DEEPFAKE VIDEO DETECTION USING NEURAL NETWORKS

A Main project thesis submitted in partial fulfillment of requirements for the award of degree for

VIII semester

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

P RAVI KUMAR REDDY (20131A05J5)

M VAMSI KRISHNA (20131A05F0)

N N V S S L SIRI VAISHNAVI (20131A05G0)

M V G AASRITHA (20131A05F3)

Under the esteemed guidance of

Ms. P. POOJA RATNAM,

Assistant Professor,

Department of Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (AUTONOMOUS)

Approved by AICTE & Affiliated to Andhra University, Visakhapatnam from 2022-23
(Affiliated to JNTUK,Kakinada upto 2021-22)

Accredited twice by NAAC with 'A' Grade with a CGPA of 3.47/4.00
Madhurawada, Visakhapatnam-530048

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING
(Autonomous)
Approved by AICTE & Affiliated to Andhra University, Visakhapatnam from 2022-23
(Affiliated to JNTUK, Kakinada upto 2021-22)
Accredited twice by NAAC with 'A' Grade with a CGPA of 3.47/4.00
Madhurawada, Visakhapatnam - 530048

CERTIFICATE

This is to certify that the main project entitled

"**DEEPFAKE VIDEO DETECTION USING NEURALNETWORKS**"

being submitted by

P RAVI KUMAR REDDY (20131A05J5)
M VAMSI KRISHNA (20131A05F0)
N N V S S L SIRI VAISHNAVI (20131A05G0)
M V G AASRITHA (20131A05F3)

in partial fulfilment for the award of the degree "**BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**" to the Jawaharlal Nehru Technological University, Kakinada is a record of bonafide work done under my guidance and supervision during VIII semester of the academic year 2023-2024.

The results embodied in this record have not been submitted to any other university orinstitution for the award of any Degree or Diploma.

Guide

Ms. P. Pooja Ratnam
Assistant Professor
Department of CSE
GVPCE(A)

Incharge Head of the Department

Dr. D. Uma Devi
Associate Professor & H.O.D
Department of CSE(AI&ML)
GVPCE(A)

External Examiner

DECLARATION

We hereby declare that this project entitled **"DEEPFAKE VIDEO DETECTION USING NEURAL NETWORKS"** is a bonafide work done by us and submitted to "**Department of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (Autonomous), Visakhapatnam**", in partial fulfilment for the award of the degree of B. Tech is of our own and it is not submitted to any other university or has been published anytime before.

PLACE: VISAKHAPATNAM	P RAVI KUMAR REDDY	(20131A05J5)
DATE:	M VAMSI KRISHNA	(20131A05F0)
	N N V S S L SIRI VAISHNAVI	(20131A05G0)
	M V G AASRITHA	(20131A05F3)

ACKNOWLEDGEMENT

We would like to express our deep sense of gratitude to our esteemed institute **Gayatri Vidya Parishad College of Engineering (Autonomous)**, which has provided us an opportunity to fulfill our cherished desire.

We express our sincere thanks to our principal **Dr. A. B. KOTESWARA RAO, Gayatri Vidya Parishad College of Engineering (Autonomous)** for his encouragement to us during this project, giving us a chance to explore and learn new technologies in the form of mini projects.

We express our deep sense of Gratitude to **Dr. D. UMA DEVI**, **Associate Professor and Associate head of CSE with AI &ML and Incharge Head of the Department of Computer Science and Engineering, Gayatri Vidya Parishad College of Engineering (Autonomous)** for giving us an opportunity to do the project in college.

We express our profound gratitude and our deep indebtedness to our guide **Ms. P. POOJA RATNAM**, **Assistant Professor, Department of Computer Science and Engineering**, whose valuable suggestions, guidance and comprehensive assessments helped us a lot in realizing our project.

We also thank our coordinator, **Dr. CH. SITA KUMARI**, **Associate Professor, Department of Computer Science and Engineering**, **Mrs. K. SWATHI**, **Assistant Professor, Department of Computer Science and Engineering, and Ms. P. POOJA RATNAM**, **Assistant Professor, Department of Computer Science and Engineering** for the kind suggestions and guidance for the successful completion of our project work.

P RAVI KUMAR REDDY (20131A05J5)
M VAMSI KRISHNA (20131A05F0)
N N V S S L SIRI VAISHNAVI (20131A05G0)
M V G AASRITHA (20131A05F3)

ABSTRACT

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

Keywords: Res-Next Convolution Neural Network, Recurrent Neural Network(RNN), Long Short Term Memory(LSTM)

INDEX

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 easilyenvisioned. 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. Deepfakes are created using tools like FaceApp[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. ResNext Convolution neural network extracts the frame-level features and these features are further used to train the Long Short Term Memory based artificial Recurrent Neural Network to classify the video as Deepfake or real. To emulate the real time scenarios and make the model perform better on real time data, we trained our method with large amount of balanced and combination of various available dataset like FaceForensic++[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 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.

1.1. OBJECTIVE

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

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

fake videos and their degrees of realism has been increasing due to availability of the editing tools, the high demand on domain expertise.

Spreading of the Deepfakes over the social media platforms have become very common leading to spamming and peculating wrong information over the plat- form. Just imagine a deep fake of our prime minister declaring war against neighboring countries, or a Deep fake of reputed celebrity abusing the fans. These types of the deep fakes will be terrible, and lead to threatening,misleading of common people.

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

1.2. ABOUT THE ALGORITHM

1.2.1 ResNext :

Instead of writing the code from scratch, we used the pre-trained model of ResNext for feature extraction. ResNext is Residual CNN network optimized for high performance on deeper neural networks. For the experimental purpose we have used resnext50_32x4d model. We have used a ResNext of 50 layers and 32 x 4 dimensions. Following, we will be fine-tuning the network by adding extra required layers and selecting a proper learning rate to properly converge the gradient descent of the model. The 2048-dimensional feature vectors after the last pooling layers of ResNext is used as the sequential LSTM input.

1.2.2 . LSTM

2048-dimensional feature vectors is fitted as the input to the LSTM. We are using 1 LSTM layer with 2048 latent dimensions and 2048 hidden layers along with 0.4 chance of dropout, which is capable to do achieve our objective. LSTM is used to process the frames in a sequential manner so that the temporal analysis of the video can be made, by comparing the frame at 't' second with the frame of 't-n' seconds. Where n can be any number of frames before t. The model also consists of Leaky ReLu activation function. A linear layer of 2048 input features and 2 output features are used to make the model capable of learning the average rate of correlation between input and output. An adaptive average polling layer with the output parameter 1 is used in the model. Which gives the target output size of the image of the form H x W. For sequential processing of the frames a Sequential Layer is used. The batch size of 4 is used to perform the batch training. A Soft Max layer is used to get the confidence of the model during prediction.

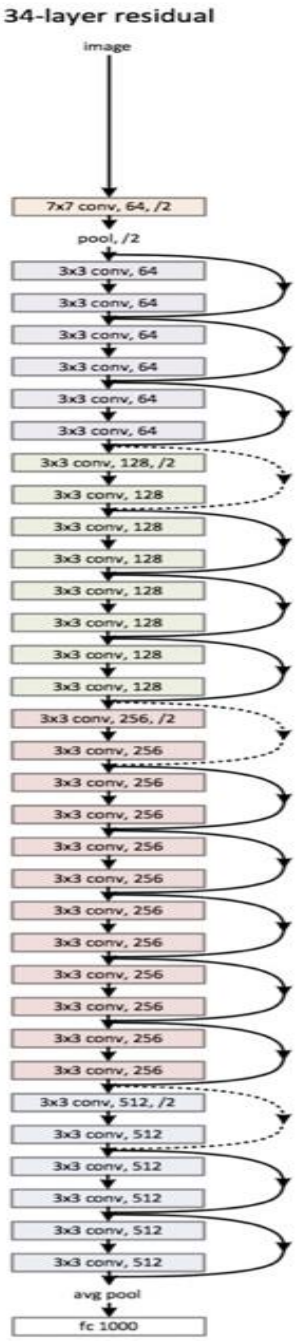


Figure 1.1: Overview of ResNeXt Architecture

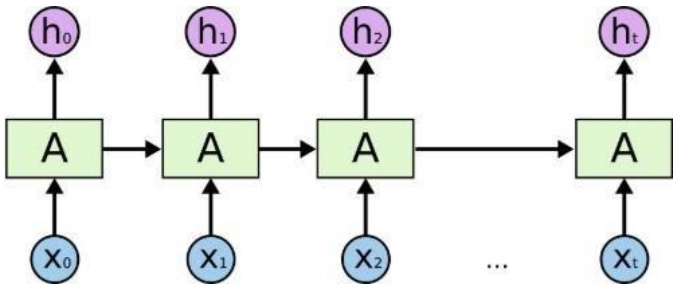


Figure 1.2: Overview of LSTM Architecture

1.3. PURPOSE

In recent years, the proliferation of deepfake technology has raised significant concerns regarding its potential misuse for generating highly convincing yet fabricated videos. Deepfake videos are created using artificial intelligence techniques, often employing deep learning models, to manipulate and superimpose images and videos onto existing footage, making it difficult to discern between real and fake content. Detecting such manipulated videos has become crucial for safeguarding against the spread of misinformation, protecting individuals' reputations, and maintaining trust in visual media.

The purpose of employing a combination of ResNext (a convolutional neural network architecture) and LSTM (Long Short-Term Memory, a type of recurrent neural network) for deepfake video detection lies in their respective strengths and capabilities:

1. **ResNeXt for Feature Extraction**: ResNeXt is a deep learning architecture renowned for its effectiveness in feature extraction tasks. Deepfake detection involves analyzing various visual cues and patterns present in both authentic and manipulated videos. ResNeXt can efficiently extract discriminative features from frames or patches of frames, capturing subtle differences that may indicate tampering or manipulation. By leveraging ResNeXt, the model can learn to distinguish between authentic and deepfake videos based on these extracted features.

2. **LSTM for Temporal Modeling**: Unlike traditional image classification tasks where individual frames are considered in isolation, deepfake detection often requires analyzing temporal dependencies and patterns across consecutive frames. LSTM, a type of recurrent neural network, excels at capturing temporal information and modeling sequences effectively. By incorporating LSTM layers into the deepfake detection model, it can learn to recognize temporal inconsistencies or irregularities that may be indicative of deepfake manipulation. This enables the model to consider the context and temporal dynamics of the video, enhancing its ability to differentiate between real and fake content.

3. **Combining Strengths for Robust Detection**: By integrating ResNeXt for feature extraction and LSTM for temporal modeling, the deepfake detection model can leverage the complementary strengths of both architectures. ResNeXt enables the model to capture fine-grained visual features, while LSTM facilitates the understanding of temporal relationships and context within the video sequence. This synergistic combination enhances the robustness and effectiveness of the detection system, enabling it to more accurately identify deepfake videos across a wide range of scenarios and manipulation techniques.

4. **Adaptability and Generalization**: Another advantage of using ResNeXt and LSTM is their adaptability and generalization capabilities. These architectures have been extensively studied and validated across various computer vision and sequential modeling tasks, demonstrating their effectiveness in diverse domains. By leveraging these well-established architectures, the deepfake detection model can benefit from pre-trained representations and transfer learning techniques, allowing for efficient training and better generalization to unseen deepfake variations and datasets.

1.4. SCOPE

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

2. SRS DOCUMENT

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform.
This document lays out a project plan for the development of Deepfake video detection using neural network. The intended readers of this document are current and future developers working on Deepfake video detection using neural network and the sponsors of the project. The plan will include, but is not restricted to, a summary of the system functionality, the scope of the project from the perspective of the "Deepfake video detection" team (me and my mentors), use case diagram, Data flow diagram, Activity diagram, functional and non-functional requirements, project risks and how those risks will be mitigated, the process by which we will develop the project, and metrics and measurements that will be recorded throughout the project.

2.1 FUNCTIONAL REQUIREMENTS

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioural requirements describing all the cases where the system uses the functional requirements are captured in use cases. Functional requirements are supported by non- functional requirements (also known as quality requirements), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). How a system implements functional requirements is detailed in the system design. In some cases, a requirements analyst generates use cases after gathering and validating a set of functional requirements. Each use case illustrates behavioural scenarios through one or more functional requirements.

Functional requirements in a deepfake video detection project encompass the essential capabilities necessary for accurately identifying manipulated videos. These requirements include compatibility with various video formats, preprocessing to standardize input data, feature extraction for capturing subtle cues of manipulation, temporal analysis for detecting inconsistencies across frames, model training and evaluation tools, deployment mechanisms for real-time detection, scalability to handle large volumes of data, user-friendly interfaces, alerting mechanisms for notifying users of detected deepfakes, integration with existing platforms, customization options, robustness against evasion tactics, comprehensive documentation, and compliance with legal and ethical standards. Together, these requirements ensure that the deepfake detection system can effectively safeguard against the proliferation of deceptive content while providing users with reliable tools for identifying and mitigating the impact of manipulated videos.

2.2. NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements in a deepfake video detection project specify the qualities, constraints, and attributes that define how the system operates rather than what it does. These requirements are crucial for ensuring the overall performance, usability, security, and maintainability of the system. Some key non-functional requirements include:

Performance: The system should be capable of processing video data efficiently, with minimal latency and high throughput, to support real-time or near-real-time detection in large-scale environments.

Accuracy: The deepfake detection algorithms should achieve high accuracy rates in distinguishing between authentic and manipulated videos while minimizing false positives and false negatives.

Scalability: The system should be scalable to handle increasing volumes of video data and user requests, leveraging distributed computing architectures and cloud resources as needed.

Security: Implementing measures to protect against potential security threats, such as unauthorized access, data breaches, or tampering with detection results, to safeguard the integrity and confidentiality of video data and system components.

2.3. MINIMUM HARDWARE REQUIREMENTS

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

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

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

Table 2.1: Hardware Requirements

2.4. MINIMUM SOFTWARE REQUIREMENTS

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

3. ANALYSIS

3.1 EXISTING SYSTEMS AND DRAWBACKS

A. Face Warping Artifacts [15] used the approach to detect artifacts by comparing the generated face areas and their surrounding regions with a dedicated Convolutional Neural Network model.In this work there were two fold of Face Artifacts.
Their method is based on the observations that current deepfake algorithm can only generate images of limited resolutions, which are then needed to be further transformed to match the faces to be replaced in the source video. Their method has not considered the temporal analysis of the frames.

B. Detection by Eye Blinking [16] describes a new method for detecting the deepfakes by the eye blinking as a crucial parameter leading to classification of the videos as deepfake or pristine. The Long-term Recurrent Convolution Network (LRCN) was used for temporal analysis of the cropped frames of eye blinking. As today the deepfake generation algorithms have become so powerful that lack of eye blinking cannot be the only clue for detection of the deepfakes. There must be certain other parameters must be considered for the detection of deep- fakes like teeth enchantment, wrinkles on faces, wrong placement of eyebrows etc.

C. Capsule networks to detect forged images and videos [17] uses a method that uses a capsule network to detect forged, manipulated images and videos in different scenarios, like replay attack detection and computer-generated video detection.
In their method, they have used random noise in the training phase which is not a good option. Still the model performed beneficial in their dataset but may fail on real time data due to noise in training. Our method is proposed to be trained on noiseless and real time datasets.

D. Recurrent Neural Network [18] (RNN) for deepfake detection used the approach of using RNN for sequential processing of the frames along with ImageNet pre-trained model. Their process used the HOHO [19] dataset consisting of just 600 videos.

Their dataset consists small number of videos and same type of videos, which may not perform very well on the real time data. We will be training out model on large number of Real time data.

E. Synthetic Portrait Videos using Biological Signals [20] approach extract bio- logical signals from facial regions on pristine and deepfake portrait video pairs. Applied transformations to compute the spatial coherence and temporal consistency, capture the signal characteristics in feature vector and photoplethysmography (PPG) maps, and further train a probabilistic Support Vector Machine (SVM) and a Convolutional Neural Network (CNN). Then, the average of authenticity probabilities is used to classify whether the video is a deepfake or a pristine.

Fake Catcher detects fake content with high accuracy, independent of the generator, content, resolution, and quality of the video. Due to lack of discriminator leading to the loss in their findings to preserve biological signals, formulating a differentiable loss function that follows the proposed signal processing steps is not straight forward process.

3.2. PROPOSED ALGORITHM

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

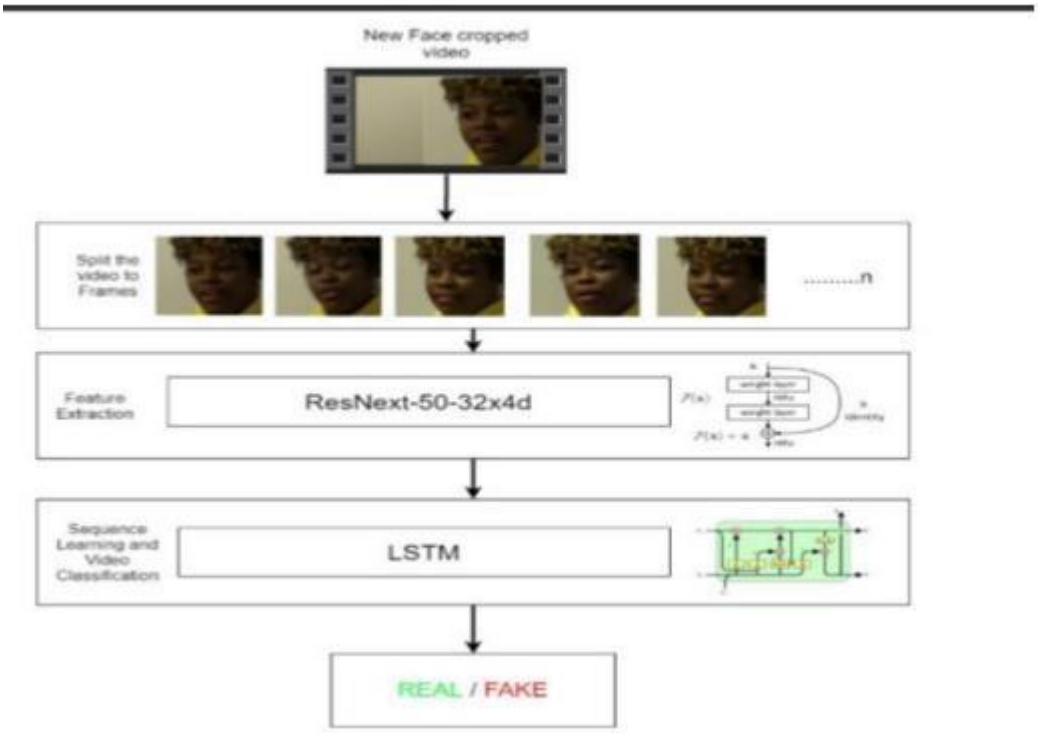
ResNext :

Instead of writing the code from scratch, we used the pre-trained model of ResNext for feature extraction. ResNext is Residual CNN network optimized for high performance on deeper neural networks. For the experimental purpose we have used resnext50_32x4d model. We have used a ResNext of 50 layers and 32 x 4 dimensions. Following, we will be fine-tuning the network by adding extra required layers and selecting a proper learning rate to properly converge the gradient descent of the model. The 2048-dimensional feature vectors after the last pooling layers of ResNext is used as the sequential LSTM input.

LSTM for Sequence Processing:

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

frames in a sequential manner so that the temporal analysis of the video can be made, by comparing the frame at ‘t’ second with the frame of ‘t-n’ seconds. Where n can be any number of frames before t. The model also consists of Leaky ReLu activation function. A linear layer of 2048 input features and 2 output features are used to make the model capable of learning the average rate of correlation between input and output. An adaptive average polling layer with the output parameter 1 is used in the model. Which gives the target output size of the image of the form H x W. For sequential processing of the frames a Sequential Layer is used. The batch size of 4 is used to perform the batch training. A Soft Max layer is used to get the confidence of the model during prediction



.

Figure 3.1: Overview of our model

3.3. FEASIBILITY STUDY

A feasibility study is an analysis that takes all a project's relevant factors into account including economic, technical, legal, and scheduling considerations to ascertain the likelihood of completing the project successfully. A feasibility study is important and essential to evolute any proposed project is feasible or not. A feasibility study is simply an assessment of the practicality of a proposed plan or project. The main objectives of feasibility are mentioned below: To determine if the product is technically and financially feasible to develop, is the main aim of the feasibility study activity. A feasibility study should provide management with enough information to decide

• Whether the project can be done.
• To determine how successful your proposed action will be.
• Whether the final product will benefit its intended users.
• To describe the nature and complexity of the project.
• What are the alternatives among which a solution will be chosen (During subsequent phases)
• To analyze if the software meets organizational requirements.
There are various types of feasibility that can be determined. They are:

Operational - Define the urgency of the problem and the acceptability of any solution, includes people-oriented and social issues: internal issues, such as manpower problems, labour objections, manager resistance, organizational conflicts, and policies; also, external issues, including social acceptability, legal aspects, and government regulations.

Technical - Is the feasibility within the limits of current technology? Does the technology exist at all? Is it available within a given resource?

Economic - Is the project possible, given resource constraints? Are the benefits that will accrue from the new system worth the costs? What are the savings that will result from the system, including tangible and intangible ones? What are the development and operational costs?

Schedule - Constraints on the project schedule and whether they could be reasonably met.

3.3.1. Economic Feasibility:

Economic analysis could also be referred to as cost/benefit analysis. It is the most frequently used method for evaluating the effectiveness of a new system. In economic analysis the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. Economic feasibility study related to price, and all kinds of expenditure related to the scheme before the project starts. This study also improves project reliability. It is also helpful for the decision-makers to decide the planned scheme processed latter or now, depending on the financial condition of the organization. This evaluation process also studies the price benefits of the proposed scheme. Economic Feasibility also performs the following tasks.

• Cost of packaged software.
• Cost of doing full system study.
• Is the system cost Effective?

3.3.2. Technical Feasibility: A large part of determining resources has to do with assessing

technical feasibility. It considers the technical requirements of the proposed project. The technical requirements are then compared to the technical capability of the organization. The systems project is considered technically feasible if the internal technical capability is sufficient to support the project requirements. The analyst must find out whether current technical resources can be where the expertise of system analysts is beneficial, since using their own experience and their contact with vendors they will be able to answer the question of technical feasibility.

3.3.3. Operational Feasibility:

Operational feasibility is a measure of how well a proposed system solves the problems and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. The operational feasibility refers to the availability of the operational resources needed to extend research results beyond on which they were developed and for which all the operational requirements are minimal and easily accommodated. In addition, the operational feasibility would include any rational compromises farmers make in adjusting the technology to the limited operational resources available to them. The operational Feasibility also perform the tasks like

- Does the current mode of operation provide adequate response time?
- Does the current of operation make maximum use of resources.
- Determines whether the solution suggested by the software development team is acceptable.
- Does the operation offer an effective way to control the data?
- Our project operates with a processor and packages installed are supported by the system

3.4 Cost Benefit Analysis:

The financial and the economic questions during the preliminary investigation are verified to estimate the following:

- The cost of the hardware and software for the class of application being considered.
- The benefits in the form of reduced cost.
- The proposed system will give the minute information, as a result.
- Performance is improved which in turn may be expected to provide increased profits.
- This feasibility checks whether the system can be developed with the available funds.
- This can be done economically if planned judicially, so it is economically feasible.
- The cost of the project depends upon the number of man-hours required.

4. SOFTWARE DESCRIPTION

4.1. VISUAL STUDIO CODE

Visual Studio Code is a lightweight but powerful source code editor developed by Microsoft, which runs on your desktop and is available for Windows, macOS and Linux. It features a robust ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, and.NET) and includes built-in support for TypeScript, JavaScript, and Node.js. with the interactive debugger built into Visual Studio Code, we can step through source code, examine variables, see call stacks, and run commands in the console. Additionally, VS Code interacts with build and scripting tools to expedite routine processes and streamline daily workflows. Because VS Code supports Git, you can work with source control - including examining pending changes diffs - without ever leaving the editor.

4.2. JUPYTER NOTEBOOK

Jupyter Notebook is an open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text. It supports various programming languages, including Python, R, and Julia, making it an ideal environment for data analysis, machine learning, and scientific research. With its interactive interface, users can execute code in a modular and flexible manner, making it easier to experiment and iterate on ideas. Jupyter Notebook promote reproducibility by enabling users to document their workflows step by step, facilitating collaboration and sharing among researchers and data scientists. Its integration with libraries like TensorFlow and Keras makes it a popular choice for developing and prototyping deep learning models due to its ability to visualize data and results seamlessly.

4.3. GOOGLE COLAB

Google Colab is a cloud-based platform that provides users with free access to GPU and TPU resources, allowing for the execution of Python code in an interactive Jupyter Notebook environment. Its integration with Google Drive facilitates seamless data management, while pre-installed libraries and customizable environments support various data science and machine learning tasks. Colab enables collaborative editing and sharing of notebooks, supports interactive visualization, Markdown formatting, and integrates with version control systems, making it an efficient and versatile platform for conducting research, developing machine learning models, and collaborating on data science projects.

4.4. PYTHON3

Python is a high-level, interpreted programming language known for its simplicity and readability. It offers extensive libraries and frameworks, making it versatile for various applications, including web development, data analysis, artificial intelligence, and scientific computing. Python's syntax emphasizes code readability and encourages

developers to write clean, maintainable code, enhancing collaboration and reducing development time. Its extensive ecosystem includes libraries like TensorFlow, Keras, and NumPy, which are essential for machine learning and deep learning tasks. Python's popularity stems from its ease of learning, broad community support, and cross-platform compatibility, making it a preferred choice for software development projects across different domains.

4.5. JAVASCRIPT

JavaScript is a high-level, interpreted programming language primarily used for front-end web development, although it's also increasingly utilized for back-end development and other applications. It enables developers to create dynamic and interactive web pages by adding behavior to HTML elements. JavaScript is known for its versatility, as it can run on both the client-side (in web browsers) and the server-side (using platforms like Node.js). It supports various programming paradigms including procedural, functional, and object-oriented programming, and it offers features such as event handling, DOM manipulation, asynchronous programming with Promises and async/await, and modularization with ES6 modules. JavaScript has a vast ecosystem of libraries and frameworks (e.g., React.js, Angular, Vue.js) that simplify web development tasks and facilitate building complex web applications. Additionally, JavaScript adheres to the ECMAScript standard, with new features and improvements regularly introduced in newer versions, ensuring its relevance and evolution within the programming community.

4.6. PYTORCH

PyTorch is an open-source machine learning framework developed primarily by Facebook's AI Research lab (FAIR). It is widely used for building and training deep learning models due to its flexibility, ease of use, and dynamic computation graph capabilities. PyTorch provides a Python-based interface that allows developers to define and manipulate computational graphs dynamically, making it intuitive for experimentation and model prototyping. Key features of PyTorch include automatic differentiation, which simplifies the process of computing gradients for optimizing neural network parameters, and a rich ecosystem of modules and utilities for tasks such as data loading, model building, and evaluation. PyTorch also supports GPU acceleration, enabling efficient training of deep neural networks on parallel hardware. Furthermore, PyTorch benefits from an active community contributing to its development, documentation, and the creation of various extensions and libraries, making it a popular choice among researchers and practitioners in the deep learning community.

4.7. DJANGO

Django is a high-level Python web framework that facilitates the rapid development of secure, scalable, and maintainable web applications. Built on the principles of DRY (Don't Repeat Yourself) and convention over configuration, Django follows the "batteries-included" philosophy, providing developers with a comprehensive set of tools and libraries for common web development tasks. Key features of Django include its

robust ORM (Object-Relational Mapping) system, which abstracts database interactions, its built-in authentication and authorization mechanisms for user management, and its powerful URL routing and view handling system. Django also includes a templating engine for generating dynamic HTML content and supports internationalization and localization. Additionally, Django emphasizes security best practices by providing protection against common web vulnerabilities such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). With its extensive documentation, active community, and ecosystem of reusable apps and packages, Django remains one of the most popular and widely-used web frameworks for building complex web applications.

4.8. GIT

Git is a distributed version control system used for tracking changes in source code during software development. It enables multiple developers to collaborate on projects simultaneously by managing revisions, branches, and merges efficiently. With Git, developers can work offline, commit changes locally, and synchronize their work with remote repositories, such as GitHub or GitLab, when they have internet access. Its branching and merging capabilities facilitate parallel development, allowing teams to work on different features or fixes concurrently without conflicts. Git's decentralized architecture provides redundancy and backup, ensuring the integrity and availability of project history. By adopting Git, developers can streamline collaboration, maintain code quality, and track the evolution of software projects effectively.

4.9. GITHUB

GitHub is a web-based platform built on top of Git, providing hosting services for software development projects. It offers features like code repository hosting, issue tracking, project management, and collaboration tools, making it a central hub for developers to work together on open-source and private projects. GitHub allows users to fork repositories, propose changes via pull requests, and review code collaboratively, fostering community-driven development and knowledge sharing. Its integration with continuous integration and deployment (CI/CD) pipelines enables automated testing and deployment workflows, ensuring code quality and reliability. GitHub's social features, such as stars, forks, and watches, promote project visibility and engagement within the developer community, facilitating contributions and feedback from users worldwide.

4.10. GOOGLE CLOUD PLATFORM

Google Cloud Platform (GCP) is a suite of cloud computing services provided by Google, offering a wide range of infrastructure and platform services for building, deploying, and managing applications and data in the cloud. GCP provides scalable and reliable computing resources, including virtual machines (Google Compute Engine), managed Kubernetes clusters (Google Kubernetes Engine), serverless computing (Google Cloud Functions), and data storage options such as relational databases (Cloud SQL), NoSQL databases (Cloud Firestore, Cloud Bigtable), and object storage (Cloud Storage). GCP also offers a comprehensive set of AI and machine learning services (Google Cloud AI) for tasks like image and speech recognition, natural language processing, and predictive analytics. With its global network of data centers and high-performance networking infrastructure, GCP ensures low-latency and high-availability

services for users worldwide. Additionally, GCP provides tools for monitoring, logging, and managing resources, as well as robust security features and compliance certifications to ensure data protection and regulatory compliance. Overall, Google Cloud Platform empowers businesses and developers to innovate and scale their applications efficiently in the cloud, leveraging Google's expertise and infrastructure to drive digital transformation and business growth.

5. PROJECT DESCRIPTION

5.1. PROBLEM DEFINITION

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

5.2. PROJECT OVERVIEW

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

Latest advances in the technology have changed the field of video manipulation. The advances in the modern open source deep learning frameworks likeTensorFlow, Keras, PyTorch along with cheap access to the high computationpower has driven the paradigm shift. The Conventional auto-encoders[10] and Generative Adversarial Network (GAN) pre-trained models have made the tampering of the realistic videos and images very easy. Moreover, access to these pre-trained 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 hairstyle, 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 [11,12] that generate deepfake videos are being extensively used to create fake celebrity pornographic videos or revenge porn [13]. Some of the examples are Brad Pitt, Angelina Jolie nude videos. The real looking nature of the deepfake videos makes the 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 [14]. Due to which it becomes very important to detect the deepfake videos and avoid the percolation of the deepfakes on the social media platforms**.**

- Task 1: Data-set gathering and analysis
  This task consists of downloading the dataset. Analysing the dataset and mak-ing the dataset ready for the preprocessing.

- Task 2 : Module 1 implementation
  Module 1 implementation consists of splitting the video to frames and cropping each frame consisting of face.

- Task 3: Pre-processing
  Pre-processing includes the creation of the new dataset which includesonly face cropped videos.

- Task 4: Module 2 implementation
  Module 2 implementation consists of implementation of Data Loader for loading the video and labels. Training a base line model on small amount ofdata.

- Task 5 : Hyper parameter tuning
  This task includes the changing of the Learning rate, batch size, weight decayand model architecture until the maximum accuracy is achieved.

- Task 6 : Training the final model
  The final model on large dataset is trained based on the best hyperparameter identified in the Task 5.

- Task 7 : Front end Development
  This task includes the front end development and integration of the back-end and front-end.

- Task 8 : Testing
  The complete application is tested using unit testing

5.2.1 SYSTEM ARCHITECTURE

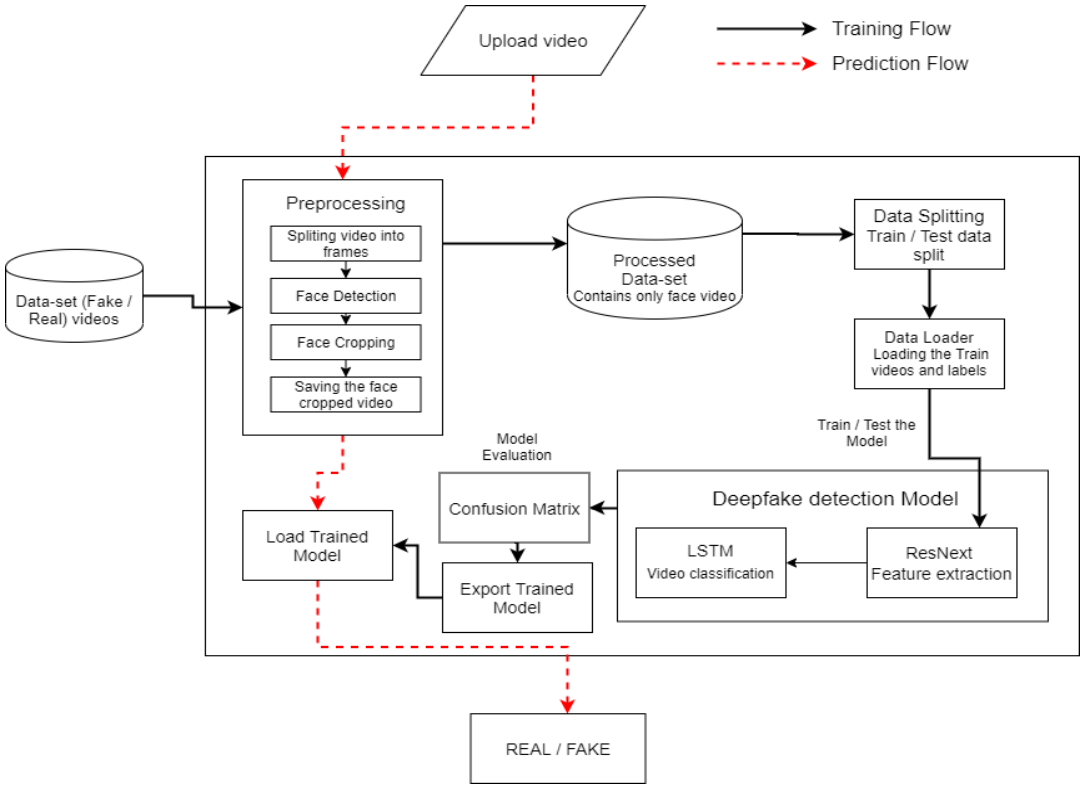


Figure 5.1: System Architecture

In this system, we have trained our PyTorch deepfake detection model on equal number of real and fake videos in order to avoid the bias in the model. The system architecture of the model is showed in the figure. In the development phase, we have taken a dataset, preprocessed the dataset and created a new processed dataset which only includes the face cropped videos.

5.2.1. CREATING DEEPFAKE VIDEO

To detect the deepfake videos it is very important to understand the creation process of the deepfake. Majority of the tools including the GAN and autoencoders takes a source image and target video as input. These tools split the video into frames , detect the face in the video and replace the source face with target face on each frame. Then the replaced frames are then combined using different pre-trained models. These models also enhance the quality of video by removing the left-over traces by the deepfake creation

model. Which result in creation of a deepfake looks realistic in nature. We have also used the same approach to detect the deepfakes. Deepfakes created using the pretrained neural networks models are very realistic that it is almost impossible to spot the difference by the naked eyes. But in reality, the deepfakes creation tools leaves some of the traces or artifacts in the video which may not be noticeable by the naked eyes. The motive of this paper to identify these unnoticeable traces and distinguishable artifacts of these videos and classified it as deepfake or real video.

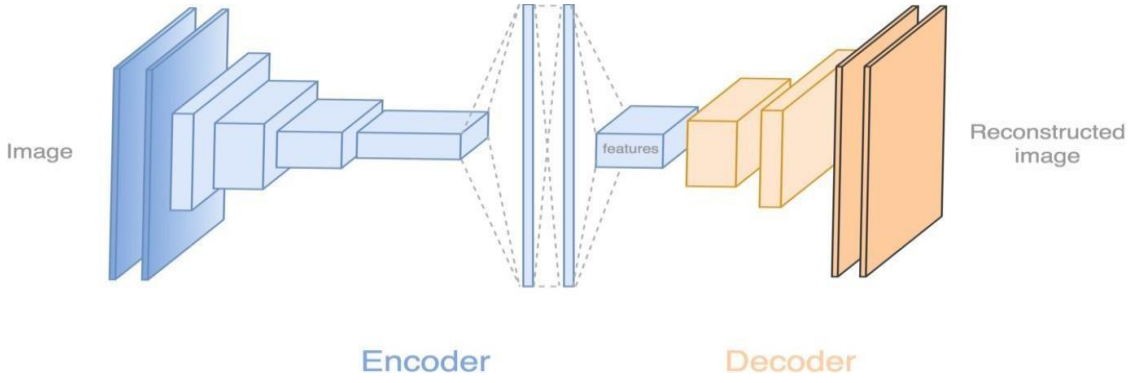


Figure 5.2: Deepfake generation

Figure 5.3: Face Swapped deepfake generation

5.2.2. ABOUT MODEL

The model consists of following layers:

• **ResNext CNN**: The pre-trained model of Residual Convolution Neural Network is used. The model name is resnext50_32x4d() [22]. This model consists of 50 layers and 32 x 4 dimensions. Figure shows the detailed implementation of model.

stage	output	ResNeXt-50 (32×4d)
conv1	112×112	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2 [1×1, 128 3×3, 128, C=32 1×1, 256] ×3
conv3	28×28	[1×1, 256 3×3, 256, C=32 1×1, 512] ×4
conv4	14×14	[1×1, 512 3×3, 512, C=32 1×1, 1024] ×6
conv5	7×7	[1×1, 1024 3×3, 1024, C=32 1×1, 2048] ×3
	1×1	global average pool 1000-d fc, softmax
# params.		$\mathbf{25.0}\times10^6$

Figure 5.1: ResNext Architecture

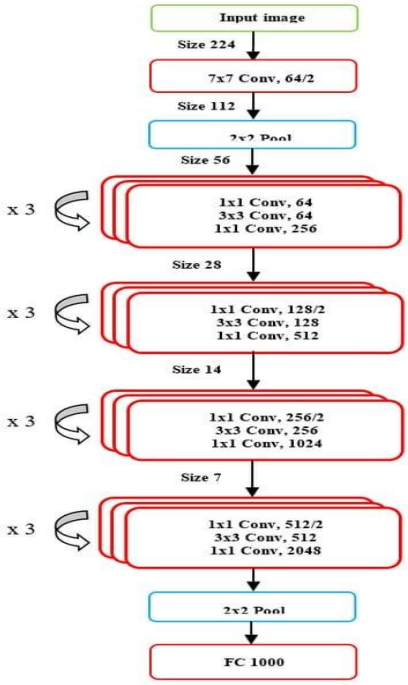


Figure 5.2: Overview of ResNext Architecture

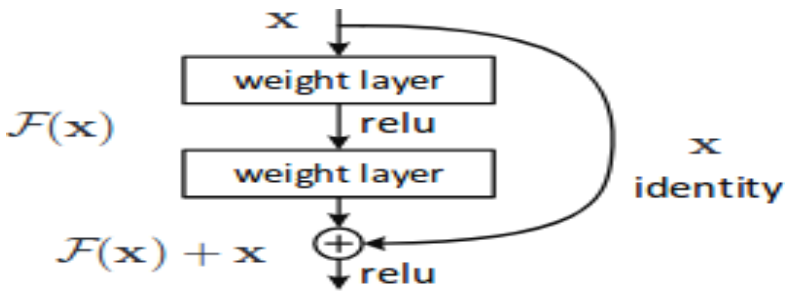


Figure 5.3: ResNext Working

• **Sequential Layer:** Sequential is a container of Modules that can be stacked together and run at the same time. Sequential layer is used to store feature vector returned by the ResNext model in a ordered way. So that it can be passed to the LSTM sequentially.

• **LSTM Layer:** LSTM is used for sequence processing and spot the temporal change between the frames.2048-dimensional feature vectors is fitted as the input to the LSTM. We are using 1 LSTM layer with 2048 latent dimensions and 2048 hidden layers along with 0.4 chance of dropout, which is capable to do achieve our objective. LSTM is used to process the frames in a sequential manner so that the temporal analysis of the video can be made, by comparing the frame at 't' second with the frame of 't-n' seconds. Where n can be any number of frames before t.

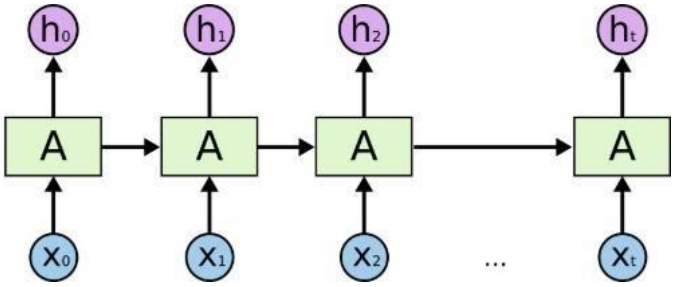


Figure 5.4: Overview of LSTM Architecture

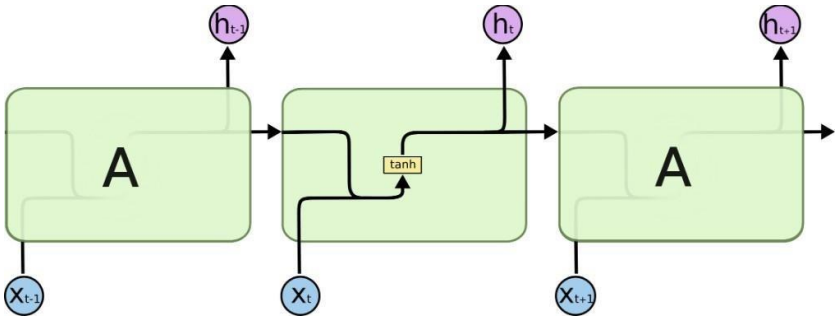


Figure 5.5: Internal LSTM Architecture

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

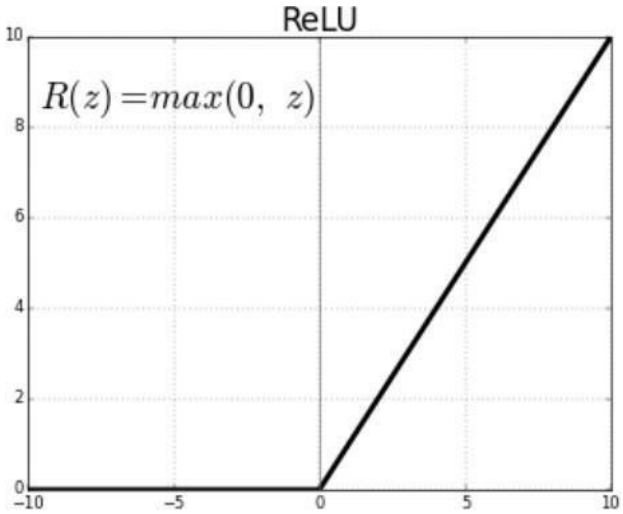


Figure 5.6: Relu Activation function

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

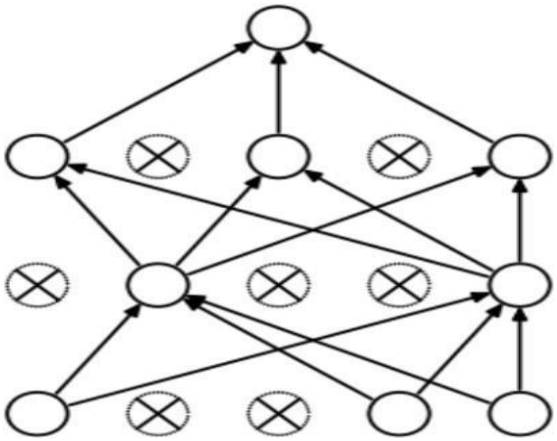


Figure 5.7: Dropout layer overview

• **Adaptive Average Pooling Layer:** It is used To reduce variance, reduce computation complexity and extract low level features from neighbourhood. 2 dimensional Adaptive Average Pooling Layer is used in the model.

• **Training:** The training is done for 20 epochs with a learning rate of 1e-5 (0.00001),weight decay of 1e-3 (0.001) using the Adam optimizer.

• **Adam optimizer[21]:** To enable the adaptive learning rate Adam optimizer with the model parameters is used.

• **Cross Entropy:** To calculate the loss function Cross Entropy approach is used because we are training a classification problem.

• **Softmax Layer:** A Softmax function is a type of squashing function. Squashing functions limit the output of the function into the range 0 to 1. This allows the output to be interpreted directly as a probability. Similarly, Softmax functions are multi-class sigmoids, meaning they are used in determining probability of multiple classes at once. Since the outputs of a Softmax function can be interpreted as a probability (i.e. they must sum to 1), a softmax layer is typically the final layer used in neural network functions. It is important to note that a softmax layer must have the same number of nodes as the output later. In our case softmax layer has two output nodes i.e REAL or FAKE, also Softmax layer provide us the confidence(probability) of prediction.

• **Confusion Matrix:** A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which your classification model is confused when it makes predictions. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made. Confusion matrix is used to evaluate our model and calculate the accuracy.

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

6 . SYSTEM DESIGN

6.1. Introduction to UML

Unified Modelling Language (UML) is a general-purpose modelling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite like blueprints used in other fields of engineering. UML is not a programming language; it is rather a visual language. We use UML diagrams to portray the behaviour and structure of a system. UML helps software engineers, businessmen and system architects with modelling, design and analysis. The Object Management Group (OMG) adopted Unified Modelling Language as a standard in 1997. It's been managed by OMG ever since. International Organization for Standardization (ISO) published UML as an approved standard in 2005. UML has been revised over the years and is reviewed periodically.

Why we need UML

• Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.

• Businessmen do not understand code. So, UML becomes essential to communicate with non programmers essential requirements, functionalities and processes of the system.

• A lot of time is saved down the line when teams can visualize processes, user interactions and static structure of the system.

UML is linked with object-oriented design and analysis. UML makes the useof elements and forms associations between them to form diagrams.
Diagrams in UMLcan be broadly classified as:

• Structural Diagrams – Capture static aspects or structure of a system. Structural Diagrams include Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.

• Behaviour Diagrams – Capture dynamic aspects or behaviour of the system. Behaviour diagrams include Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

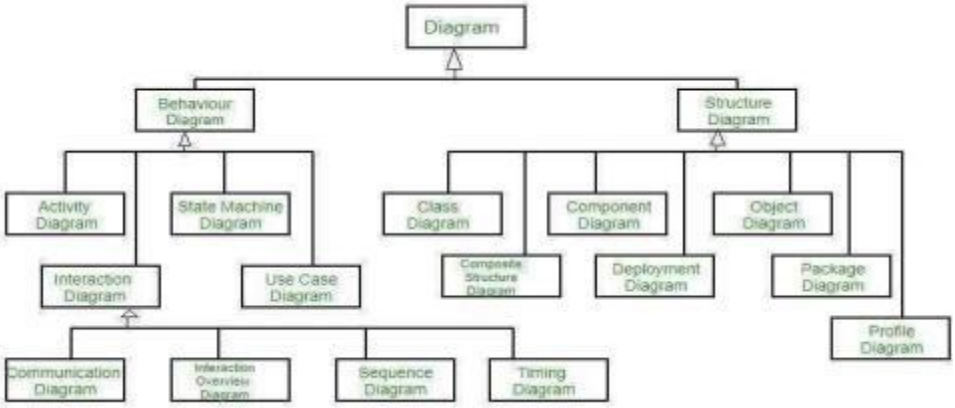


Figure 6.1. Building Blocks in UML

6.2 Building Blocks of the UML

The vocabulary of the UML encompasses three kinds of building blocks:
- Things
- Relationships
- Diagrams

Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

Things in the UML

There are four kinds of things in the UML:
- Structural things
- Behavioural things
- Grouping things
- Annotational things

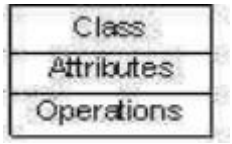
These things are the basic object-oriented building blocks of the UML. You use them to write well-formed models.

Structural Things

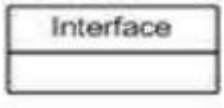
Structural things are the nouns of UML models. These are the mostly static parts of a model, representing elements that are either conceptual or physical.Collectively, the structural things are called classifiers.

A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics. A class implements one or more interfaces. Graphically, a class is rendered as a rectangle, usually including its name, attributes, and operations.

Class - A Class is a set of identical things that outlines the functionality and propertiesof an object. It also represents the abstract class whose functionalities are not defined.Its notation is as follows.



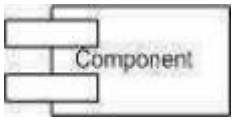
Interface - A collection of functions that specify a service of a class or component, i.e. , Externally visible behavior of that class.



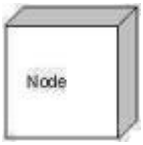
Collaboration - A larger pattern of behaviors and actions. Example: All classes and behaviors that create the modeling of a moving tank in a simulation.

Use Case - A sequence of actions that a system performs that yields an observable result. Used to structure behavior in a model. Is realized by collaboration

Component - A physical and replaceable part of a system that implements a number of interfaces. Example: a set of classes, interfaces, and collaborations.



Node - A physical element existing at run time and represents are source.



Behavioral Things

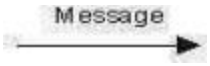
Behavioral things are the dynamic parts of UML models. These are the verbs of a model, representing behavior over time and space. In all, there are three primary kinds of behavioral things

• Interaction

• State Machine

Interaction

It is a behavior that comprises a set of messages exchanged among a set of objects or roles within a particular context to accomplish a specific purpose. The behavior of a society of objects or of an individual operation may be specified with an interaction. An interaction involves a number of other elements, including messages, actions, and connectors (the connection between objects). Graphically, a message is rendered as a directed line, almost always including the name of its operation.

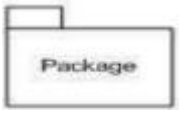


State Machine

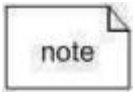
State machine is a behavior that specifies the sequences of states an object or an interaction goes through during its lifetime in response to events, together with its responses to those events. The behavior of an individual class or a collaboration of classes may be specified with a state machine. A state machine involves a number of other elements, including states, transitions (the flow from state to state), events (things that trigger a transition), and activities (the response to a transition). Graphically, a state is rendered as a rounded rectangle, usually including its name and its substates.

Grouping things can be defined as a mechanism to group elements of a UMLmodel together. There is only one grouping thing available.

Package − Package is the only one grouping thing available for gathering structuraland behavioural things.



Annotational Things -Annotational things are the explanatory parts of UML models. These are the comments you may apply to describe, illuminate, and remark about any element in a model. There is one primary kind of annotational thing, called a note. A note is simply a symbol for rendering constraints and comments attached to an element or a collection of elements.



Relationships in the UML

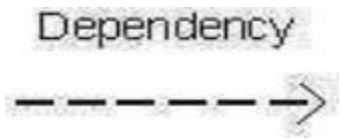
Relationship is another most important building block of UML. It shows the elements are associated with each other and this association describes the functionality of an application.

There are four kinds of relationships in the UML:

- Dependency
- Association
- Generalization
- Realization

Dependency

It is an element (the independent one) that may affect the semantics of the other element (the dependent one). Graphically, a dependency is rendered as a dashed line, possibly directed, and occasionally including a label.

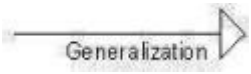


Association

Association is basically a set of links that connects the elements of a UML model. It also describes how many objects are taking part in that relationship**.**

Generalization

It is a specialization/generalization relationship in which the specialized element (the child) builds on the specification of the generalized element (the parent).The child shares the structure and the behavior of the parent. Graphically, a generalization relationship is rendered as a solid line with a hollow arrowhead pointing to the parent.



Realization

Realization can be defined as a relationship in which two elements are connected. One element describes some responsibility, which is not implemented and the other one implements them. This relationship exists in case of interfaces.

6.3 UML Diagrams

UML is a modern approach to modeling and documenting software. It is based on diagrammatic representations of software components. It is the final output,and the diagram represents the system.

UML includes the following

- Class diagram
- Object diagram
- Component diagram
- Composite structure diagram
- Use case diagram
- Sequence diagram
- Communication diagram
- State diagram
- Activity diagram

Use Case Diagram

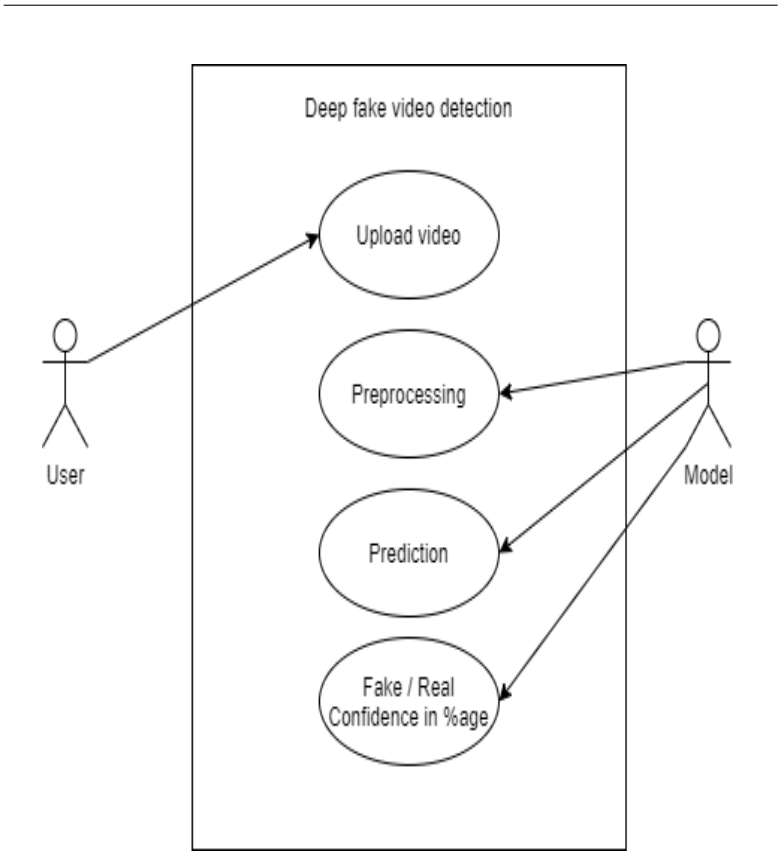


Figure 6.1: Use case diagram

Sequence Diagram

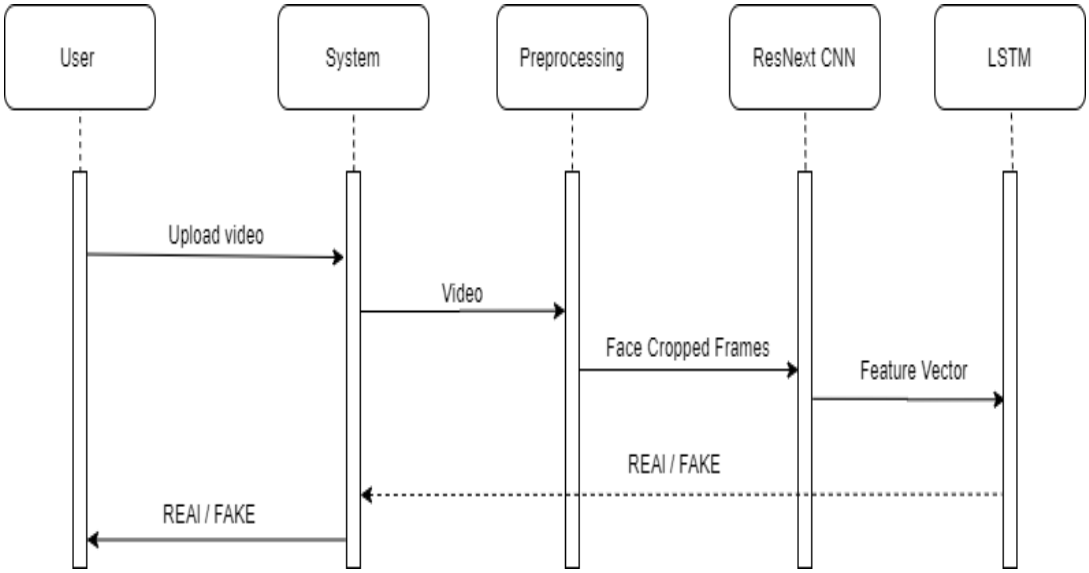


Figure 6.2: Sequence Diagram

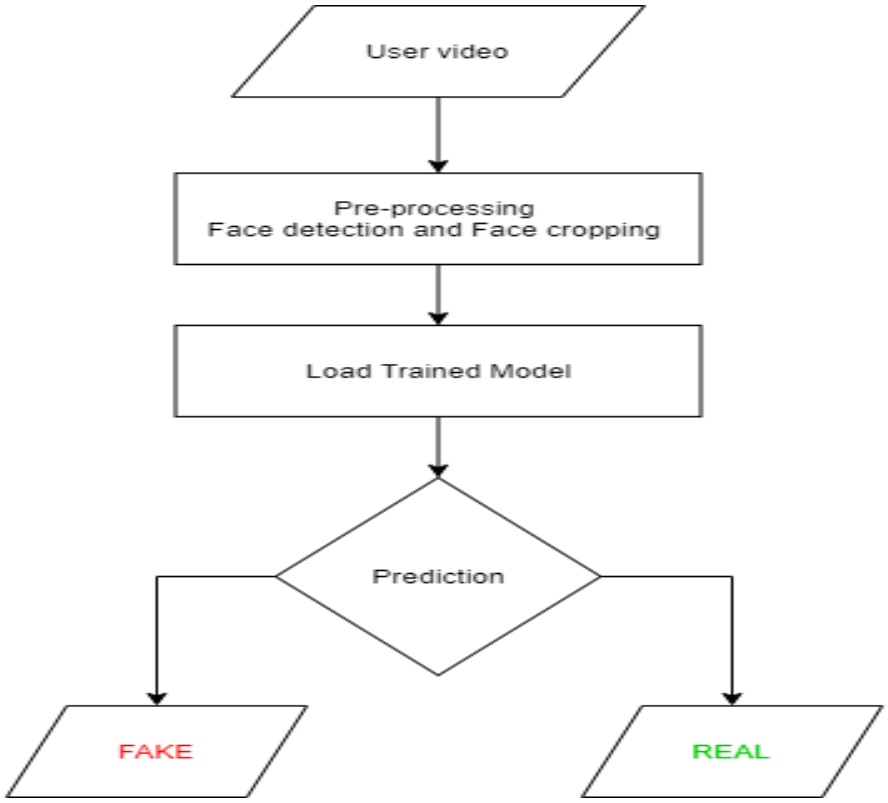


Figure 6.3: Testing Workflow(Activity Diagram)

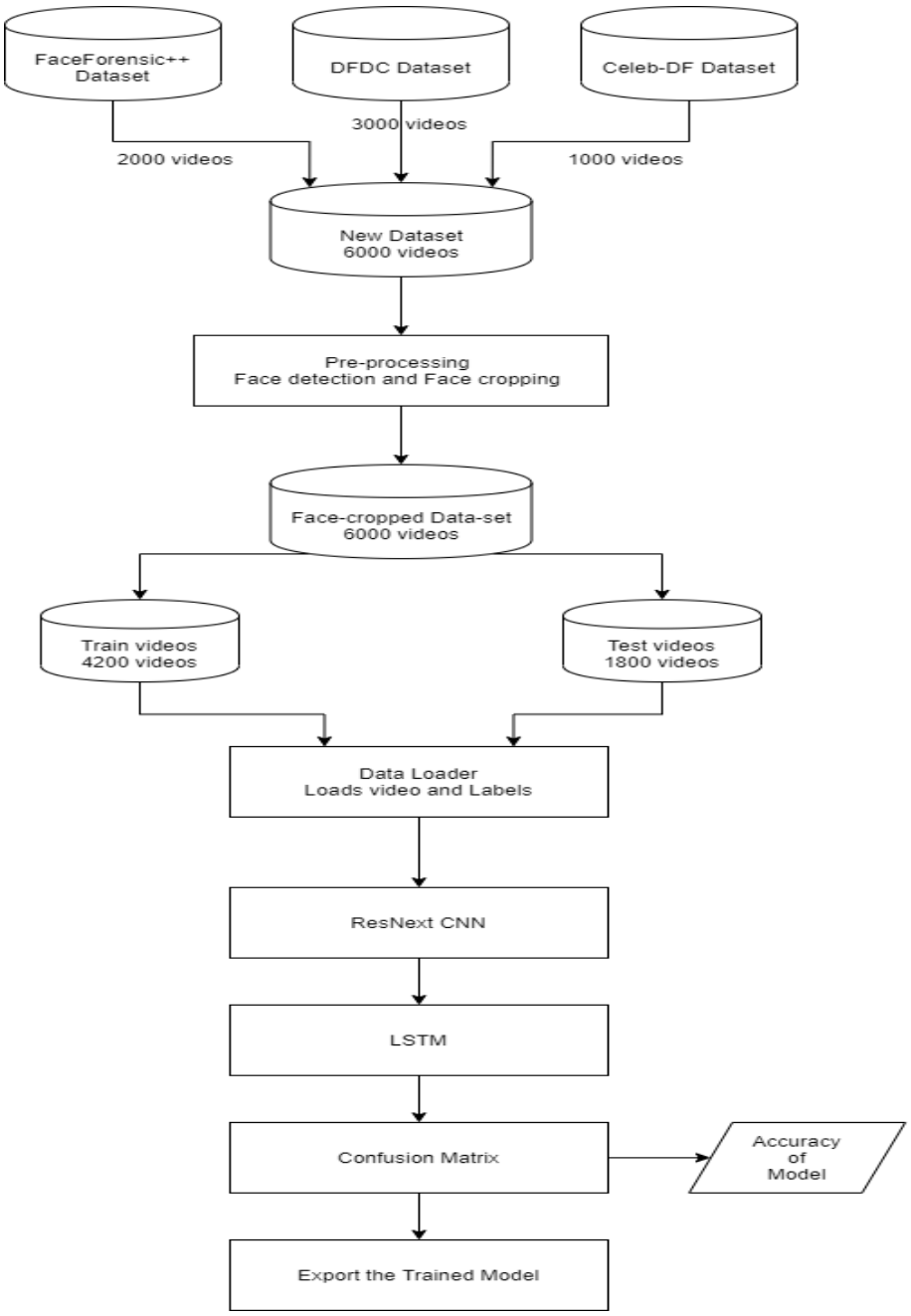


Figure 6.4: Training Workflow(Activity Diagram)

7. DEVELOPMENT

7.1. DATASETS USED

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

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

Name of Dataset	Number of videos trained to model	
	Real	Fake
Deep fake detection challenge	1500	1500
FaceForensic++	1000	1000
Celeb-DF	500	500

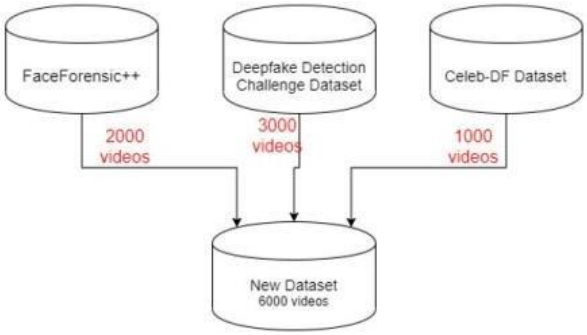


Figure 7.1: Dataset

7.2. SAMPLE CODE

The execution of the model was done in Google Colab.

1. **Loading the Data Set From Google Drive**:

```
from google.colab import drive
drive.mount('/content/drive')
```

Output:

Mounted at /content/drive

2. **Checking the Number Of Frames in Each Video**

```
import json
import glob
import numpy as np
import cv2
import copy
#change the path accordingly
video_files = glob.glob('/content/Real videos/*.mp4')
#video_files1 = glob.glob('/content/dfdc_train_part_0/*.mp4')
#video_files += video_files1
frame_count = []
for video_file in video_files:
  cap = cv2.VideoCapture(video_file)
  if(int(cap.get(cv2.CAP_PROP_FRAME_COUNT))<150):
  video_files.remove(video_file)
    continue
frame_count.append(int(cap.get(cv2.CAP_PROP_FRAME_COUNT)))
print("frames" , frame_count)
print("Total number of videos: " , len(frame_count))
print('Average frame per video:',np.mean(frame_count))
```

output: frames [300, 299, 300, 299, 300, 299, 300, 299, 300, 299]

Total number of videos: 10

Average frame per video: 299.5

3. CROPPING THE FACE ONLY AREA FROM EACH VIDEO

```
 # to extract frame
def frame_extract(path):
  vidObj = cv2.VideoCapture(path)
  success = 1
  while success:
      success, image = vidObj.read()
      if success:
          yield image
!pip3 install face_recognition
!mkdir '/content/drive/My Drive/FF_REAL_Face_only_data'
import torch
import torchvision
from torchvision import transforms
from torch.utils.data import DataLoader
from torch.utils.data.dataset import Dataset
import os
import numpy as np
import cv2
import matplotlib.pyplot as plt
import face_recognition
from tqdm.autonotebook import tqdm
# process the frames
def create_face_videos(path_list,out_dir):
  already_present_count =  glob.glob(out_dir+'*.mp4')
  print("No of videos already present " , len(already_present_count))
  for path in tqdm(path_list):
    out_path = os.path.join(out_dir,path.split('/')[-1])
    file_exists = glob.glob(out_path)
    if(len(file_exists) != 0):
      print("File Already exists: " , out_path)
```

```
    continue
  frames = []
  flag = 0
  face_all = []
  frames1 = []
  out = cv2.VideoWriter(out_path,cv2.VideoWriter_fourcc('M','J','P','G'), 30, (112,112))
  for idx,frame in enumerate(frame_extract(path)):
    #if(idx % 3 == 0):
    if(idx <= 150):
      frames.append(frame)
      if(len(frames) == 4):
        faces = face_recognition.batch_face_locations(frames)
        for i,face in enumerate(faces):
          if(len(face) != 0):
            top,right,bottom,left = face[0]
          try:
            out.write(cv2.resize(frames[i][top:bottom,left:right,:],(112,112)))
          except:
            pass
        frames = []
  try:
    del top,right,bottom,left
  except:
    pass
  out.release()
```

4. STORING THE CROPED VIDEOS INTO ANOTHER FILE

```
create_face_videos(video_files,'/content/drive/MyDrive/FF_REAL_Face_only_data/')
```

5. Check If The Video Is Corrupted Or Not. If The Video Is Corrupted Delete The Video:

```
#This code is to check if the video is corrupted or not..
#If the video is corrupted delete the video.
import glob
import torch
import torchvision
from torchvision import transforms
from torch.utils.data import DataLoader
```

```
from torch.utils.data.dataset import Dataset
import os
import numpy as np
import cv2
import matplotlib.pyplot as plt
import face_recognition
#Check if the file is corrupted or not
def validate_video(vid_path,train_transforms):
    transform = train_transforms
    count = 20
    video_path = vid_path
    frames = []
    a = int(100/count)
    first_frame = np.random.randint(0,a)
    temp_video = video_path.split('/')[-1]
    for i,frame in enumerate(frame_extract(video_path)):
      frames.append(transform(frame))
      if(len(frames) == count):
        break
    frames = torch.stack(frames)
    frames = frames[:count]
    return frames
#extract a from from video
def frame_extract(path):
  vidObj = cv2.VideoCapture(path)
  success = 1
  while success:
      success, image = vidObj.read()
      if success:
          yield image

im_size = 112
mean = [0.485, 0.456, 0.406]
std = [0.229, 0.224, 0.225]

train_transforms = transforms.Compose([
                                transforms.ToPILImage(),
```

```
                                transforms.Resize((im_size,im_size)),
                                transforms.ToTensor(),
                                transforms.Normalize(mean,std)])
video_fil = glob.glob('/content/drive/My Drive/Celeb_fake_face_only/*.mp4')
video_fil += glob.glob('/content/drive/My Drive/Celeb_real_face_only/*.mp4')
video_fil += glob.glob('/content/drive/My Drive/DFDC_FAKE_Face_only_data/*.mp4')
video_fil += glob.glob('/content/drive/My Drive/DFDC_REAL_Face_only_data/*.mp4')
video_fil += glob.glob('/content/drive/My Drive/FF_Face_only_data/*.mp4')
print("Total no of videos :" , len(video_fil))
print(video_fil)
count = 0;
for i in video_fil:
  try:
    count+=1
  validate_video(i,train_transforms)
  except:
    print("Number of video processed: " , count ," Remaining : " , (len(video_fil) - count))
    print("Corrupted video is : " , i)
    continue
print((len(video_fil) - count))
```

output: Total no of videos : 10

```
['/content/drive/MyDrive/FF_REAL_Face_only_data/ercqmajdid.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/blszgmxkvu.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/eyguqfmgzh.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/lnuwkizkiw.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/hypozprqhm.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/mkzaekkvej.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/uubgqnvfdl.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/nfsztvjqpk.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/wixbuuzygv.mp4',
'/content/drive/MyDrive/FF_REAL_Face_only_data/xrhqtmxlvx.mp4']
0
```

6. Function that acess Global meta dataset .csv file and label the video whether it is real or fake :

```
 #count the number of fake and real videos
def number_of_real_and_fake_videos(data_list):
  header_list = ["file","label"]
```

```
  lab = pd.read_csv('/content/drive/My Drive/Gobal_metadata.csv',names=header_list)
  fake = 0
  real = 0
  for i in data_list:
    temp_video = i.split('/')[-1]
    label = lab.iloc[(labels.loc[labels["file"] == temp_video].index.values[0]),1]
    if(label == 'FAKE'):
      fake+=1
    if(label == 'REAL'):
      real+=1
  return real,fake
```

7. Splitting the videos into Training and validation section ,also checking the cropped video

```
# load the labels and video in data loader
import random
import pandas as pd
from sklearn.model_selection import train_test_split

header_list = ["file","label"]
labels = pd.read_csv('/content/drive/My Drive/Gobal_metadata.csv',names=header_list)
#print(labels)
train_videos = video_files[:int(0.8*len(video_files))]
valid_videos = video_files[int(0.8*len(video_files)):]
print("train : " , len(train_videos))
print("test : " , len(valid_videos))
# train_videos,valid_videos = train_test_split(data,test_size = 0.2)
# print(train_videos)

print("TRAIN: ", "Real:",number_of_real_and_fake_videos(train_videos)[0],"
Fake:",number_of_real_and_fake_videos(train_videos)[1])
print("TEST: ", "Real:",number_of_real_and_fake_videos(valid_videos)[0],"
Fake:",number_of_real_and_fake_videos(valid_videos)[1])


im_size = 112
mean = [0.485, 0.456, 0.406]
std = [0.229, 0.224, 0.225]
```

```
train_transforms = transforms.Compose([
                        transforms.ToPILImage(),
                        transforms.Resize((im_size,im_size)),
                        transforms.ToTensor(),
                        transforms.Normalize(mean,std)])

test_transforms = transforms.Compose([
                        transforms.ToPILImage(),
                        transforms.Resize((im_size,im_size)),
                        transforms.ToTensor(),
                        transforms.Normalize(mean,std)])
train_data = video_dataset(train_videos,labels,sequence_length = 10,transform = train_transforms)
#print(train_data)
val_data = video_dataset(valid_videos,labels,sequence_length = 10,transform = train_transforms)
train_loader = DataLoader(train_data,batch_size = 4,shuffle = True,num_workers = 4)
valid_loader = DataLoader(val_data,batch_size = 4,shuffle = True,num_workers = 4)
image,label = train_data[0]
im_plot(image[0,:,:,:])
```

output:

TRAIN : 8 TEST : 2

TRAIN: Real: 4 Fake: 4

TEST: Real: 1 Fake:1

8. Functions of Resnext Model ,Training epoch,Testing epoch,Confusion matrix,Graphs of Loss and Accuracy:

```
#Model with feature visualization
from torch import nn
```

```
from torchvision import models
class Model(nn.Module):
    def__init_(self, num_classes,latent_dim= 2048, lstm_layers=1 , hidden_dim = 2048,
bidirectional = False):
        super(Model, self).__init__()
        model = models.resnext50_32x4d(pretrained = True) #Residual Network CNN
        self.model = nn.Sequential(*list(model.children())[:-2])
        self.lstm = nn.LSTM(latent_dim,hidden_dim, lstm_layers, bidirectional)
        self.relu = nn.LeakyReLU()
        self.dp = nn.Dropout(0.4)
        self.linear1 = nn.Linear(2048,num_classes)
        self.avgpool = nn.AdaptiveAvgPool2d(1)
    def forward(self, x):
        batch_size,seq_length, c, h, w = x.shape
        x = x.view(batch_size * seq_length, c, h, w)
        fmap = self.model(x)
        x = self.avgpool(fmap)
        x = x.view(batch_size,seq_length,2048)
        x_lstm,_ = self.lstm(x,None)
        return fmap,self.dp(self.linear1(torch.mean(x_lstm,dim = 1)))
model = Model(2).cuda()
a,b = model(torch.from_numpy(np.empty((1,20,3,112,112))).type(torch.cuda.FloatTensor))
import torch
from torch.autograd import Variable
import time
import os
import sys
import os
def train_epoch(epoch, num_epochs, data_loader, model, criterion, optimizer):
    model.train()
    losses = AverageMeter()
    accuracies = AverageMeter()
    t = []
    for i, (inputs, targets) in enumerate(data_loader):
        if torch.cuda.is_available():
            targets = targets.type(torch.cuda.LongTensor)
            inputs = inputs.cuda()
```

```
        _,outputs = model(inputs)
        loss  = criterion(outputs,targets.type(torch.cuda.LongTensor))
        acc = calculate_accuracy(outputs, targets.type(torch.cuda.LongTensor))
        losses.update(loss.item(), inputs.size(0))
        accuracies.update(acc, inputs.size(0))
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        sys.stdout.write(
            "\r[Epoch %d/%d] [Batch %d / %d] [Loss: %f, Acc: %.2f%%]"
            % (
                epoch,
                num_epochs,
                i,
                len(data_loader),
                losses.avg,
                accuracies.avg))
    torch.save(model.state_dict(),'/content/checkpoint.pt')
    return losses.avg,accuracies.avg
def test(epoch,model, data_loader ,criterion):
    print('Testing')
    model.eval()
    losses = AverageMeter()
    accuracies = AverageMeter()
    pred = []
    true = []
    count = 0
    with torch.no_grad():
        for i, (inputs, targets) in enumerate(data_loader):
            if torch.cuda.is_available():
                targets = targets.cuda().type(torch.cuda.FloatTensor)
                inputs = inputs.cuda()
            _,outputs = model(inputs)
            loss = torch.mean(criterion(outputs, targets.type(torch.cuda.LongTensor)))
            acc = calculate_accuracy(outputs,targets.type(torch.cuda.LongTensor))
            _,p = torch.max(outputs,1)
```

```
            true +=
(targets.type(torch.cuda.LongTensor)).detach().cpu().numpy().reshape(len(targets)).tolist()
            pred += p.detach().cpu().numpy().reshape(len(p)).tolist()
            losses.update(loss.item(), inputs.size(0))
            accuracies.update(acc, inputs.size(0))
            sys.stdout.write(
                "\r[Batch %d / %d]  [Loss: %f, Acc: %.2f%%]"
                % (
                    i,
                    len(data_loader),
                    losses.avg,
                    accuracies.avg
                    )
                )
        print('\nAccuracy {}'.format(accuracies.avg))
    return true,pred,losses.avg,accuracies.avg
class AverageMeter(object):
    """Computes and stores the average and current value"""
    def__init_(self):
        self.reset()
    def reset(self):
        self.val = 0
        self.avg = 0
        self.sum = 0
        self.count = 0

    def update(self, val, n=1):
        self.val = val
        self.sum += val * n
        self.count += n
        self.avg = self.sum / self.count
def calculate_accuracy(outputs, targets):
    batch_size = targets.size(0)

    _, pred = outputs.topk(1, 1, True)
    pred = pred.t()
    correct = pred.eq(targets.view(1, -1))
```

```
    n_correct_elems = correct.float().sum().item()
    return 100* n_correct_elems / batch_size
import seaborn as sn
#Output confusion matrix
def print_confusion_matrix(y_true, y_pred):
    cm = confusion_matrix(y_true, y_pred)
    print('True positive = ', cm[0][0])
    print('False positive = ', cm[0][1])
    print('False negative = ', cm[1][0])
    print('True negative = ', cm[1][1])
    print('\n')
    df_cm = pd.DataFrame(cm, range(2), range(2))
    sn.set(font_scale=1.4) # for label size
    sn.heatmap(df_cm, annot=True, annot_kws={"size": 16}) # font size
    plt.ylabel('Actual label', size = 20)
    plt.xlabel('Predicted label', size = 20)
    plt.xticks(np.arange(2), ['Fake', 'Real'], size = 16)
    plt.yticks(np.arange(2), ['Fake', 'Real'], size = 16)
    plt.ylim([2, 0])
    plt.show()
    calculated_acc = (cm[0][0]+cm[1][1])/(cm[0][0]+cm[0][1]+cm[1][0]+ cm[1][1])
    print("Calculated Accuracy",calculated_acc*100)
def plot_loss(train_loss_avg,test_loss_avg,num_epochs):
  loss_train = train_loss_avg
  loss_val = test_loss_avg
  print(num_epochs)
  epochs = range(1,num_epochs+1)
  plt.plot(epochs, loss_train, 'g', label='Training loss')
  plt.plot(epochs, loss_val, 'b', label='validation loss')
  plt.title('Training and Validation loss')
  plt.xlabel('Epochs')
  plt.ylabel('Loss')
  plt.legend()
  plt.show()
def plot_accuracy(train_accuracy,test_accuracy,num_epochs):
  loss_train = train_accuracy
  loss_val = test_accuracy
```

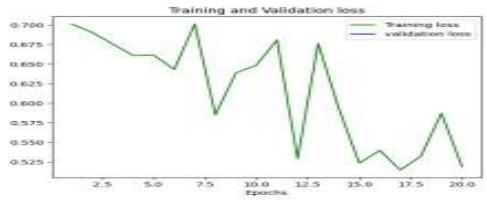
```
  epochs = range(1,num_epochs+1)
  plt.plot(epochs, loss_train, 'g', label='Training accuracy')
  plt.plot(epochs, loss_val, 'b', label='validation accuracy')
  plt.title('Training and Validation accuracy')
  plt.xlabel('Epochs')
  plt.ylabel('Accuracy')
  plt.legend()
  plt.show()
from sklearn.metrics import confusion_matrix
#learning rate
lr = 1e-5#0.001
#number of epochs
num_epochs = 20

optimizer = torch.optim.Adam(model.parameters(), lr= lr,weight_decay = 1e-5)

#class_weights = torch.from_numpy(np.asarray([1,15])).type(torch.FloatTensor).cuda()
#criterion = nn.CrossEntropyLoss(weight = class_weights).cuda()
criterion = nn.CrossEntropyLoss().cuda()
train_loss_avg =[]
train_accuracy = []
test_loss_avg = []
test_accuracy = []
for epoch in range(1,num_epochs+1):
    l, acc = train_epoch(epoch,num_epochs,train_loader,model,criterion,optimizer)
    train_loss_avg.append(l)
    train_accuracy.append(acc)
    true,pred,tl,t_acc = test(epoch,model,valid_loader,criterion)
    test_loss_avg.append(tl)
    test_accuracy.append(t_acc)
plot_loss(train_loss_avg,test_loss_avg,len(train_loss_avg))
plot_accuracy(train_accuracy,test_accuracy,len(train_accuracy))
print(confusion_matrix(true,pred))
print_confusion_matrix(true,pred)
```

output:

```
 [Epoch 1/20] [Batch 1 / 2] [Loss: 0.701347, Acc: 50.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
```

Accuracy 50.0
[Epoch 2/20] [Batch 1 / 2] [Loss: 0.690577, Acc: 50.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 3/20] [Batch 1 / 2] [Loss: 0.675753, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 4/20] [Batch 1 / 2] [Loss: 0.660965, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0 39
[Epoch 5/20] [Batch 1 / 2] [Loss: 0.661423, Acc: 62.50%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 6/20] [Batch 1 / 2] [Loss: 0.643254, Acc: 62.50%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 7/20] [Batch 1 / 2] [Loss: 0.701372, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 8/20] [Batch 1 / 2] [Loss: 0.585102, Acc: 62.50%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 9/20] [Batch 1 / 2] [Loss: 0.638605, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 10/20] [Batch 1 / 2] [Loss: 0.648571, Acc: 50.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 11/20] [Batch 1 / 2] [Loss: 0.680937, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 12/20] [Batch 1 / 2] [Loss: 0.528896, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 13/20] [Batch 1 / 2] [Loss: 0.676286, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0

[Epoch 14/20] [Batch 1 / 2] [Loss: 0.593668, Acc: 100.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 15/20] [Batch 1 / 2] [Loss: 0.523310, Acc: 100.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 16/20] [Batch 1 / 2] [Loss: 0.539514, Acc: 87.50%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 17/20] [Batch 1 / 2] [Loss: 0.514449, Acc: 100.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 18/20] [Batch 1 / 2] [Loss: 0.532006, Acc: 100.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 19/20] [Batch 1 / 2] [Loss: 0.587142, Acc: 75.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
[Epoch 20/20] [Batch 1 / 2] [Loss: 0.518556, Acc: 100.00%]Testing
[Batch 0 / 1] [Loss: nan, Acc: 50.00%]
Accuracy 50.0
2



[[1 0]
[1 0]]
True positive = 1
False positive = 0
False negative = 1
True negative = 0

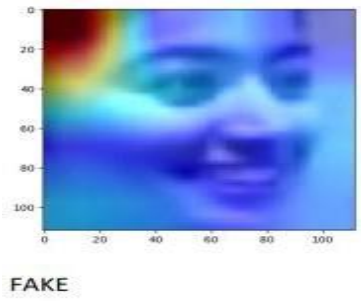
Calculated Accuracy 50.0

9. Code to make Predictions:

```
#Code for making prediction
im_size = 112
mean=[0.485, 0.456, 0.406]
std=[0.229, 0.224, 0.225]

train_transforms = transforms.Compose([
                                transforms.ToPILImage(),
                                transforms.Resize((im_size,im_size)),
                                transforms.ToTensor(),
                                transforms.Normalize(mean,std)])
path_to_videos = ['/content/drive/My Drive/Balanced_Face_only_data/aagfhgtpmv.mp4',
                            '/content/drive/My Drive/Balanced_Face_only_data/aczrgyricp.mp4',
                            '/content/drive/My Drive/Balanced_Face_only_data/agdkmztvby.mp4',
                            '/content/drive/My Drive/Balanced_Face_only_data/abarnvbtwb.mp4']

path_to_videos = ['/content/drive/My Drive/Youtube_Face_only_data/000_003.mp4',
                '/content/drive/My Drive/Youtube_Face_only_data/000.mp4',
                '/content/drive/My Drive/Youtube_Face_only_data/002_006.mp4',
                '/content/drive/My Drive/Youtube_Face_only_data/002.mp4'



]

path_to_videos= ["/content/drive/My Drive/DFDC_REAL_Face_only_data/aabqyygbaa.mp4"]

video_dataset = validation_dataset(path_to_videos,sequence_length = 20,transform =
train_transforms)
model = Model(2).cuda()
path_to_model = '/content/drive/My Drive/Models/model_87_acc_20_frames_final_data.pt'
model.load_state_dict(torch.load(path_to_model))
model.eval()
for i in range(0,len(path_to_videos)):
  print(path_to_videos[i])
  prediction = predict(model,video_dataset[i],'./')
  if prediction[0] == 1:
```

```
  print("REAL")
else:
print("FAKE")
```

Output: confidence of prediction:



7.3. RESULTS

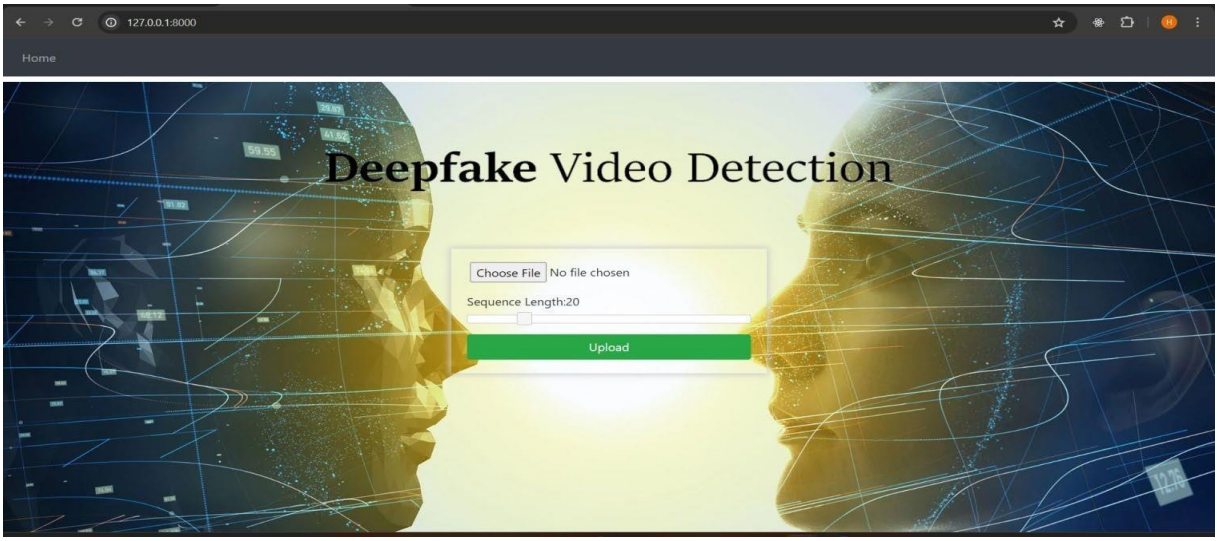


Figure 7.1: Home Page

The website appears to allow users to upload videos and check if they are deepfakes.

Figure 7.2: Uploading Real Video

It allows users to upload videos. These uploaded videos are presumably analyzed by the website to determine if they are deepfakes. Here we upload real video.

Figure 7.3: Real Video Output

1. Users upload a video.
2. The website splits the video into individual frames.
3. The website crops faces from the frames.
4. The website analyzes the faces to see if they contain signs of manipulation that would indicate a deepfake.
5. The website displays the results, letting the user know if the video is a real or deepfake
6. In this case it shows real .

Figure 7.4: Uploading Fake Video

It allows users to upload videos. These uploaded videos are presumably analyzed by the website to determine if they are deepfakes. Here we upload fake video.

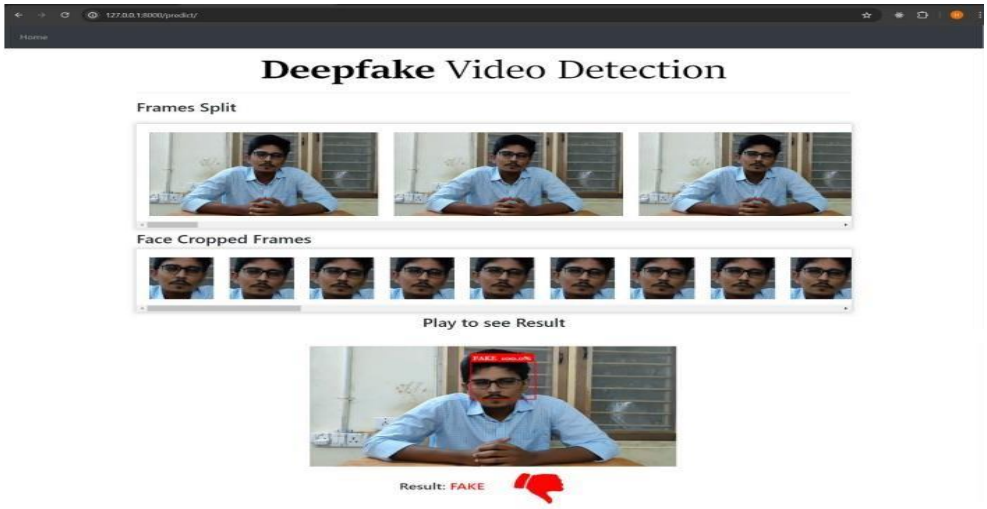


Figure 7.5: Fake video Output

1. Users upload a video.
2. The website splits the video into individual frames.
3. The website crops faces from the frames.
4. The website analyzes the faces to see if they contain signs of manipulation that would indicate a deepfake.
5. The website displays the results, letting the user know if the video is a real or deepfake
6. In this case it shows fake .

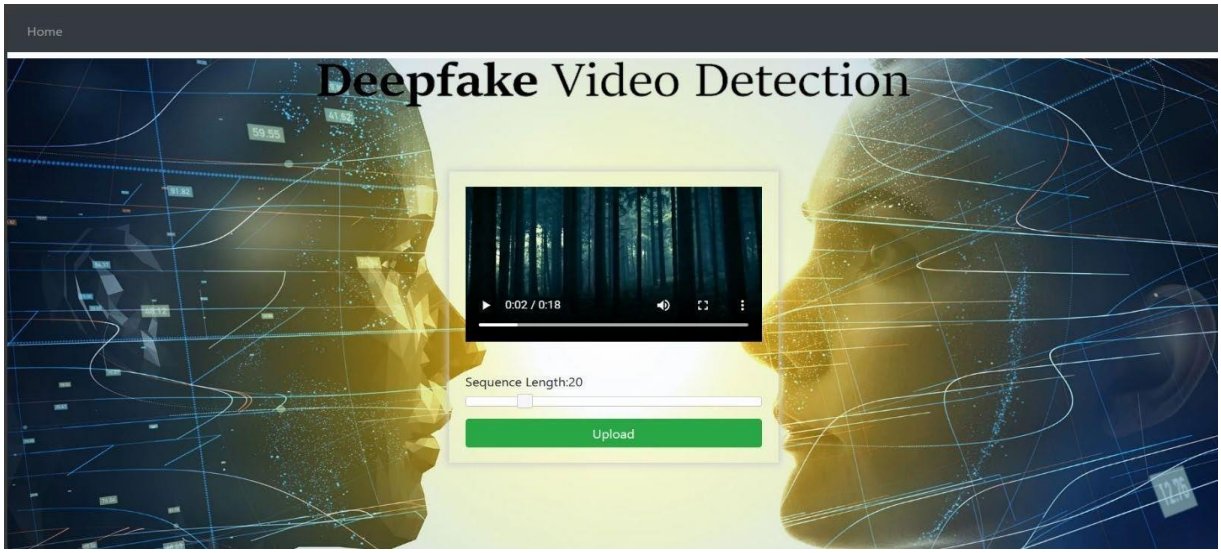


Figure 7.6: Uploading Video with no faces

It accepts videos without faces. It analyses the video and then shows output

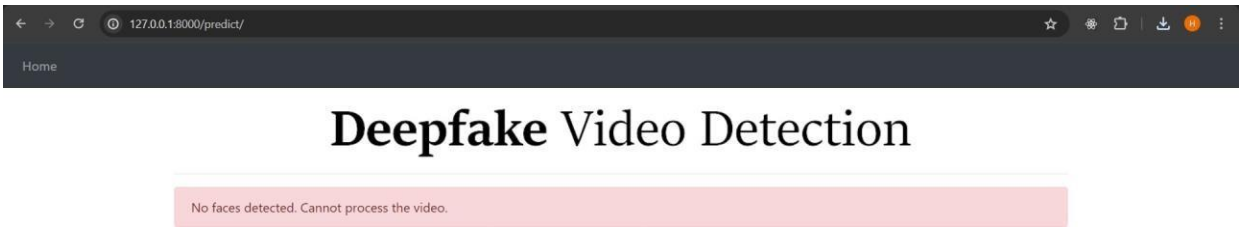


Figure 7.7: Output of Uploaded video with no faces

Figure 7.8: Uploading file greater than 100MB

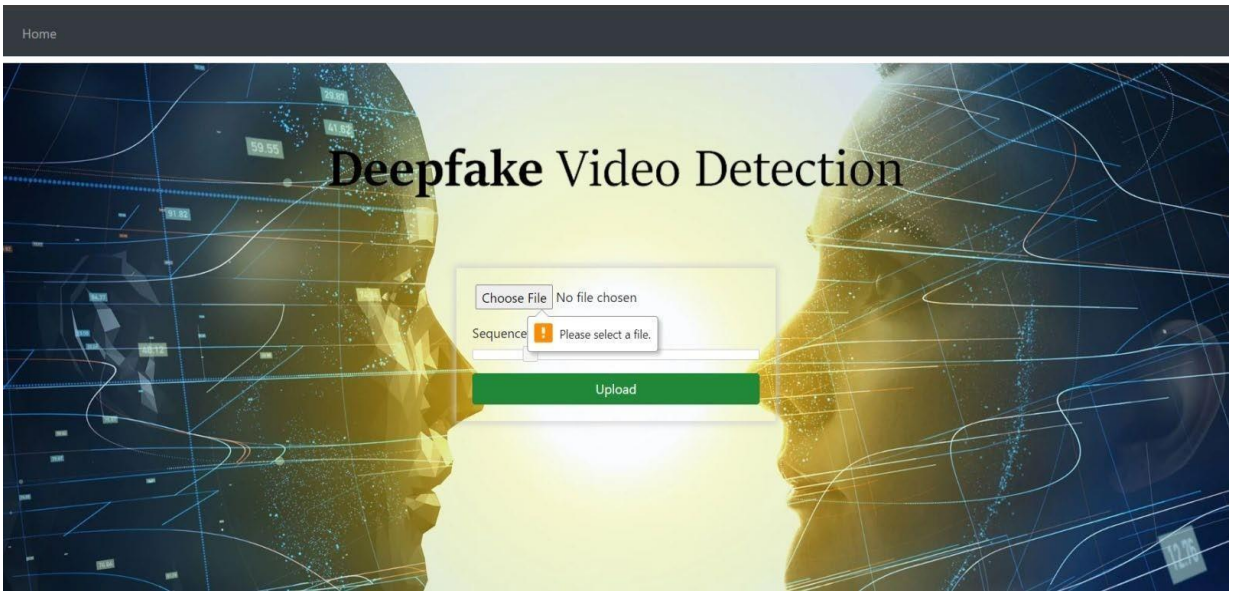


Figure 7.9: Pressing Upload button without selecting video

\

MODEL RESULTS

Model Name	Dataset	No. of videos	Sequence length	Accuracy
model_84_acc_10_frames_final_data	Our Dataset	6000	10	84.21461
model_87_acc_20_frames_final_data	Our Dataset	6000	20	87.79160
model_89_acc_40_frames_final_data	Our Dataset	6000	40	89.34681
model_91_acc_60_frames_final_data	Our Dataset	6000	60	91.59097
model_93_acc_100_frames_celeb_FF_data	Celeb-DF + FaceForen-sic++	3000	100	93.97781
model_90_acc_20_frames_FF_data	FaceForensic++	2000	20	90.95477
model_95_acc_40_frames_FF_data	FaceForensic++	2000	40	95.22613
model_97_acc_60_frames_FF_data	FaceForensic++	2000	60	97.48743
model_97_acc_80_frames_FF_data	FaceForensic++	2000	80	97.73366
model_97_acc_100_frames_FF_data	FaceForensic++	2000	100	97.76180

8. TESTING

8.1 INTRODUCTION TO TESTING

Software Testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves the execution of a software component or system component to evaluate one or more properties of interest. It is required for evaluating the system. This phase is the critical phase of software quality assurance and presents the ultimate view of coding.

Importance of Testing

The importance of software testing is imperative. A lot of times this process is skipped, therefore, the product and business might suffer. To understand the importance of testing, here are some key points to explain

• Software Testing saves money

• Provides Security

• Improves Product Quality

• Customer satisfaction

Testing is of different ways The main idea behind the testing is to reduce the errors and do it with a minimum time and effort.

Benefits of Testing

• **Cost-Effective:** It is one of the important advantages of software testing. Testing any IT project on time helps you to save your money for the long term. In case if the bugs caught in the earlier stage of software testing, it costs less to fix.

• **Security:** It is the most vulnerable and sensitive benefit of software testing. People are looking for trusted products. It helps in removing risks and problems earlier. • Product quality: It is an essential requirement of any software product. Testing ensures a quality product is delivered to customers.

• **Customer Satisfaction:** The main aim of any product is to give satisfaction to their customers. UI/UX Testing ensures the best user experience.

Different types of testing used:

Functional Testing

1. **Unit Testing**: Unit testing is a software testing method where individual units orcomponents of a software are tested in isolation to ensure they function correctlyas designed, typically through automated testing frameworks.

   Unit testing in deep fake detection involves verifying the functionality of individual components or algorithms within the detection system to ensure accurate identification of manipulated media.

2. **Integration Testing:** Integration testing verifies interactions between different components/modules of a system to ensure they work together correctly, typically involving testing interfaces and data flow.

   In deepfake detection, integration testing ensures seamless collaboration between various detection methods and modules to enhance accuracy and reliability of identifying manipulated content.

3. **System Testing:** System testing assesses the entire software system's behavior inaccordance with specified requirements, ensuring its functionality, performance, and reliability meet expectations.

   In deepfake detection, system testing examines the overall performance and effectiveness of the detection system in identifying manipulated media across various scenarios and conditions.

4. **Interface Testing:** Interface testing evaluates the interactions between different software components, focusing on ensuring smooth communication and data exchange between interfaces, APIs, and modules.

   Interface testing specifically evaluates the integration and interaction between different algorithms, models, and components within the deepfake detectionsystem, ensuring seamless communication and accurate data exchange to enhancedetection accuracy.

Non-functional Testing

1. **Performance Testing:** Performance testing evaluates the speed, responsiveness,and resource usage of deepfake detection algorithms and systems to ensureefficient and timely identification of manipulated media.

   Performance testing in deepfake detection assesses algorithmic efficiency and system scalability to swiftly and accurately detect manipulated content.

2. **Load Testing:** Load testing involves evaluating a system's performance under anticipated load levels, assessing its ability to handle concurrent users and transactions while maintaining acceptable response times and resource utilization.

   Load testing in deepfake detection evaluates the algorithmic robustness and system scalability under varying levels of computational demand, ensuring efficient and accurate identification of manipulated media amidst increased processing loads.

3. **Compatibility Testing:** Compatibility testing ensures that software or systems arecompatible with different environments, devices, and configurations, verifying seamless functionality across various platforms to guarantee a consistent user experience.

   Compatibility testing in deepfake detection ensures that detection algorithms and systems function effectively across different media formats, resolutions, and platforms, ensuring consistent and accurate identification of manipulated content across diverse sources and contexts.

8.2. Test Cases and Test Results

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

Table 8.1: Test Case Report

9. CONCLUSION

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

FUTURE SCOPE

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

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

REFERENCES

[1] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies,Matthias Nießner, “FaceForensics++: Learning to Detect Manipulated Facial Images” in arXiv:1901.08971.

[2] Deepfake detection challenge dataset : https://www.kaggle.com/c/deepfake-detection- challenge/data

[3] Yuezun Li , Xin Yang , Pu Sun , Honggang Qi and Siwei Lyu “Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics” in arXiv:1909.12962

[4] Deepfake Video of Mark Zuckerberg Goes Viral on Eve of House A.I. Hearing : https://fortune.com/2019/06/12/deepfake-mark-zuckerberg/

[5] 10 deepfake examples that terrified and amused the internet : https://www.creativebloq.com/features/deepfake-examples

[6] TensorFlow: https://www.tensorflow.org/

[7] Keras: https://keras.io/

[8] PyTorch : https://pytorch.org/

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

[10] 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.

[11] Face app: https://www.faceapp.com/

[12] Face Swap : https://faceswaponline.com/

[13] Deepfakes, Revenge Porn, And The Impact On Women : https://www.forbes.com/sites/chenxiwang/2019/11/01/deepfakes-revenge-porn-and- the-impact-on-women/

[14] The rise of the deepfake and the threat to democracy https://www.theguardian.com/technology/ng-interactive/2019/jun/22/the-rise-of-the-deepfake-and-the-threat-to-democracy

[15] Yuezun Li, Siwei Lyu, “ExposingDF Videos By Detecting Face Warping Artifacts,” in arXiv:1811.00656v3.

[16] Yuezun Li, Ming-Ching Chang and Siwei Lyu “Exposing AI Created Fake Videos by Detecting Eye Blinking” in arXiv:1806.02877v2.

.

[17] Huy H. Nguyen , Junichi Yamagishi, and Isao Echizen " Using capsule net- works to detect forged images and videos " in arXiv:1810.11215.

[18] D. Güera and E. J. Delp, "Deepfake Video Detection Using Recurrent Neural Networks," 2018 15th IEEE International Conference on Advanced Video and Sig-nal Based Surveillance (AVSS), Auckland, New Zealand, 2018, pp. 1-6.

[19] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic hu- man actions from movies. Proceedings of the IEEE Conference on Computer Visionand Pattern Recognition, pages 1–8, June 2008. Anchorage, AK

[20] Umur Aybars Ciftci, ˙Ilke Demir, Lijun Yin "Detection of Synthetic Portrait Videos using Biological Signals" in arXiv:1901.02212v2

[21] D. P. Kingma and J. Ba. Adam:

A method for stochastic optimization.arXiv:1412.6980, Dec. 2014.

[22] ResNext Model : https://pytorch.org/hub/pytorch_vision_resnext/

[23] https://www.geeksforgeeks.org/software-engineering-cocomo-model/

[24] DeepfakeVideo Detection using Neural Networks:
http://www.ijsrd.com/articles/IJSRDV8I10860.pdf

[25] International Journal for Scientific Research and Development http://ijsrd.com/