726/IRJMETS75444

Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Rohan Avatade**" with paper ID "**IRJMETS70500029158**" has published a paper entitled "DEEPFAKE VIDEO DETECTION USING NEURAL NETWORKS" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500029158

DOI : https://www.doi.org/10.56726/IRJMETS75444

Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Yashkumar Bagal**" with paper ID "**IRJMETS70500029158**" has published a paper entitled "DEEPFAKE VIDEO DETECTION USING NEURAL NETWORKS" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500029158

DOI : https://www.doi.org/10.56726/IRJMETS75444

Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Nikhil Shinde**" with paper ID "**IRJMETS70500029158**" has published a paper entitled "DEEPFAKE VIDEO DETECTION USING NEURAL NETWORKS" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500029158

DOI : https://www.doi.org/10.56726/IRJMETS75444

Date: 09/05/2025

Certificate of Publication

*This is to certify that author "**Jaydeep Nigade**" with paper ID "**IRJMETS70500029158**" has published a paper entitled "DEEPFAKE VIDEO DETECTION USING NEURAL NETWORKS" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

Annexure D

Certificates - Cretechnova / Smart India Hackathon-2024

Shivnagar Vidya Prasarak Mandal's

College Of Engineering Malegaon (Bk.)

Tal -Baramati, Dist - Pune 413115

CRETECHNOVA 2K25

A National Level Technical Symposium

CERTIFICATE

This is to certify that Mr. / ~~Miss~~ Avatade Rohan Ramdas

has Participated / Co-ordinated / Secured participated in the technical event

Comp pro-expo held during **"CRETECHNOVA 2K25"**

organised by SVPM's College of Engineering, Malegaon (Bk) on 28th & 29th March, 2025.

Dr. Sangram D. Jadhav
Convener

Dr. Shailendrakumar M. Mukane
Principal

Shivnagar Vidya Prasarak Mandal's

College Of Engineering Malegaon (Bk.)

Tal -Baramati, Dist - Pune 413115

CRETECHNOVA 2K25

A National Level Technical Symposium

CERTIFICATE

This is to certify that Mr. / ~~Miss~~ Shinde Nikhil Kailas

has Participated / Co-ordinated / Secured participated in the technical event

Comp pro-expo held during **"CRETECHNOVA 2K25"**

organised by SVPM's College of Engineering, Malegaon (Bk) on 28th & 29th March, 2025.

Dr. Sangram D. Jadhav
Convener

Dr. Shailendrakumar M. Mukane
Principal

Ministry of Education
Government of India

SMART INDIA HACKATHON 2024

MoE's INNOVATION CELL
(GOVERNMENT OF INDIA)

Smart India Hackathon 2024

(#SIH2024)

This certificate is awarded to

Mr./ Ms. Jaydeep Popatrao Nigade

for participating in the Internal Smart India Hackathon Competition organized by **SVPM's College of Engineering, Malegaon(Bk)** in collaboration with Smart India Hackathon(SIH)-2024.

Dr.Devendra P. Agrawal
SPOC-SIH

Dr.Shailendra M. Mukane
PRINCIPAL

Ministry of Education
Government of India

SMART INDIA HACKATHON 2024

MoE's INNOVATION CELL
(GOVERNMENT OF INDIA)

Smart India Hackathon 2024

(#SIH2024)

This certificate is awarded to

Mr./ Ms. Bagal Yashkumar Ramchandra

for participating in the Internal Smart India Hackathon Competition organized by **SVPM's College of Engineering, Malegaon(Bk)** in collaboration with Smart India Hackathon(SIH)-2024.

Dr.Devendra P. Agrawal
SPOC-SIH

Dr.Shailendra M. Mukane
PRINCIPAL

Annexure E

Plagiarism Report

GROUP16 Report.pdf

ORIGINALITY REPORT

18%	13%	9%	10%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	"Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2013", Springer Science and Business Media LLC, 2014 Publication	3%
2	Submitted to Army Institute of Technology Student Paper	3%
3	Submitted to University of Newcastle upon Tyne Student Paper	2%
4	www.coursehero.com Internet Source	2%
5	Submitted to Higher Education Commission Pakistan Student Paper	1%
6	Submitted to Pace University Student Paper	1%
7	ijritcc.org Internet Source	1%
	Submitted to Canterbury College, Kent	

8	Student Paper	1 %
9	Bhanu Chander, Koppala Guravaiah, B. Anoop, G. Kumaravelan. "Handbook of AI-Based Models in Healthcare and Medicine - Approaches, Theories, and Applications", CRC Press, 2024 Publication	1 %
10	Submitted to Rutgers University, New Brunswick Student Paper	1 %
11	www.ijraset.com Internet Source	<1 %
12	Submitted to Manipal University Student Paper	<1 %
13	www.nature.com Internet Source	<1 %
14	Hamid Jahankhani, Stefan Kendzierskyj, Reza Montasari, Nishan Chelvachandran. "Social Media Analytics, Strategies and Governance", CRC Press, 2022 Publication	<1 %
15	hdl.handle.net Internet Source	<1 %
16	Submitted to University of Hertfordshire Student Paper	<1 %

17	www.suppotech.com Internet Source	<1 %
18	"Biometric Recognition", Springer Science and Business Media LLC, 2019 Publication	<1 %
19	www.mdpi.com Internet Source	<1 %
20	pubmed.ncbi.nlm.nih.gov Internet Source	<1 %
21	robspcr.files.wordpress.com Internet Source	<1 %
22	123dok.net Internet Source	<1 %
23	"Database and Expert Systems Applications", Springer Science and Business Media LLC, 2021 Publication	<1 %
24	"Information and Communications Security", Springer Science and Business Media LLC, 2020 Publication	<1 %
25	M. M. El-Gayar, Mohamed Abouhawwash, S. S. Askar, Sara Sweidan. "A novel approach for detecting deep fake videos using graph neural network", Journal of Big Data, 2024 Publication	<1 %

26	d-scholarship.pitt.edu Internet Source	<1 %
27	www.essaysauce.com Internet Source	<1 %
28	Mingsong Dou, Philip Davidson, Sean Ryan Fanello, Sameh Khamis et al. "Motion2fusion", ACM Transactions on Graphics, 2017 Publication	<1 %
29	discovery.researcher.life Internet Source	<1 %
30	ijircce.com Internet Source	<1 %
31	www.researchgate.net Internet Source	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

Annexure F

Group Members

1. Group Members

Project Title: Deepfake Detection System

Group Members and Contribution:

- Avatade Rohan Ramdas – Dataset Collection and Initial Idea
- Bagal Yashkumar Ramchandra – Developer and Implementation
- Nigade Jaydeep Popatrao – Planning and Testing
- Shinde Nikhil Kailas – Documentation and Research Publication

Project Guide: Prof. R. K. Taware
Department of Computer Engineering,
SVPM's College of Engineering, Malegaon (Bk.), Baramati