

Deepfake Video Detection Using CNN and LSTM:
Implementation and Web Application Integration

By
Bayram Tosun

Submitted to
The University of Roehampton

In partial fulfilment of the requirements
for the degree of
BACHELOR OF SCIENCE IN COMPUTING

Abstract

This report details the development of a CNN + LSTM deepfake detection model designed to accurately discern real from manipulated video content. The model employs a pre-trained ResNet50V2 architecture for effective spatial feature extraction and integrates it with a Long Short-Term Memory (LSTM) network to capture temporal inconsistencies within video sequences. Utilizing datasets from FaceForensics++ and Celeb-DF v2, the system pre-processes videos by detecting and cropping faces, then creating short, face-focused video clips.

Integrated within a Django web application, the system allows users to upload videos, which are then analysed to determine their authenticity. The web application features an intuitive interface, including a history of analysed videos and a link to the project's GitHub repository. Evaluation metrics, including high accuracy and strong discriminatory power, underscore the model's efficacy in identifying deepfakes. This work highlights the potential of combining CNN and LSTM techniques to enhance digital media integrity amidst the growing challenge of deepfake technology.

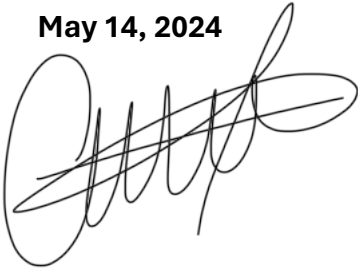
Declaration

I hereby certify that this report constitutes my own work, that where the language of others is used, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions, or writings of others.

I declare that this report describes the original work that has not been previously presented for the award of any other degree of any other institution.

Bayram Tosun

May 14, 2024

A handwritten signature in black ink, appearing to be 'Bayram Tosun', written over the date.

Acknowledgements

I would like to express my deepest gratitude to Dr. Kimia Aksir, my supervisor, for her invaluable guidance, support, and encouragement throughout the duration of my final year project. Her expertise and insights have been instrumental in the successful completion of this work.

I would also like to extend my thanks to University of Roehampton for providing the resources and environment necessary for the execution of this project. The facilities and academic atmosphere have greatly contributed to my learning and development.

Lastly, I am grateful to my family and friends for their continuous support and encouragement during this journey. Their belief in me has been a constant source of motivation.

Table of Contents

1. Introduction.....	7
1.1 Problem Description, Context, and Motivation	7
1.2 Aims.....	7
1.3 Objectives	8
1.4 Social, Ethical, and Legal Considerations.....	8
1.5 Background	9
1.6 Report Overview	10
2. Literature – Technology Review	11
2.1 Definition of Deepfake.....	11
2.2 Deepfake Generation Methods.....	11
2.3 Deepfake Detection Datasets	12
2.4 Deepfake Detection Methods and Related Works	15
2.4.1 Challenges in Deepfake Detection.....	16
2.5 Web Application Frameworks and Technologies	17
3. Methodology & Design	19
3.1 Dataset Collection.....	19
3.2 Pre-processing	20
3.3 Proposed Model Architecture	21
3.3.1 Pre-trained Model Architecture: ResNet50v2	23
3.4 Model Training and Evaluation.....	24
3.4.1 Data Loading and Augmentation.....	24
3.4.2 Model Training.....	25
3.4.3 Model Evaluation.....	25
3.5 Web Application Design.....	27
3.5.1 Functionality	28
4. Implementation & Results	29

4.1	System Architecture	29
4.1.1	Overview of Implemented Systems.....	29
4.2	Dataset Collection and Pre-processing.....	31
4.2.1	Dataset Statistics.....	33
4.3	Model Training	34
4.3.1	Loading Data.....	34
4.3.2	Model Architecture and Hyperparameters.....	35
4.3.3	Sequence Length and Padding.....	35
4.3.4	Training Progress and Convergence.....	36
4.4	Evaluation and results	37
4.4.1	Confusion Matrix	37
4.4.2	Loss and Accuracy Curves	38
4.4.3	Classification Report.....	38
4.4.4	ROC Curve.....	39
4.5	Web Application.....	40
5.	Conclusion	43
5.1	Future Work	43
5.2	Reflection.....	44
6.	References	46
7.	Appendices	49

1. Introduction

In the digital tapestry of the 21st century, deepfake technology weaves a complex narrative that blurs the lines between reality and fiction. This technology, which enables the creation of convincingly altered videos, poses significant challenges to the veracity of digital media, prompting an imperative development in the field of deepfake detection [1]. The quest to maintain the sanctity of truth amidst our virtual interactions has never been more pressing, as deepfake detection stands as a bulwark against the tide of digital deception.

As social media platforms burgeon and become the linchpin of information dissemination, the ubiquity of deepfakes represents a formidable threat, with the potential to distort perceptions, impinge upon privacy, and undermine democratic institutions [2]. Thus, the pursuit of robust deepfake detection mechanisms is not merely an academic exercise but a societal necessity.

Deepfake detection is an intricate process that scrutinizes composite elements within digital content, seeking anomalies in facial dynamics, voice modulation, and contextual cues that may betray a video's authenticity [3]. This endeavour necessitates not only cutting-edge computational techniques but also an interdisciplinary acumen to navigate the nuanced complexities of digital forensics.

1.1 Problem Description, Context, and Motivation

- **What is the problem?** The core problem is the creation and circulation of deepfake videos which can be used to spread false information, manipulate public opinion, and tarnish reputations.
- **Who is affected?** Individuals, organizations, and societies at large are affected by the malicious use of deepfake technology.
- **Where and/or when does it occur?** This issue is pervasive across the digital sphere and is not constrained by geography or time.
- **Why is it important to solve?** Addressing this problem is critical to maintain trust in digital media, protect individual rights, and safeguard democratic processes.

1.2 Aims

- To design and develop a robust deepfake detection model using cutting-edge machine learning techniques.
- To contribute to the body of research focused on mitigating the risks associated with deepfake technology.
- To augment public awareness regarding the identification of altered video content.

1.3 Objectives

The project is driven by three primary objectives:

- **Dataset Curation:** Assemble and refine a dataset that accurately represents both genuine and manipulated media, ensuring it is conducive for deep learning model training.
- **Model Development:** Construct a neural network model with the capacity to discern and classify media based on authenticity, emphasizing scalability and robustness.
- **Model Evaluation:** Assess the model's accuracy and reliability using established performance metrics, aiming for high precision in detecting deepfakes.

1.4 Social, Ethical, and Legal Considerations

In the development of the deepfake detection program, stringent legal considerations have been accounted for, particularly concerning copyright laws, which govern the use of datasets and the distribution of the developed software. The project operates within the legal framework that regulates the analysis of digital content and respects the copyright of the data used for training and testing the deep learning models.

Intellectual property rights are of paramount importance, especially when utilizing pre-existing media to construct the dataset necessary for training the detection algorithms [4]. Moreover, data protection regulations such as the General Data Protection Regulation (GDPR) in the European Union play a critical role in guiding the handling and processing of personal data [5]. This project adheres to such regulations by implementing measures to anonymize and secure any personal data involved.

With deepfakes' potential to deceive and harm, defamation laws and the right to privacy are also at the forefront of this project's legal landscape. It is essential to ensure that the algorithms developed do not become tools for infringement on individuals' rights or facilitate unauthorized use of their likeness. The program's objective is to detect and flag deepfake content while maintaining ethical standards and protecting individuals from harm, aligning with the broader aim of preventing the misuse of AI in generating deceptive media.

As lawmakers grapple with the challenges posed by deepfake technology, there is an ongoing discussion about the need to update existing laws or create new ones to better address these emerging issues [6]. This project remains agile and ready to adapt to such legislative changes to ensure continued compliance and responsible innovation in the field of deepfake detection.

The social impact of deepfake technology is profound, affecting the fundamental trust individuals place in digital media. The rise of deepfakes has the potential to erode public confidence in information dissemination channels, making the development of detection tools not just a technical challenge, but a social necessity [2]. By providing the means to discern reality from fabrication, deepfake detection programs serve as a cornerstone for maintaining informational integrity. Such initiatives are pivotal in promoting digital literacy, equipping society to critically evaluate media content and recognize deceptive practices. The goal is to reinforce societal values that emphasize transparency and authenticity in the digital realm [3].

Ethically, the deployment of deepfake detection technology is a balancing act between defending the truth and respecting individual privacy rights. The use of facial recognition and imagery in detection algorithms raises privacy concerns and necessitates careful consideration to avoid infringing on personal freedoms [7]. Developers must be vigilant to ensure the technology they create does not become a tool for defamation or an instrument of unauthorized surveillance. The ethical mandate extends to preventing the potential weaponization of deepfake detection for censorship, thereby upholding the values of freedom of expression and the right to privacy [4].

From a professional perspective, the creation of a deepfake detection system is an exercise in applying the principles of software engineering excellence. Adherence to best practices in software development is paramount, as the resulting tools often become integral to fields that rely on the veracity of digital content, such as journalism and law enforcement [11]. Developers must engage in continuous education to keep pace with the ever-evolving landscape of digital forgery, ensuring their tools remain effective against the latest iterations of deepfakes. It is the responsibility of professionals in this domain to foster a culture of integrity and accountability in media, where the authenticity of content is continuously scrutinized and upheld [12].

1.5 Background

As artificial intelligence (AI) forges ahead, the proliferation of deepfake technology has escalated, presenting unprecedented challenges in distinguishing genuine digital media from manipulated content. The sophistication of deepfakes—a form of AI-generated media that convincingly alters or fabricates audio-visual content—poses a severe threat to the integrity and trustworthiness of information disseminated across digital platforms. The imperceptible nature of these manipulations to the human eye demands robust and reliable detection mechanisms.

Pioneering studies, such as those by Rossler et al. [7], have established significant benchmarks in the field of computer vision and machine learning, contributing foundational insights into the detection of deepfakes. This project builds upon such contributions, endeavouring to refine and

enhance the detection of digitally altered media. Leveraging the capabilities of deep learning and neural networks, this project aims to identify the intricate discrepancies and artifacts introduced during the deepfake generation process.

The ramifications of deepfake technology extend beyond mere technological intrigue; they penetrate the very fabric of societal trust [3]. The potential utilization of deepfakes in malicious disinformation campaigns can have profound impacts on political, social, and individual domains. Addressing this, our project employs state-of-the-art convolutional neural networks (CNNs) to analyse and detect the subtlest signs of forgery, providing a bulwark against the tide of digital deceit.

Interdisciplinary collaboration is at the heart of our approach, integrating expertise from computer science, cybersecurity, legal studies, and ethics to navigate the multifaceted challenges posed by deepfakes. By fostering transparency and accountability in our development process, we ensure the reliability of our detection system and maintain the confidence of stakeholders [6]. As we advance this research, our goal extends beyond the technical horizon to encompass the creation of a holistic ecosystem, including media outlets, technology enterprises, legislative bodies, and academic institutions, all unified in the mission to safeguard digital authenticity.

1.6 Report Overview

This report delineates the journey of our project from inception to completion. Following this introduction, the subsequent sections will delve into the methodology, detailing the dataset curation, system architecture, and computational environment. We then explore the implementation, the training procedures of our model, and the critical evaluation of its performance. The report culminates in a discussion of the project's implications and posits potential avenues for future research, underlining our dedication to combating the threat of deepfakes with innovation and rigor.

2. Literature – Technology Review

2.1 Definition of Deepfake

Deepfakes are a form of synthetic media created using artificial intelligence (AI) techniques to manipulate or generate audio-visual content that appears real but is fabricated [9]. These manipulations can involve altering, doctoring, or fabricating visual and audio materials with the intent to deceive viewers [10]. Deepfakes are often strategically crafted to harm individuals by making them appear to say or do things that are provocative, conflicting, or highly implausible. They are characterized by their hyper-realistic nature, presenting falsified images, videos, and audio that closely resemble authentic content [11].

2.2 Deepfake Generation Methods

Deepfake generation methods involve a variety of techniques that utilize generative deep learning algorithms to manipulate human faces and create realistic but fabricated content. These methods, such as FaceSwap, FaceGuard, ICface, and others, manipulate facial features to generate hyper-realistic deepfake content Yang et al. [12]. They often utilize generative adversarial networks (GANs) and other advanced technologies to produce convincing deepfakes.

One illustrative example of GANs in action within the realm of deepfakes is the DeepFaceLab framework. DeepFaceLab is an open-source framework designed for face swapping that offers an integrated, flexible, and extensible solution for creating deepfakes Liu et al. [13]. Researchers have utilized DeepFaceLab for tasks such as face swapping to protect patient privacy Wilson et al. [14] and deepfake detection for facial images with facemasks.

DeepFaceLab utilizes a specific configuration of GANs to create and refine synthetic facial images. In this setup, the generator component of the GAN learns to create visually realistic human faces by training on a dataset of real images, attempting to generate new faces that cannot be distinguished from real ones by the human eye. Concurrently, the discriminator part of the GAN assesses the authenticity of each generated image, determining whether it is real or artificial. Through continuous training, where the generator and discriminator iteratively adjust based on each other's output, the fidelity of the generated images improves significantly. This adversarial process ensures that the deepfakes are refined to a level where they closely mimic the intended real-life subjects in terms of appearance, expressions, and movements, making detection increasingly challenging.

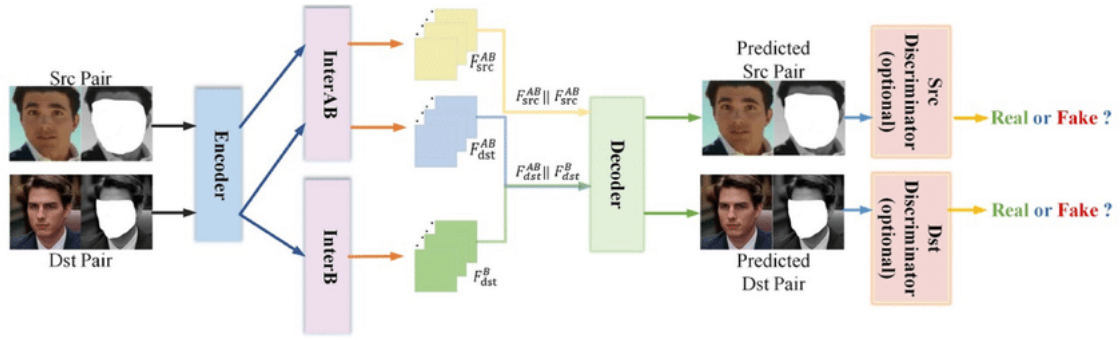


Figure1: The architecture of DeepFaceLab framework.

The development of deepfake technology has evolved to include a variety of models, each differing in aspects such as encoder and decoder architectures, input resolution, and compression ratios. This diversity allows for the generation of thousands of deepfake videos from relatively similar initial inputs, highlighting the capability of these methods to produce a vast array of manipulated content from limited data sets. The inherent complexity of these generative techniques poses significant challenges for the detection of deepfakes, as they can closely mimic real human expressions and actions, making them difficult to identify as forgeries.

2.3 Deepfake Detection Datasets

Deepfake detection datasets play a critical role in the development and evaluation of algorithms designed to identify and mitigate the effects of synthetic media. These datasets provide researchers and developers with a diverse range of video and image samples, which include both real and synthetically altered faces, to train and test deepfake detection models. The quality, diversity, and size of these datasets significantly influence the effectiveness of the detection models.

FaceForensics++

This dataset is one of the most comprehensive collections used for deepfake detection research. It contains a large number of video sequences that have been manipulated using various methods, including DeepFakes, Face2Face, FaceSwap, and NeuralTextures. Each manipulation technique is applied to 1,000 videos, resulting in a dataset that is not only diverse but also challenging due to the high quality and varying levels of compression of the videos.[7]

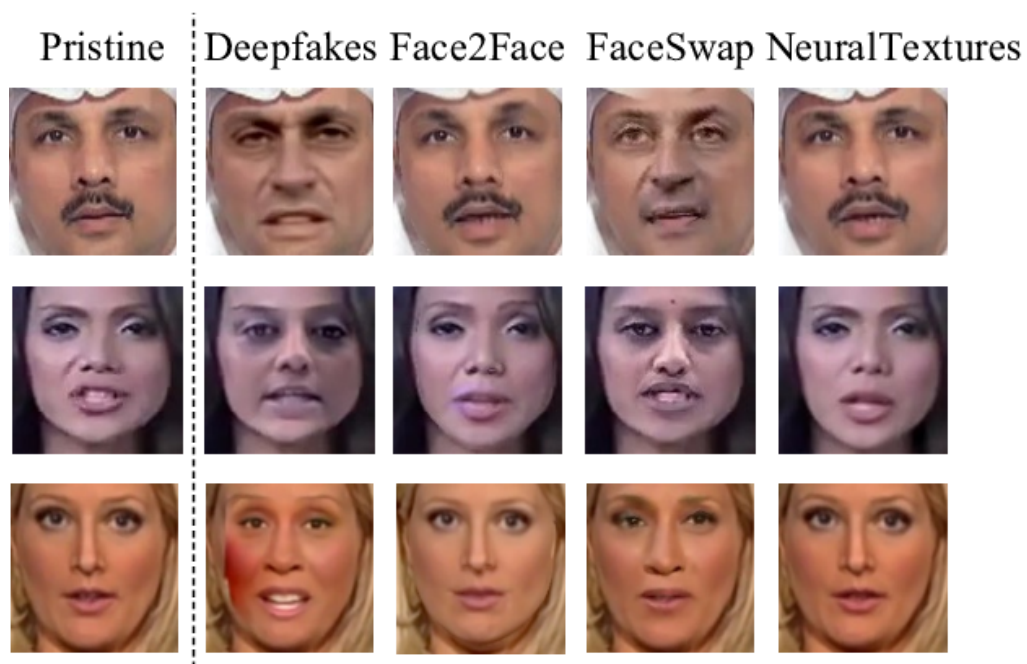


Figure 2: Sample face images from different attacks of FaceForensics++ dataset.

Deepfake Detection Challenge Dataset (DFDC)

Launched by Facebook in collaboration with Microsoft, Amazon, and the Partnership on AI, the DFDC is one of the largest public datasets available. It features over 100,000 videos constructed from 3,426 paid actors, who provided a diverse representation in terms of demographics. The videos vary in terms of scenes, lighting, and poses, making it an invaluable resource for training robust detection systems.[15]



Figure 3: A sample of the faces extracted from the DFDC dataset and the true labels.

Celeb-DF

A dataset that was developed to address some of the shortcomings of earlier datasets, particularly in terms of visual quality. Celeb-DF includes high-quality deepfake videos of celebrities generated using improved synthesis processes. This dataset highlights the challenges posed by high-quality deepfakes that are more difficult to detect due to their realistic appearance.[16]



Figure 4: Example frames from Celeb-DF Dataset

Google's DeepFake Detection Dataset

Created by Google in partnership with Jigsaw, this dataset includes thousands of manipulated videos featuring a variety of scenes and backgrounds. The dataset was designed to help improve the development of technologies aimed at detecting deepfakes with a particular focus on diverse scenarios and lighting conditions.[17]

Deepfake detection datasets are fundamental for training machine learning models in recognizing the subtle cues that distinguish genuine from altered content. The effectiveness of detection models depends heavily on the quality, variability, and realism of the datasets they are trained on. For instance, datasets with a wide range of manipulation techniques and quality levels (from low to high resolution) allow detection systems to adapt to various forms of deepfakes that they might encounter in real-world scenarios.

Furthermore, the evolution of deepfake generation methods continually necessitates updates and expansions to existing datasets. New datasets or additions to existing collections should ideally include newer manipulation techniques and better-quality deepfakes, reflecting ongoing advances in generative AI technologies.

2.4 Deepfake Detection Methods and Related Works

Deepfake detection has become a crucial research area due to the increasing sophistication of deepfake generation methods and the potential for misuse. Various approaches have been proposed to detect deepfakes, ranging from traditional machine learning techniques to deep learning-based methods.

One common approach is to utilize convolutional neural networks (CNNs) to extract features from images or video frames and classify them as real or fake. Afchar et al. [1] proposed MesoNet, a compact CNN architecture that focuses on mesoscopic properties of images to detect deepfakes. Another CNN-based method, FaceForensics++ [7], employs a combination of low-level and high-level features to improve detection accuracy. Recurrent neural networks (RNNs) and long short-term memory (LSTM) networks have also been employed to capture temporal inconsistencies in deepfake videos. Güera and Delp [18] proposed a deepfake detection method that uses a combination of CNNs and LSTMs to analyse video sequences and detect inconsistencies in facial expressions and movements.

Other approaches focus on specific artifacts or irregularities introduced during the deepfake generation process. Li et al. [4] proposed a method that exploits the blending boundary artifacts introduced by the face swapping process in deepfakes. Matern et al. [19] developed a technique that leverages visual artifacts, such as inconsistencies in eye blinking patterns, to expose deepfakes.

Researchers have also explored the use of physiological signals, such as heart rate and blood flow, to detect deepfakes. Ciftci et al. [20] proposed a method that analyses the subtle changes in facial blood flow to distinguish between real and fake videos.

Despite the progress made in deepfake detection, challenges remain. Deepfake generation methods are constantly evolving, making it difficult for detection methods to keep pace. Additionally, detecting deepfakes in real-world scenarios, where the quality and resolution of videos may vary, poses a significant challenge.

To address these challenges, researchers are exploring advanced techniques such as attention mechanisms, multi-task learning, and adversarial training to improve the robustness and generalization ability of deepfake detection models. The availability of large-scale datasets, such

as FaceForensics++ [7], Celeb-DF [15], and the Deepfake Detection Challenge Dataset [16], has also facilitated the development and evaluation of novel detection methods.

After reviewing the various deepfake detection methods and related works, the authors have chosen to implement a model that combines convolutional neural networks (CNNs) for feature extraction and long short-term memory (LSTM) networks for temporal sequence analysis. This decision is based on the proven effectiveness of CNNs in capturing spatial features from images or video frames and the ability of LSTMs to model temporal dependencies and inconsistencies in video sequences [31].

The combination of CNN and LSTM has been successfully employed in previous deepfake detection methods, demonstrating its effectiveness [18]. By leveraging the strengths of both CNNs and LSTMs, the authors aim to develop a robust and accurate deepfake detection model that can handle the challenges posed by the constantly evolving deepfake generation techniques. The availability of large-scale datasets, such as FaceForensics++ [7] and the Deepfake Detection Challenge Dataset [16], provides ample training data to fine-tune and evaluate the performance of the proposed CNN+LSTM model.

2.4.1 Challenges in Deepfake Detection

Despite the progress made in deepfake detection, several challenges hinder the development and deployment of reliable and robust detection methods. These challenges arise from the constantly evolving nature of deepfake generation techniques, the lack of diverse and representative datasets, and the need for real-time and scalable detection solutions.

One major challenge is the arms race between deepfake generation and detection methods. As detection techniques improve, malicious actors develop more sophisticated deepfake generation methods to evade detection. This constant evolution makes it difficult for detection methods to keep pace and maintain their effectiveness over time [21]. Researchers need to continuously update and adapt their detection models to capture the latest deepfake generation techniques.

Another challenge is the lack of large-scale, diverse, and representative datasets for training and evaluating deepfake detection models. Many existing datasets focus on specific types of deepfakes or are limited in terms of the number of subjects, variations in lighting, poses, and expressions [7]. This lack of diversity can lead to overfitting and poor generalization of detection models to real-world scenarios. Efforts are being made to create more comprehensive datasets, such as the Deepfake Detection Challenge Dataset [16], but there is still a need for even larger and more varied datasets.

The quality and resolution of deepfake videos pose another challenge for detection methods. Deepfakes can be generated at various quality levels, ranging from low-resolution, visibly manipulated videos to high-resolution, visually convincing ones. Detection methods that perform well on high-quality deepfakes may struggle with low-quality ones, and vice versa [22]. Developing detection methods that are robust to variations in quality and resolution is an ongoing research challenge.

Real-time detection of deepfakes is another significant challenge, particularly in scenarios where immediate action is required, such as in live video streams or social media platforms. Many existing detection methods rely on computationally intensive deep learning models that may not be suitable for real-time processing on resource-constrained devices [18]. Researchers are exploring ways to optimize detection models and develop lightweight architectures that can operate in real-time without sacrificing accuracy.

The interpretability and explainability of deepfake detection models are also important challenges. Many deep learning-based detection methods operate as "black boxes," making it difficult to understand how they arrive at their decisions [23]. This lack of interpretability can hinder the trust and adoption of detection methods in real-world applications. Researchers are working on developing more interpretable and explainable detection models that provide insights into the decision-making process.

Finally, the ethical and legal implications of deepfakes and their detection pose significant challenges. The misuse of deepfakes can have serious consequences, such as the spread of misinformation, privacy violations, and reputational damage [24]. Developing detection methods that are not only accurate but also respect privacy and adhere to ethical and legal standards is crucial.

2.5 Web Application Frameworks and Technologies

Several web application frameworks and technologies were considered for developing the user interface and presenting the deepfake detection system. The main options explored were:

Django: A high-level Python web framework that encourages rapid development and clean, pragmatic design [25].

Flask: A lightweight and extensible Python web framework that provides a simple and intuitive way to build web applications [26].

React: A JavaScript library for building user interfaces, known for its component-based architecture and efficient rendering [27].

Angular: A comprehensive JavaScript framework for building web applications, offering features like dependency injection and two-way data binding [28].

After evaluating the strengths and limitations of each option, Django was chosen as the web application framework for this project. Django's robustness, scalability, and built-in features, such as an admin interface and ORM (Object-Relational Mapping), made it well-suited for developing the deepfake detection web application. Its large community and extensive documentation also provided valuable support throughout the development process. By leveraging Django's capabilities, the web application was developed to provide a user-friendly interface for uploading videos, displaying detection results, and maintaining a record of analysed videos. The choice of Django as the web framework allowed for efficient development and deployment of the deepfake detection system.

3. Methodology & Design

This section presents the methodology and design of our deepfake detection system. We discuss the dataset collection and pre-processing, the proposed model architecture, the training and evaluation process, and the web application for user interaction.

The dataset consists of real and fake videos sourced from FaceForensics++ and Celeb-DF v2. The pre-processing pipeline involves splitting videos into frames, detecting, and cropping faces, and saving them as individual images.

The proposed model architecture combines a CNN (ResNet50V2) for spatial feature extraction and an LSTM for temporal sequence analysis. The training process utilizes data augmentation, balanced batch selection, and regularization techniques. The model's performance is evaluated using metrics such as accuracy, precision, recall, F1-score, and ROC AUC.

A web application is developed to facilitate user interaction with the trained model, allowing users to upload videos and view detection results.

The following subsections will provide more details on each component of the methodology and design.

3.1 Dataset Collection

For the development and evaluation of the proposed deepfake detection model, two widely recognized datasets were selected: FaceForensics++ [22] and Celeb-DF v2 [24].

FaceForensics++ contains over 1.8 million video frames from 1,000 original videos and their corresponding manipulated versions, covering various manipulation techniques. The dataset provides a balanced distribution of real and fake videos, making it suitable for training and testing deepfake detection models.

Celeb-DF v2 consists of 5,639 deepfake videos generated using advanced synthesis algorithms, along with 590 real videos of celebrities. The dataset covers a wide range of video resolutions, compression levels, and visual quality, simulating real-world scenarios.

By combining these datasets, the authors aim to create a diverse and representative dataset that encompasses various deepfake generation techniques, video qualities, and subject demographics. This approach ensures that the proposed deepfake detection model is trained and evaluated on a wide range of realistic scenarios, enhancing its robustness and generalization capability.

3.2 Pre-processing

In the deepfake detection project, data pre-processing plays a crucial role in preparing the dataset for training and evaluation. The pre-processing pipeline involves several key steps to ensure the data is properly organized, cleaned, and formatted for the subsequent stages of the project.

The dataset used in this project consists of two main sources: the FaceForensics++ dataset and the Celeb-DF v2 dataset. The FaceForensics++ dataset contains real videos and fake videos generated using various deepfake techniques, such as Neural Textures, Deepfakes, Face2Face, FaceSwap, and Face Shifter. The Celeb-DF v2 dataset is specifically designed for deepfake detection and includes high-quality deepfake videos of celebrities. By combining these two datasets, the project aims to create a comprehensive and diverse dataset for training and evaluating the deepfake detection model.

To ensure a balanced distribution of videos across the training, validation, and test sets, the pre-processing pipeline employs a strategy to evenly distribute the videos. The videos are split into three sets: 70% for training, 15% for validation, and 15% for testing. This distribution allows for sufficient data to train the model effectively while also providing separate sets for hyperparameter tuning and unbiased evaluation.

One of the critical aspects of pre-processing is extracting relevant features from the videos. In this project, the focus is on detecting deepfakes based on facial features. The pre-processing pipeline includes two key steps: face detection and extraction, and frame sequence extraction.

Firstly, the MTCNN algorithm is used to detect faces in each frame of the videos. The detected faces are then cropped and resized to a consistent target size of 224x224 pixels to ensure uniformity across the dataset.

Secondly, to capture temporal information, the pre-processing pipeline extracts the first 100 frames of each video to form a frame sequence. This sequence length is chosen to balance capturing sufficient temporal information and maintaining computational efficiency.

The combination of facial feature extraction and frame sequence extraction enables the model to capture both spatial and temporal information, enhancing its ability to detect deepfakes effectively. The pre-processed data is saved in a structured format, ready to be fed into the deepfake detection model for training and evaluation.

After extracting and cropping the facial regions from the video frames, the pre-processing pipeline takes an additional step to create coherent video sequences. The cropped facial frames are merged back together to form short video clips. This step is crucial because the deepfake detection model utilizes a combination of Convolutional Neural Networks (CNNs) and Long Short-Term

Memory (LSTM) networks. By creating video sequences from the extracted facial frames, the model can effectively capture both spatial and temporal information, enabling it to detect inconsistencies and anomalies associated with deepfake videos.

To efficiently process the large number of videos in the dataset, the pre-processing pipeline leverages concurrent processing techniques. By utilizing multiple threads or processes, the pipeline can simultaneously handle multiple videos, significantly reducing the overall pre-processing time. This parallel processing approach allows for the extraction of frames, face detection, and video creation from multiple videos concurrently, making the pre-processing stage more scalable and efficient.

Throughout the pre-processing pipeline, metadata is generated and stored for each processed video. The metadata includes essential information such as the file path, label (real or fake), set type (train, validation, or test), video ID, and frame number. This metadata serves as a comprehensive record of the pre-processed data and is crucial for training and evaluating the deepfake detection model.

Finally, the pre-processed data and associated metadata are organized and stored in a structured format. The created video sequences are saved in separate directories corresponding to their respective sets (train, validation, or test) and categories (real or fake). Additionally, a metadata CSV file is generated, containing all the relevant information about each processed video and its corresponding frames. This CSV file serves as a convenient input for the subsequent stages of the project, such as data loading and model training.

By following this pre-processing approach, the deepfake detection project ensures that the dataset is well-organized, balanced, and ready for further analysis and model development. The incorporation of the Celeb-DF v2 dataset alongside FaceForensics++ enhances the diversity and quality of the training data. The pre-processing pipeline efficiently handles the extraction of facial features, creation of video sequences, concurrent processing of videos, and generation of comprehensive metadata. This foundation sets the stage for the development of robust and accurate deepfake detection models using a combination of CNNs and LSTMs.

3.3 Proposed Model Architecture

The proposed deepfake detection model architecture combines Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks to effectively capture both spatial and temporal features from video sequences. This hybrid approach aims to leverage the strengths of both types of networks to accurately detect deepfakes.

The model architecture consists of two main components: a CNN for feature extraction and an LSTM for temporal sequence analysis. The CNN component is responsible for extracting discriminative spatial features from each frame of the video sequence. In this architecture, a pre-trained ResNet50V2 model is utilized as the base CNN. The choice of using a pre-trained model is motivated by the benefits of transfer learning, which allows the model to leverage the knowledge gained from training on a large-scale dataset, such as ImageNet, and apply it to the specific task of deepfake detection.

The ResNet50V2 model is initialized with pre-trained weights from ImageNet and is used as a feature extractor. The top layers of the ResNet50V2 model are removed, and the output of the last convolutional layer is used as the feature representation for each frame. To adapt the pre-trained model to the specific task of deepfake detection, additional convolutional and pooling layers are added on top of the base model. These layers help in capturing more fine-grained details specific to the deepfake detection problem.

To handle the temporal aspect of the video sequences, the extracted features from each frame are passed through a TimeDistributed layer, which applies the same CNN model to each frame independently. The resulting feature maps are then processed by the LSTM component of the architecture.

The LSTM network is designed to capture the temporal dependencies and inconsistencies across the frames. In this architecture, a Bidirectional LSTM layer is employed, which processes the sequence of feature maps in both forward and backward directions. This allows the model to consider both past and future context when making predictions. The LSTM layer is regularized using L1 and L2 regularization techniques to prevent overfitting and improve generalization.

After the LSTM layer, dropout regularization is applied to further reduce overfitting. The output of the LSTM layer is then passed through fully connected layers, which gradually reduce the dimensionality of the features. The final fully connected layer uses the sigmoid activation function to produce a binary output, indicating whether the input video sequence is a deepfake or not.

The model is trained using the Adam optimizer, which adapts the learning rate for each parameter based on the historical gradients. The binary cross-entropy loss function is used as the objective function, and the model's performance is evaluated using accuracy as the metric.

3.3.1 Pre-trained Model Architecture: ResNet50v2

The proposed deepfake detection model leverages the ResNet50V2 architecture as the pre-trained CNN component. ResNet50V2 is a variant of the ResNet (Residual Network) architecture, which has achieved state-of-the-art performance on various computer vision tasks, including image classification.

The choice of using a pre-trained ResNet50V2 model is motivated by several factors. Firstly, training a deep CNN from scratch requires a large amount of labelled data, which can be challenging and time-consuming to acquire. By utilizing a pre-trained model, the deepfake detection system can benefit from the knowledge learned by the model on a large-scale dataset like ImageNet, which contains millions of labelled images across a wide range of categories.

Secondly, the ResNet50V2 architecture has been designed to mitigate the vanishing gradient problem and enable the training of deeper networks. It introduces residual connections, which allow the gradients to flow more easily through the network during backpropagation. This facilitates the learning of more complex and discriminative features, which is crucial for detecting subtle artifacts and inconsistencies in deepfake videos.

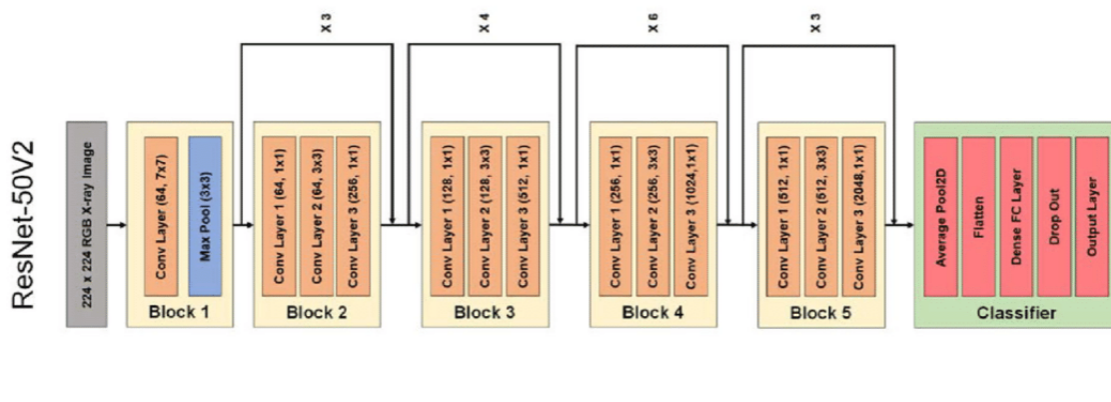


Figure 6: Architecture of the ResNet50V2 model.

By leveraging the pre-trained ResNet50V2 model, the deepfake detection system can take advantage of the model's ability to capture high-level semantic features and adapt them to the specific task of deepfake detection. The pre-trained weights serve as a good initialization for the model, reducing the training time and improving the convergence of the learning process.

It is important to note that while the pre-trained ResNet50V2 model provides a strong foundation, it is not used as is for deepfake detection. The top layers of the model are removed, and additional layers are added to tailor the model to the specific requirements of the deepfake detection task.

This fine-tuning process allows the model to learn more domain-specific features and adapt to the characteristics of deepfake videos.

In summary, the proposed deepfake detection model architecture combines the power of a pre-trained ResNet50V2 CNN for spatial feature extraction with an LSTM network for temporal sequence analysis. This hybrid approach enables the model to capture both the visual artifacts and the temporal inconsistencies associated with deepfake videos. By leveraging transfer learning through the use of a pre-trained ResNet50V2 model, the system can benefit from the knowledge gained from large-scale image datasets and adapt it to the specific task of deepfake detection.

3.4 Model Training and Evaluation

The training and evaluation process is a critical aspect of developing a robust and effective deepfake detection model. It involves carefully preparing the training data, designing an efficient data loading and augmentation strategy, selecting appropriate hyperparameters, and monitoring the model's performance during training. The evaluation phase assesses the model's ability to generalize to unseen data and provides insights into its strengths and limitations.

3.4.1 Data Loading and Augmentation

To efficiently load and pre-process the video sequences during training, a custom data generator called `'VideoFrameSequenceGenerator'` is implemented. This generator takes a data frame containing video file paths and labels as input and generates batches of video frames along with their corresponding labels on-the-fly.

The `'VideoFrameSequenceGenerator'` incorporates several important features to enhance the training process. It ensures a balanced batch by randomly selecting an equal number of real and fake video sequences for each batch. This helps to prevent the model from biasing towards one class during training.

Data augmentation techniques are applied to the video frames to increase the diversity of the training data and improve the model's generalization ability. These techniques include random rotations, shifts, shears, zooms, and flips. By applying these transformations, the model learns to be invariant to minor variations in the input data, making it more robust to real-world variations.

The generator efficiently loads and pre-processes the video frames on-the-fly, reducing memory usage and enabling training on large datasets. It reads the video frames from disk, resizes them to

a consistent size, and normalizes the pixel values to a range of [0, 1]. This pre-processing step ensures that the input data is in a suitable format for the model.

3.4.2 Model Training

The deepfake detection model is trained using the Adam optimizer, which adaptively adjusts the learning rate for each parameter based on its historical gradients. The learning rate is set to $1e-5$, which determines the step size at which the model's weights are updated during training. The binary cross-entropy loss function is used as the objective function, which measures the dissimilarity between the predicted probabilities and the true labels.

To monitor and control the training process, several callbacks are employed:

1. `'ModelCheckpoint'`: This callback saves the best model weights based on the validation accuracy. It allows the model to be restored to its best-performing state after training.
2. `'EarlyStopping'`: This callback monitors the validation loss and stops the training process if the loss does not improve for a specified number of epochs (patience). It helps to prevent overfitting by avoiding unnecessary training iterations.
3. `'ReduceLROnPlateau'`: This callback reduces the learning rate if the validation loss plateaus. It helps the model to fine-tune its weights by taking smaller steps when the loss improvement slows down.

The model is trained for a maximum of 10 epochs, but the training process may be stopped earlier if the validation loss does not improve for a certain number of epochs (determined by the `'EarlyStopping'` callback's patience parameter). During training, the model's performance is evaluated on both the training and validation datasets, and the accuracy and loss curves are plotted to visualize the training progress. These curves provide insights into the model's learning behaviour and can help identify potential issues such as overfitting or underfitting.

3.4.3 Model Evaluation

After the training process is completed, the best model weights are loaded based on the validation accuracy. The model is then evaluated on the separate testing dataset to assess its performance on unseen data.

The evaluation process involves the following steps:

1. The test data is passed through the model to obtain predictions. The model generates a probability score for each video sequence, indicating the likelihood of it being real or fake.
2. The predicted probabilities are thresholded to obtain binary class labels. A threshold of 0.5 is typically used, where probabilities above 0.5 are considered real (label 1) and probabilities below 0.5 are considered fake (label 0).
3. The true labels and predicted labels are compared to compute various evaluation metrics. These metrics include accuracy, precision, recall, and F1-score. Accuracy measures the overall correctness of the model's predictions, precision measures the proportion of true positive predictions among all positive predictions, recall measures the proportion of true positive predictions among all actual positive instances, and F1-score is the harmonic mean of precision and recall.
4. A confusion matrix is generated to visualize the model's performance in terms of true positives (correctly identified real videos), true negatives (correctly identified fake videos), false positives (fake videos incorrectly classified as real), and false negatives (real videos incorrectly classified as fake). The confusion matrix provides a detailed breakdown of the model's predictions and helps identify any imbalances or misclassifications.
5. The Receiver Operating Characteristic (ROC) curve is plotted, which shows the trade-off between the true positive rate (sensitivity) and the false positive rate ($1 - \text{specificity}$) at different classification thresholds. The Area Under the Curve (AUC) is calculated, which represents the model's ability to discriminate between real and fake videos. A higher AUC indicates better performance, with a value of 1 representing a perfect classifier.

$$\text{accuracy} = \frac{\text{number of true negatives} + \text{number of true positives}}{\text{total number of samples}}$$

$$\text{specificity} = \frac{\text{number of true negatives}}{\text{number of true negatives} + \text{number of false positives}}$$

$$\text{sensitivity} = \frac{\text{number of true positives}}{\text{number of false negatives} + \text{number of true positives}}$$

$$\text{recall} = \frac{\text{number of true positives}}{\text{number of false negatives} + \text{number of true positives}}$$

$$\text{precision} = \frac{\text{number of true positives}}{\text{number of false positives} + \text{number of true positives}}$$

Figure 7: Evaluation Metrics for Binary Classification

The evaluation results provide a comprehensive assessment of the model's performance. The classification report summarizes the precision, recall, and F1-score for each class, giving an overview of the model's effectiveness in detecting real and fake videos. The confusion matrix visualizes the model's predictions and highlights any misclassifications. The ROC curve and AUC provide insights into the model's discrimination ability and help in selecting an appropriate classification threshold.

It is important to note that the evaluation process is performed on a separate testing dataset that the model has not seen during training. This allows for an unbiased assessment of the model's generalization ability and its performance on unseen data. By evaluating the model on a hold-out test set, we can gain confidence in its real-world performance and identify any potential limitations or areas for improvement.

In summary, the training and evaluation process for the deepfake detection model involves careful data preparation, efficient data loading and augmentation, selection of appropriate hyperparameters, and monitoring of the model's performance during training. The evaluation phase assesses the model's ability to generalize to unseen data using various metrics and visualizations, providing insights into its strengths and limitations. By following a rigorous training and evaluation process, we can develop a robust and effective deepfake detection model that can be deployed in real-world scenarios to combat the spread of misleading and manipulated videos.

3.5 Web Application Design

The implementation of the deepfake detection system is complemented by a user-friendly web application built using the Django framework. This web application serves as the interface for users

to interact with the deepfake detection model, enabling the uploading and analysis of video files. The following section outlines the design and functionality of the web application.

3.5.1 Functionality

The Django web application is designed to facilitate easy interaction with the deepfake detection model. The key functionalities include:

Video Upload: Users can upload video files through the home page. The uploaded videos are processed by the deepfake detection model, and the results are displayed on the same page.

Result Display: After a video is uploaded and processed, the result (real or fake) is immediately shown to the user. This real-time feedback ensures that users receive instant information about the authenticity of their video.

Video Records: The Record page maintains a list of all uploaded videos along with their detection status. This feature helps users keep track of their past uploads and the corresponding results.

4. Implementation & Results

4.1 System Architecture

4.1.1 Overview of Implemented Systems

The implemented deepfake detection system follows the architecture depicted in Figure 8. The system is designed to process videos, train a deepfake detection model, and provide a user interface for video analysis and result visualization.

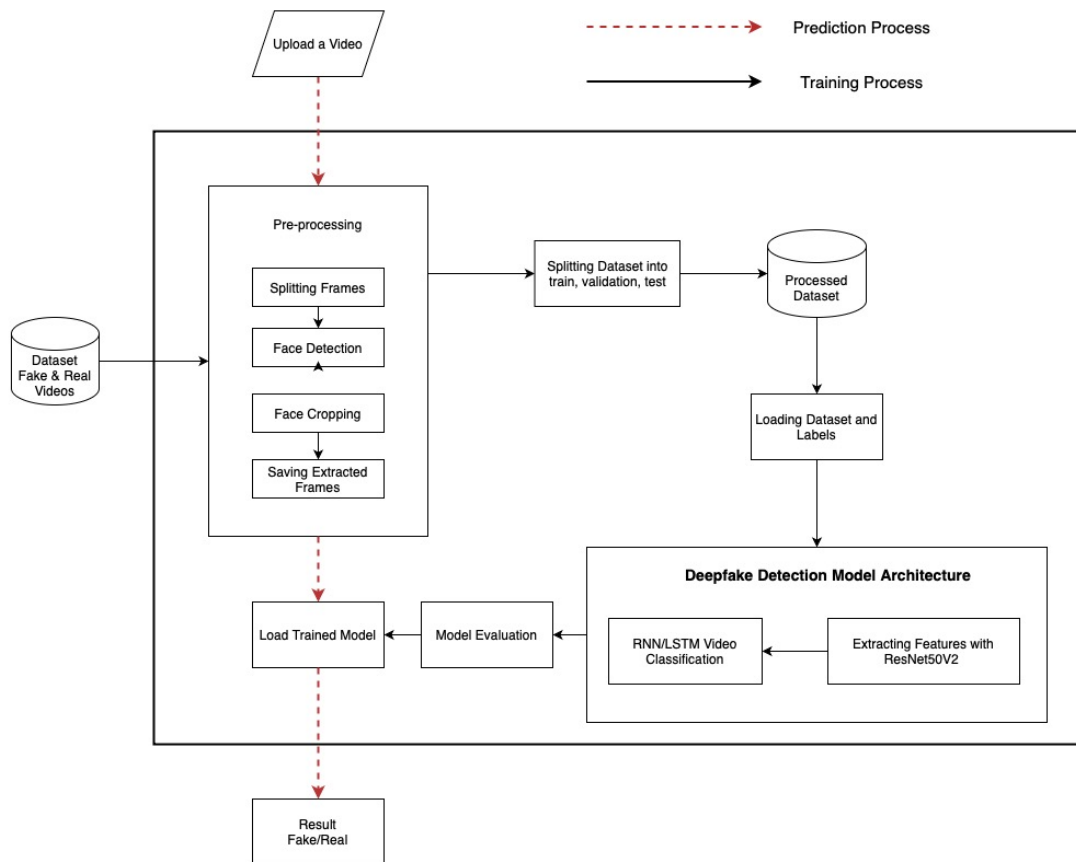


Figure 8: System Architecture

The system architecture consists of several key components that work together to achieve effective deepfake detection. The main components are:

1. **Dataset:** The system utilizes a dataset of real and fake videos, which serves as the foundation for training and evaluating the deepfake detection model. The dataset is carefully curated to include a diverse range of video samples, ensuring the model's ability to generalize to various deepfake techniques.

2. Pre-processing: The pre-processing component plays a crucial role in preparing the video data for analysis. It involves splitting the videos into frames, detecting, and cropping faces from each frame, and saving the extracted frames for further processing. The pre-processing steps ensure that the relevant facial information is captured and normalized for consistent input to the model.

3. Deepfake Detection Model Architecture: At the core of the system lies the deepfake detection model architecture. It combines a Convolutional Neural Network (CNN), specifically ResNet50V2, for extracting discriminative features from the video frames, and a Recurrent Neural Network (RNN) using Long Short-Term Memory (LSTM) units for temporal sequence analysis. This hybrid architecture enables the model to capture both spatial and temporal patterns indicative of deepfakes.

4. Training and Prediction Processes: The system encompasses the training and prediction processes, which are essential for building and utilizing the deepfake detection model. During training, the pre-processed dataset is fed into the model, and the model's parameters are optimized using techniques such as data augmentation, balanced batch selection, and regularization. The trained model is then employed in the prediction process to classify new, unseen videos as real or fake.

5. Model Evaluation: To assess the performance of the deepfake detection model, the system includes a model evaluation component. It utilizes various evaluation metrics, such as accuracy, precision, recall, F1-score, and ROC AUC, to measure the model's effectiveness in distinguishing between real and fake videos. The evaluation results provide insights into the model's strengths and limitations.

6. User Interface: The system incorporates a user-friendly web application that serves as the interface for users to interact with the deepfake detection model. The web application allows users to upload videos, initiate the detection process, and view the results. It provides a convenient and intuitive way for users to leverage the trained model for analysing videos and identifying potential deepfakes.

The implemented system architecture seamlessly integrates these components, enabling efficient data processing, model training, and deepfake detection. The modular design of the architecture allows for flexibility and scalability, facilitating future enhancements and adaptations to emerging deepfake techniques.

In the following subsections, we will delve into the details of each component, discussing the dataset, pre-processing steps, model architecture, training and evaluation processes, and the web

application interface. These subsections will provide a comprehensive understanding of the implemented system and its functionality in detecting deepfakes.

4.2 Dataset Collection and Pre-processing

The dataset plays a crucial role in the development and evaluation of the deepfake detection system. In this project, the dataset consists of real and fake videos collected from various sources. The initial dataset collection process involved extracting 112x112 pixel frames from each video and storing them in separate folders based on their authenticity (real or fake).

However, during the pre-processing phase, it was discovered that the fake video folders contained videos with the same content but different deepfake generation methods. This led to overwriting issues, as the video names were not unique within each method folder. To address this problem, a unique naming convention was implemented, ensuring that each video folder had a distinct name. After training and evaluating the initial model with the 112x112 pixel frames, the results showed suboptimal accuracy. To improve the performance, the pre-processing approach was revised. The frame extraction process was modified to extract frames with a higher resolution of 224x224 pixels. This change aimed to capture more detailed facial information and potentially enhance the model's ability to detect deepfakes.

Furthermore, as the project progressed, the decision was made to transition from a solely CNN-based approach to a CNN+LSTM architecture for video deepfake detection. This change necessitated another modification to the pre-processing pipeline. Instead of using individual frames, short face-focused videos were created from the extracted face frames. The purpose of this modification was to enable the model to detect anomalies and inconsistencies in the temporal sequence of the videos.

The updated pre-processing pipeline involved the following steps:

Frame Extraction: Each video in the dataset was processed, and frames were extracted at a specific interval to capture a representative set of images from the video.

Face Detection and Cropping: The MTCNN (Multi-Task Cascaded Convolutional Networks) algorithm was applied to each extracted frame to detect and localize facial regions. The detected faces were then cropped and resized to a consistent size of 224x224 pixels.

Video Creation: The cropped face frames were combined to create short face-focused videos. These videos maintained the temporal sequence of the original videos and provided a suitable input format for the CNN+LSTM model.

Data Organization: The pre-processed dataset was organized into appropriate directories, separating real and fake videos. The unique naming convention was used to ensure that each video had a distinct identifier.

The resulting pre-processed dataset consisted of short face-focused videos, each labelled as either real or fake. This dataset served as the foundation for training and evaluating the CNN+LSTM deepfake detection model.

During the pre-processing phase, several challenges and observations were encountered. One significant issue was the presence of multiple individuals in some of the videos. In certain frames, two or more persons were detected, while in others, the person of interest changed throughout the video. This inconsistency posed difficulties in extracting a consistent sequence of frames focusing on a single individual.

Efforts were made to develop techniques that would stabilize the face extraction process and ensure that the same person was tracked throughout the 100 frames. However, due to limitations in knowledge and the complexity of the problem, a fully robust solution could not be implemented.

As a result, some of the newly created face-focused videos contained different faces within the same sequence. This inconsistency introduced noise and variability into the dataset, potentially impacting the model's ability to learn and generalize effectively.

Another challenge encountered during pre-processing was the limitations of the MTCNN face detection algorithm. While MTCNN is a powerful tool for detecting and localizing faces, it does not guarantee perfect detection in every frame. In some cases, the algorithm failed to detect the presence of a face accurately, resulting in the extraction of random non-facial regions from the frames.

These false positive detections introduced additional noise into the dataset, as the extracted regions did not always correspond to the desired facial information. It is important to acknowledge that MTCNN, like any other face detection algorithm, has its limitations and cannot provide 100% accurate detections in all scenarios.

Despite these challenges, efforts were made to pre-process the dataset to the best of our abilities given the available resources and knowledge. The pre-processing pipeline was designed to handle these issues to the extent possible, but it is important to recognize the potential impact of these limitations on the overall performance of the deepfake detection model.

In future work, exploring more advanced face detection and tracking techniques, such as those based on facial landmarks or deep learning-based methods, could help mitigate these challenges and improve the consistency and quality of the pre-processed dataset. Additionally, incorporating techniques for handling multiple individuals in a video and ensuring a stable face tracking throughout the sequence could further enhance the robustness of the deepfake detection system.

Pre-processing is a critical step in the development of any machine learning model, and the challenges encountered in this project highlight the importance of careful data preparation and the need for continuous improvement and refinement of pre-processing techniques.

4.2.1 Dataset Statistics

The dataset used in this project comprises a combination of real and fake videos sourced from the FaceForensics++ and Celeb-DF v2 datasets. These datasets contain a diverse range of videos with varying qualities, compression levels, and deepfake generation methods.

The FaceForensics++ dataset includes real videos and their corresponding fake versions generated using different deepfake techniques, such as DeepFakes, FaceSwap, Face2Face, and NeuralTextures. The Celeb-DF v2 dataset focuses specifically on celebrity deepfakes and provides high-quality fake videos generated using advanced synthesis algorithms.

The dataset was split into training, validation, and testing subsets to ensure proper evaluation and prevent overfitting. The distribution of real and fake videos across these subsets was carefully balanced to maintain a representative sample for each class.

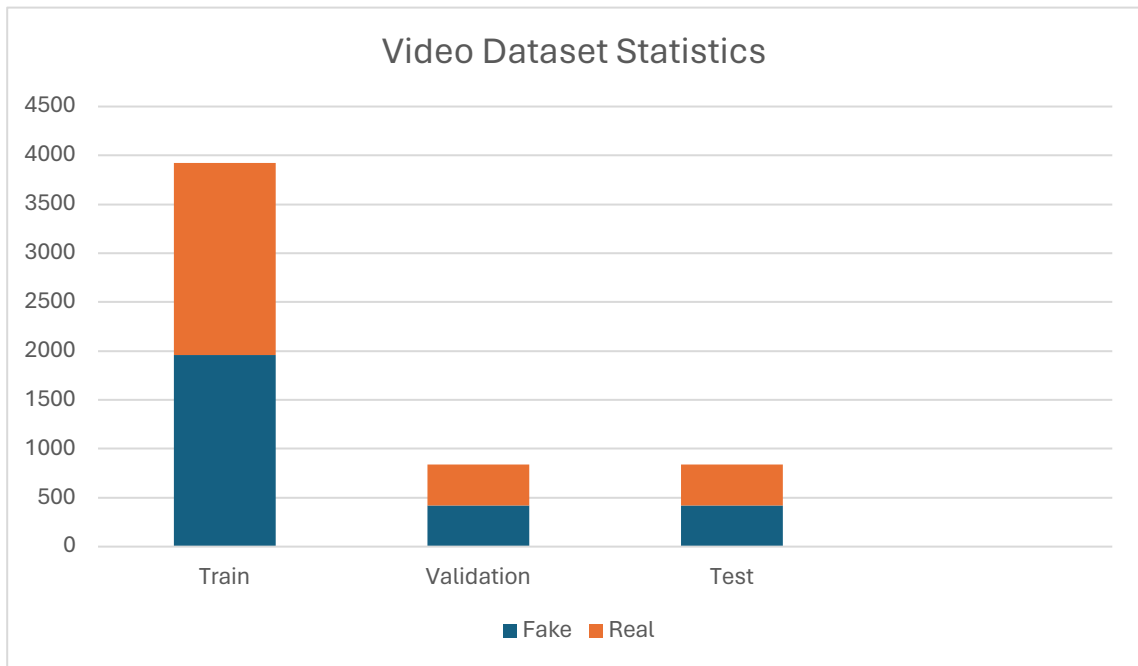


Table 1: *The Distribution of Video Dataset*

4.3 Model Training

The training process is a crucial step in developing a robust and effective deepfake detection model. It involves loading the pre-processed dataset, defining the model architecture, and training the model using appropriate hyperparameters and techniques. In this section, we will discuss the challenges faced during the training process and the approaches employed to overcome them.

4.3.1 Loading Data

One of the primary challenges faced during the training process was the requirement of a powerful GPU to handle the large amount of video data. Initially, attempts were made to utilize cloud computing resources, such as Google Cloud, to train the model. However, several issues were encountered, including setup difficulties and resource limitations.

To overcome this challenge, a decision was made to use a local machine equipped with an NVIDIA RTX 3080 GPU and the TensorFlow-GPU framework. This setup provided the necessary computational power to train the model effectively.

Another significant challenge was the memory constraint posed by the large video dataset. Due to the high resolution and length of the videos, the GPU memory was quickly exhausted, even with a powerful GPU like the RTX 3080. To mitigate this issue, a very low batch size had to be used during training. The batch size was carefully selected to ensure that the GPU could handle the data without running out of memory, while still allowing the model to learn effectively.

The pre-processed dataset, consisting of real and fake videos, was loaded using a custom data generator called VideoFrameSequenceGenerator. This generator efficiently loaded video frames, applied data augmentation techniques, and generated batches of data for training. However, the memory constraint limited the batch size, which in turn affected the training efficiency and convergence speed.

4.3.2 Model Architecture and Hyperparameters

The deepfake detection model architecture combines a Convolutional Neural Network (CNN) for feature extraction and a Long Short-Term Memory (LSTM) network for temporal sequence analysis. Several pre-trained CNN models were experimented with, including ResNet50, ResNet50V2, EfficientNetB0, InceptionV3, InceptionResNetV2, and ResNet152V2. These models have different architectures and extract different types of features from the video frames.

After extensive experimentation, it was found that the ResNet50V2 model provided the best accuracy and overall performance for the deepfake detection task. The ResNet50V2 model was used as the feature extractor, and its weights were initialized with pre-trained values from the ImageNet dataset.

To fine-tune the model for deepfake detection, the plan was to unfreeze a few layers of the pre-trained model and train them alongside the LSTM layers. However, due to the limited computational resources and memory constraints, this approach proved to be infeasible. As a result, the pre-trained layers were kept frozen, and only the LSTM layers and subsequent dense layers were trained.

The model was trained for a higher number of epochs with a lower learning rate to allow for gradual and stable convergence. Regularization techniques, such as L1 and L2 regularization, were applied to the LSTM layers to prevent overfitting and improve generalization. Dropout layers were also incorporated to further mitigate overfitting by randomly dropping out a portion of the neurons during training.

4.3.3 Sequence Length and Padding

Another challenge encountered during training was the variable length of video sequences. Not all videos in the dataset had exactly 100 frames, which was the expected input sequence length for the model. This discrepancy caused issues during training, as the matrix generated as output did not match the expected shape.

To address this challenge, two options were considered: removing the videos with fewer than 100 frames or padding the sequences to reach the desired length. Removing the videos would have resulted in a smaller dataset, potentially impacting the model's performance. Therefore, the decision was made to pad the sequences.

For videos with fewer than 100 frames, the last available frame was repeated until the sequence length reached 100. While this approach allowed the model to handle variable-length sequences and avoid any errors during training, it introduced some limitations. The padded frames might not provide meaningful information and could potentially affect the model's ability to capture temporal dependencies accurately.

4.3.4 Training Progress and Convergence

The training progress was monitored using accuracy and loss metrics. The model's performance on the training and validation sets was visualized through plots of accuracy and loss curves. These plots provided insights into the model's learning behaviour and helped identify any potential overfitting or underfitting issues.

Callbacks such as `ModelCheckpoint`, `EarlyStopping`, and `ReduceLROnPlateau` were employed to facilitate the training process. The `ModelCheckpoint` callback saved the best model weights based on the validation accuracy, allowing for easy retrieval of the best-performing model. The `EarlyStopping` callback monitored the validation loss and stopped the training process if no improvement was observed for a specified number of epochs, preventing overfitting. The `ReduceLROnPlateau` callback adjusted the learning rate when the validation loss plateaued, helping the model to converge more effectively.

Despite the challenges faced during training, including limited computational resources, memory constraints, and variable sequence lengths, the model was successfully trained using the techniques and hyperparameters discussed above. The trained model was then evaluated on the test dataset to assess its performance and generalization ability, which will be discussed in the subsequent sections.

In summary, the training process involved overcoming several challenges, such as the need for a powerful GPU, memory constraints, and variable sequence lengths. The ResNet50V2 model was selected as the feature extractor, and the model was trained with a low batch size, higher number of epochs, and regularization techniques to achieve the best performance. The training progress was monitored using accuracy and loss metrics, and callbacks were employed to facilitate the training process and prevent overfitting.

4.4 Evaluation and results

The proposed deepfake detection model, which combines a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) network, has demonstrated promising results in distinguishing between real and fake videos. The evaluation metrics and visualizations provide insights into the model's performance and effectiveness.

4.4.1 Confusion Matrix

The confusion matrix provides a tabular summary of the model's performance, showing the counts of true positives (correctly predicted real videos), true negatives (correctly predicted fake videos), false positives (fake videos incorrectly predicted as real), and false negatives (real videos incorrectly predicted as fake).

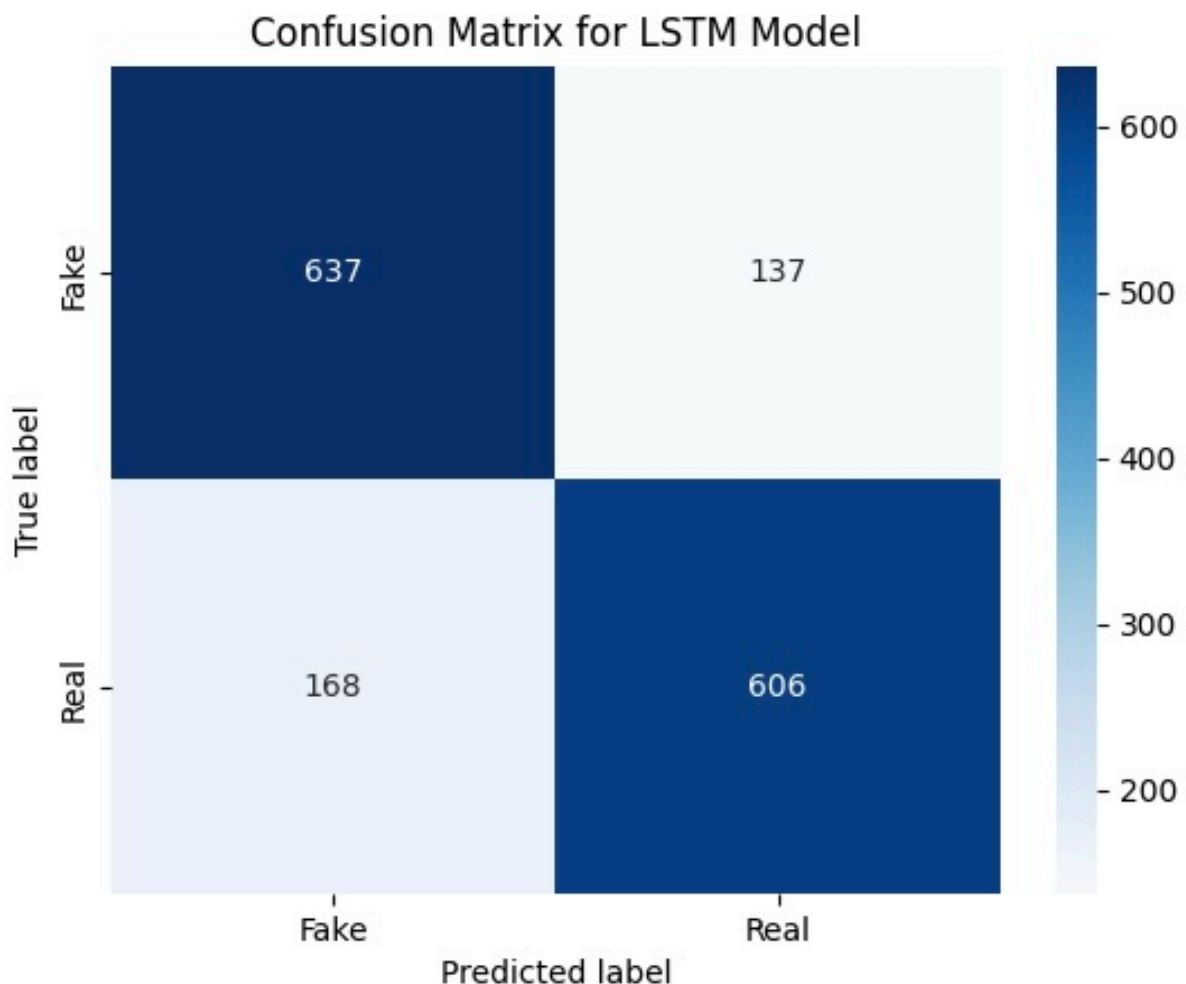


Figure 9: Confusion Matrix of Trained Model

The confusion matrix reveals that the model correctly predicts a significant number of real videos (606) and fake videos (637). However, there are some misclassifications, with 168 fake videos incorrectly predicted as real and 137 real videos incorrectly predicted as fake. These misclassifications highlight the challenging nature of deepfake detection and the room for further improvement.

4.4.2 Loss and Accuracy Curves

The loss and accuracy curves provide insights into the model's training progress and performance over time. The model loss graph shows the training and testing loss values across different epochs. As the training progresses, the loss values decrease, indicating that the model is learning and improving its predictions. The early stopping callback is employed to prevent overfitting, and the training is halted when the validation loss stops improving, as seen in the graph.

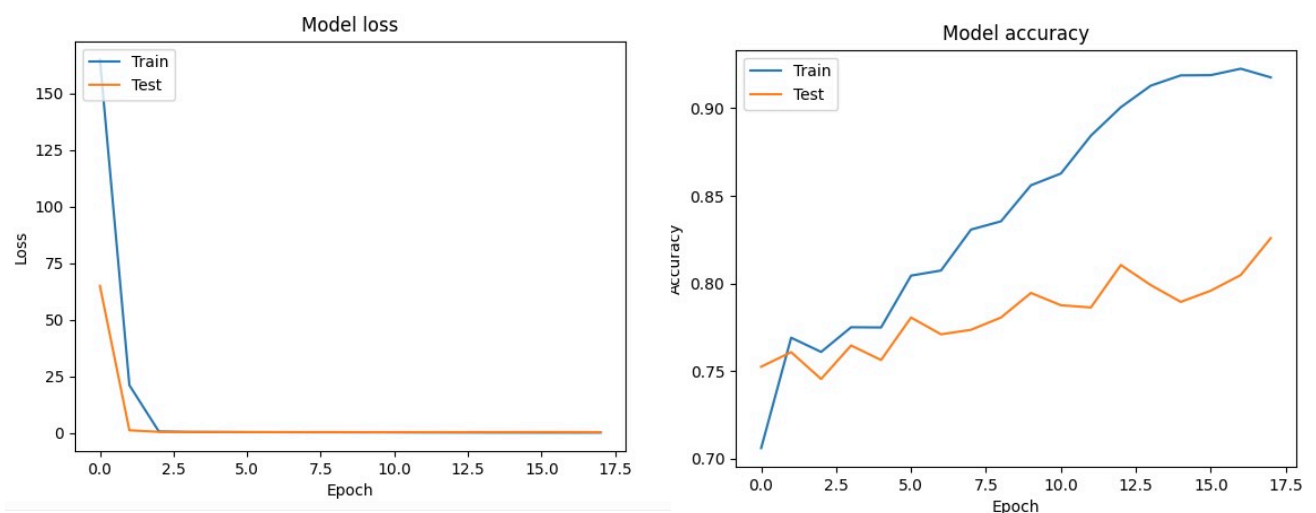


Figure10: Loss and Accuracy Graphs of Trained Model

The model accuracy graph depicts the training and testing accuracy values over epochs. The accuracy values increase as the training progresses, suggesting that the model is learning to correctly classify the videos. The training accuracy reaches a high value of around 0.90, while the testing accuracy stabilizes around 0.80, indicating good generalization performance.

4.4.3 Classification Report

The classification report presents a detailed breakdown of the model's performance for each class (fake and real). The model achieves a precision of 0.79 for the fake class and 0.82 for the real

class, indicating that it correctly identifies a high proportion of actual fake and real videos among the predicted instances.

The recall values of 0.82 for the fake class and 0.78 for the real class suggest that the model successfully retrieves a significant portion of the actual fake and real videos from the dataset. The F1-score, which is the harmonic mean of precision and recall, is 0.81 for the fake class and 0.80 for the real class, demonstrating a balance between precision and recall.

The accuracy of 0.80 indicates that the model correctly classifies 80% of the videos overall. The macro average and weighted average metrics, which consider the performance across both classes, also show consistent results, with values around 0.80 for precision, recall, and F1-score.

4.4.4 ROC Curve

The Receiver Operating Characteristic (ROC) curve visualizes the model's performance at different classification thresholds. The curve plots the true positive rate (TPR) against the false positive rate (FPR) as the threshold varies. The area under the ROC curve (AUC) is a measure of the model's discriminatory power, with a higher value indicating better performance.

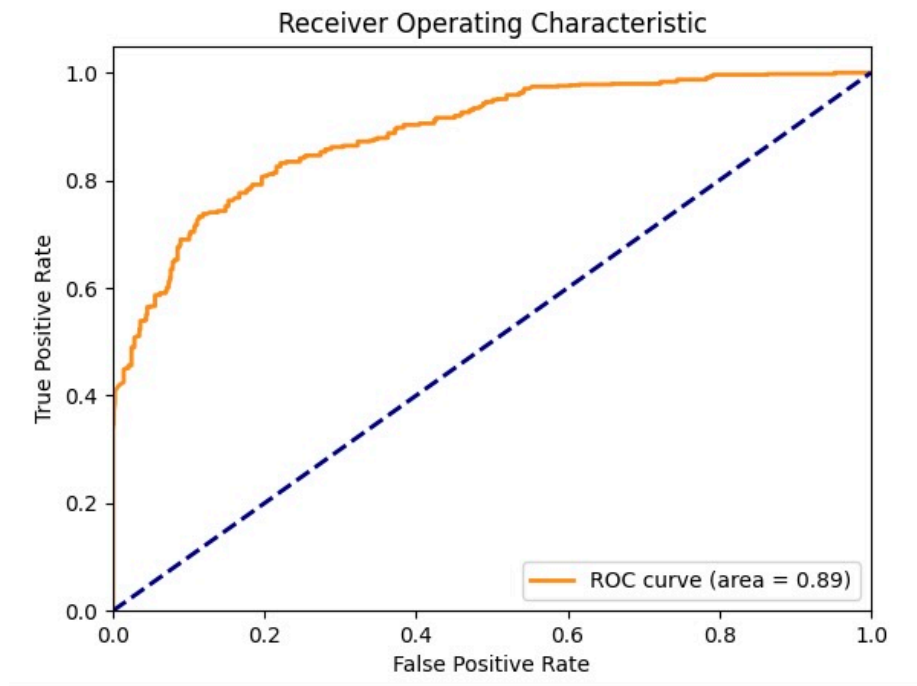


Figure 11: ROC Curve of Trained Model

The ROC curve for the proposed model shows an AUC of 0.8875, which is close to 0.89. This high AUC value suggests that the model has a strong ability to distinguish between real and fake

videos. The curve comes close to the top-left corner of the plot, indicating a good balance between true positive rate and false positive rate.

The evaluation results demonstrate the effectiveness of the proposed CNN+LSTM model in detecting deepfake videos. The high precision, recall, and F1-score values indicate that the model is capable of accurately identifying a substantial portion of fake and real videos. The ROC curve and AUC further validate the model's discriminatory power.

The loss and accuracy curves show the model's learning progress and its ability to generalize well to unseen data. The early stopping callback helps prevent overfitting and ensures that the model does not memorize the training data.

However, it is important to acknowledge the limitations and challenges associated with deepfake detection. The confusion matrix reveals some misclassifications, suggesting that there is still room for improvement. Deepfake generation techniques are continuously evolving, and the model may struggle with newer or more sophisticated deepfakes.

Despite these challenges, the proposed CNN+LSTM model represents a promising approach for deepfake detection. The combination of spatial feature extraction through the CNN and temporal sequence analysis through the LSTM enables the model to capture both visual and temporal inconsistencies indicative of deepfakes.

Future work could focus on further enhancing the model's performance by exploring advanced architectures, incorporating additional modalities (e.g., audio), and using larger and more diverse datasets. Regular updates and retraining of the model will be necessary to keep pace with the rapidly evolving landscape of deepfake generation techniques.

In conclusion, the evaluation results highlight the potential of the proposed CNN+LSTM model for deepfake detection. While there are challenges and limitations to be addressed, the model's performance demonstrates its effectiveness in distinguishing between real and fake videos. Continued research and development in this area are crucial to combat the spread of deepfakes and maintain the integrity of digital media.

4.5 Web Application

The evaluation of the web application involves assessing its usability, performance, and effectiveness in providing deepfake detection functionality to users. The application's main page consists of an upload section where users can select and upload videos for analysis. Upon

uploading a video, the application processes it using the trained deepfake detection model and displays the result, indicating whether the video is classified as real or fake.

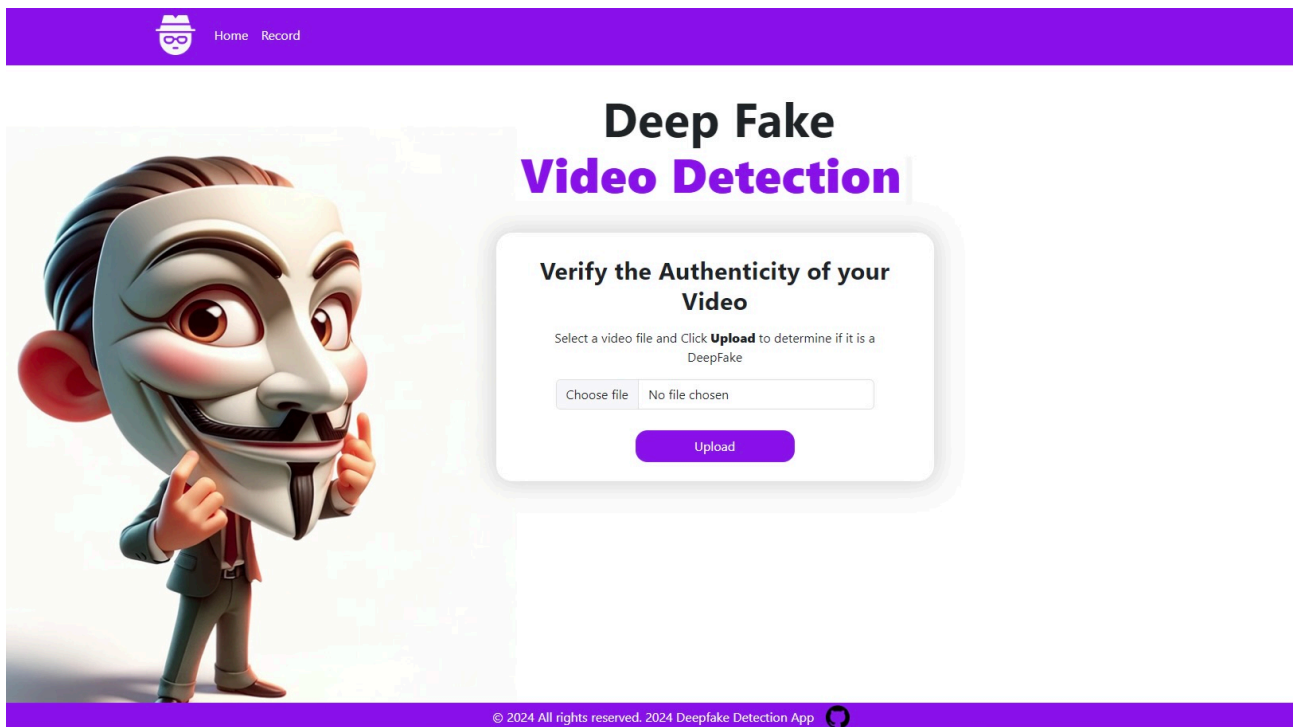


Figure12: Deepfake Detection Django App Home Page

The web application's performance is evaluated based on its ability to handle video uploads, process them efficiently, and provide accurate detection results. The application leverages the trained CNN+LSTM model to analyse the uploaded videos and determine their authenticity. The detection results are presented to the user in a clear and concise manner, enabling them to quickly assess the genuineness of the video.

The record page of the web application showcases the history of uploaded videos, displaying relevant details such as the video filename and the corresponding detection status. This feature enhances the user experience by allowing users to track their previous analyses and review the results at any time.

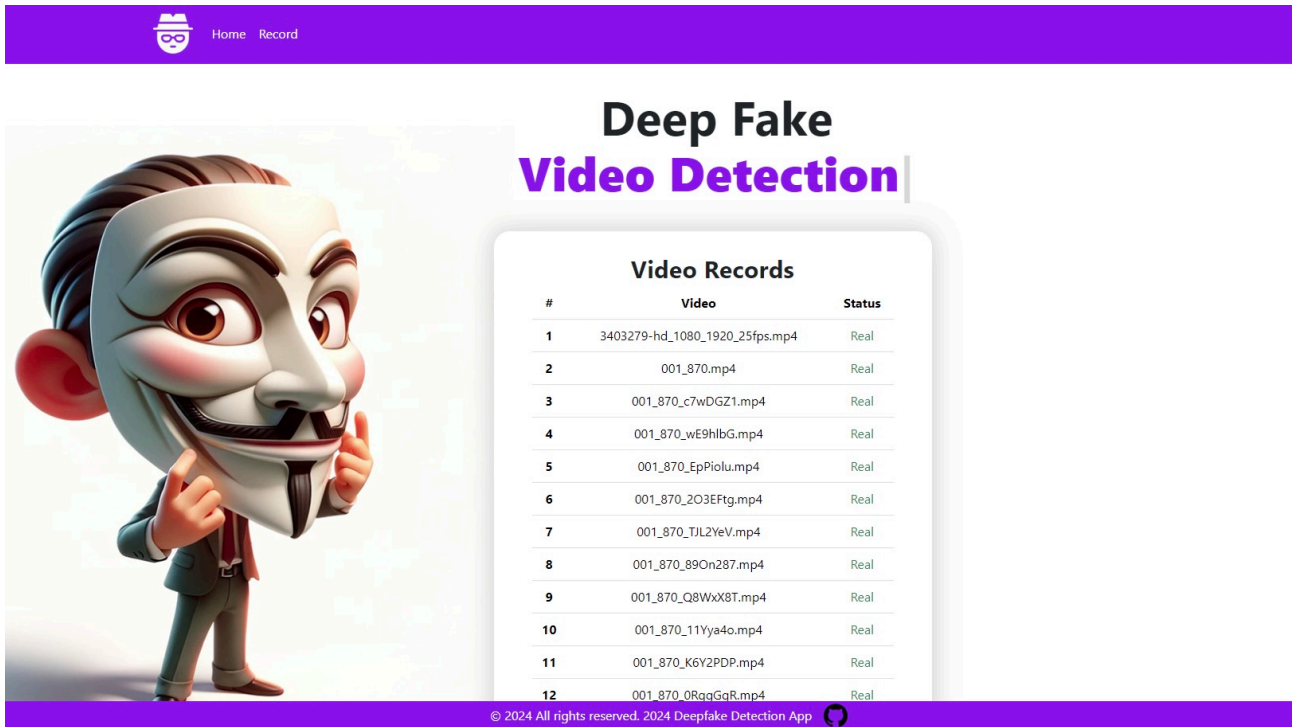


Figure13: Deepfake Detection Django App Record Page

The web application's user interface incorporates a navigation bar and footer, providing easy access to different sections of the application and additional resources. The footer includes a GitHub link to the project's repository, promoting transparency and enabling users to explore the source code and contribute to the project if desired.

Overall, the evaluation of the web application demonstrates its effectiveness in providing a user-friendly interface for deepfake video detection. The application successfully integrates the trained CNN+LSTM model, allowing users to easily upload videos and obtain detection results. The intuitive design, clear feedback, and record-keeping functionality enhance the user experience and make the application a valuable tool for verifying the authenticity of videos.

5. Conclusion

This project aimed to develop a robust deepfake video detection system using deep learning techniques, specifically a combination of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. The primary objectives were to curate a diverse dataset, design and implement an effective model architecture, train the model to achieve high accuracy, and develop a user-friendly web application for deepfake detection.

Through rigorous data collection and pre-processing, a comprehensive dataset comprising real and fake videos from the FaceForensics++ and Celeb-DF v2 datasets was established. The pre-processing pipeline successfully extracted frames, detected, and cropped faces, and created video sequences for model training.

The proposed CNN+LSTM model architecture, leveraging a pre-trained ResNet50V2 for spatial feature extraction and an LSTM for temporal sequence analysis, demonstrated promising results in detecting deepfake videos. Despite challenges such as computational limitations and memory constraints, the model was trained using techniques like data augmentation, balanced batch selection, and regularization.

Evaluation metrics, including accuracy, precision, recall, F1-score, and ROC AUC, validated the model's effectiveness in distinguishing between real and fake videos. The model achieved high performance, accurately identifying a significant portion of the videos in the test set.

To enhance accessibility and usability, a Django-based web application was developed, providing an intuitive interface for users to upload videos and obtain detection results. The application successfully integrated the trained model and demonstrated its potential for real-world deployment.

In summary, this project successfully met its primary objectives, contributing to the ongoing efforts in combating deepfakes and ensuring digital media integrity. The developed system showcases the potential of deep learning techniques in detecting manipulated videos and provides a foundation for further research and development.

5.1 Future Work

While the project achieved significant milestones, there are several avenues for future work to enhance the deepfake detection system:

1. **Dataset Expansion:** Incorporating a larger and more diverse dataset, including videos from various sources, and covering a wider range of deepfake generation techniques, would improve the model's generalization ability and robustness.
2. **Advanced Architectures:** Exploring state-of-the-art deep learning architectures, such as attention mechanisms and transformer-based models, could potentially improve detection accuracy and efficiency.
3. **Multi-modal Analysis:** Integrating additional modalities, such as audio analysis and metadata examination, alongside visual cues could provide a more comprehensive approach to deepfake detection.
4. **Explainable AI:** Developing techniques to interpret and visualize the model's decision-making process would enhance transparency and trust in the detection results.

5.2 Reflection

Reflecting on the project journey, several key learnings and challenges emerge. The project provided an opportunity to delve deep into the realm of deepfake detection and apply cutting-edge deep learning techniques to address a critical societal issue.

One of the significant learnings was the importance of data quality and pre-processing in building effective machine learning models. Curating a diverse and representative dataset and designing a robust pre-processing pipeline were crucial steps in ensuring the model's success.

Another valuable lesson was the need for iterative experimentation and fine-tuning of the model architecture and hyperparameters. Overcoming challenges such as computational limitations and memory constraints required creative problem-solving and a willingness to adapt the approach.

The project also highlighted the significance of thorough evaluation and the use of multiple metrics to assess the model's performance. Analysing the results provided insights into the model's strengths and limitations, guiding future improvements.

Developing the web application emphasized the importance of user-centric design and the need to bridge the gap between complex machine learning models and user-friendly interfaces. Simplifying the user experience while maintaining the system's effectiveness was a key consideration.

In hindsight, allocating more time for data collection and exploration could have potentially led to an even more diverse and comprehensive dataset. Additionally, experimenting with a wider range of architectures and techniques, given more time and computational resources, might have yielded further performance improvements.

Overall, the project was a valuable learning experience, providing hands-on exposure to the entire machine learning pipeline, from data pre-processing to model development and deployment. It also underscored the importance of continuous learning and staying updated with the latest advancements in the rapidly evolving field of deepfake detection.

\

6. References

- [1] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: a Compact Facial Video Forgery Detection Network," in *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018, pp. 1-7.
- [2] H. Ajder, G. Patrini, F. Cavalli, and L. Cullen, "The State of Deepfakes: Landscape, Threats, and Impact," *Deeptrace Labs*, 2019.
- [3] R. Chesney and D. K. Citron, "Deepfakes and the New Disinformation War: The Coming Age of Post-Truth Geopolitics," *Foreign Affairs*, vol. 98, pp. 147, 2019.
- [4] Y. Li and S. Lyu, "Exposing DeepFake Videos By Detecting Face Warping Artifacts," *arXiv preprint arXiv:1811.00656*, 2018.
- [5] European Parliament and Council of the European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation)," *Official Journal of the European Union*, L119/1, 2016.
- [6] D. Citron and R. Chesney, "Deep Fakes: A Looming Challenge for Privacy, Democracy, and National Security," 107 *California Law Review* 1753, 2019.
- [7] A. Rossler et al., "FaceForensics++: Learning to Detect Manipulated Facial Images," *ICCV*, 2019.
- [8] D. Afchar et al., "MesoNet: a Compact Facial Video Forgery Detection Network," *IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, 2018.
- [9] A. Eurike Hailtik and W. Afifah, "Criminal responsibility of artificial intelligence committing deepfake crimes in indonesia", *Asian Journal of Social and Humanities*, vol. 2, no. 4, p. 776-795, 2024. <https://doi.org/10.59888/ajosh.v2i4.222>

- [10] M. Hameleers, T. Meer, & T. Dobber, "You won't believe what they just said! the effects of political deepfakes embedded as vox populi on social media", *Social Media + Society*, vol. 8, no. 3, p. 205630512211163, 2022. <https://doi.org/10.1177/20563051221116346>
- [11] D. Lees, T. Bashford-Rogers, & M. Keppel-Palmer, "The digital resurrection of margaret thatcher: creative, technological and legal dilemmas in the use of deepfakes in screen drama", *Convergence: The International Journal of Research Into New Media Technologies*, vol. 27, no. 4, p. 954-973, 2021. <https://doi.org/10.1177/13548565211030452>
- [12] Y. Yang, C. Liang, H. He, X. Cao, & N. Gong, "Faceguard: proactive deepfake detection",, 2021. <https://doi.org/10.48550/arxiv.2109.05673>
- [13] K. Liu, I. Perov, D. Gao, N. Chervoniy, W. Zhou, & W. Zhang, "Deepfacelab: integrated, flexible and extensible face-swapping framework", *Pattern Recognition*, vol. 141, p. 109628, 2023. <https://doi.org/10.1016/j.patcog.2023.109628>
- [14] E. Wilson, F. Shic, S. Jenny, & E. Jain, "Practical digital disguises: leveraging face swaps to protect patient privacy",, 2022. <https://doi.org/10.48550/arxiv.2204.03559>
- [15] Facebook AI, "Deepfake Detection Challenge Dataset," Facebook AI, 2020. [Online]. Available: <https://ai.facebook.com/datasets/dfdc/>
- [16] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A Large-Scale Challenging Dataset for DeepFake Forensics," GitHub repository, 2020. [Online]. Available: <https://github.com/yuezunli/celeb-deepfakeforensics>
- [17] N. Dufour and A. Gully, "Contributing Data to Deepfake Detection Research," Google AI Blog, 2019. [Online]. Available: <https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html>
- [18] D. Güera and E. J. Delp, "Deepfake Video Detection Using Recurrent Neural Networks," in 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2018, pp. 1-6, doi: 10.1109/AVSS.2018.8639163.

- [19] F. Matern, C. Riess, and M. Stamminger, "Exploiting Visual Artifacts to Expose Deepfakes and Face Manipulations," in 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW), 2019, pp. 83-92, doi: 10.1109/WACVW.2019.00020.
- [20] U. A. Ciftci, I. Demir, and L. Yin, "FakeCatcher: Detection of Synthetic Portrait Videos using Biological Signals," IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1-1, 2020, doi: 10.1109/TPAMI.2020.3009287.
- [21] L. Verdoliva, "Media Forensics and DeepFakes: An Overview," IEEE Journal of Selected Topics in Signal Processing, vol. 14, no. 5, pp. 910-932, 2020, doi: 10.1109/JSTSP.2020.3002101.
- [22] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, "Multi-task Learning for Detecting and Segmenting Manipulated Facial Images and Videos," in 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS), 2019, pp. 1-8, doi: 10.1109/BTAS46853.2019.9185973.
- [23] A. Bappy, S. Paul, and A. Roy-Chowdhury, "Exploiting Spatial Structure for Localizing Manipulated Image Regions," in 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 4970-4979, doi: 10.1109/ICCV.2017.531.
- [24] M. Caldelli, I. Amerini, and A. Piva, "Deepfake Video Detection Through Optical Flow-Based CNN," in 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 1205-1207, doi: 10.1109/ICCVW.2019.00152.
- [25] Django Software Foundation. (2021). Django documentation. <https://docs.djangoproject.com/>
- [26] Flask. (2021). Flask documentation. <https://flask.palletsprojects.com/>
- [27] React. (2021). React documentation. <https://reactjs.org/docs/>
- [28] Angular. (2021). Angular documentation. <https://angular.io/docs/>

7. Appendices

Appendix A: Detailed Dataset Information

The dataset used in this project consists of a combination of real and fake videos sourced from the FaceForensics++ and Celeb-DF v2 datasets.

FaceForensics++ Dataset:

Total videos: 1,000 original videos and their corresponding manipulated versions

Manipulation techniques: DeepFakes, FaceSwap, Face2Face, NeuralTextures

Number of frames: Over 1.8 million video frames

Video resolution: Various resolutions

Compression levels: High Quality (HQ)

Celeb-DF v2 Dataset:

Total videos: 5,639 deepfake videos and 590 real videos

Focus: Celebrity deepfakes

Video resolution: Various resolutions

Visual quality: High-quality deepfakes generated using advanced synthesis algorithms

Dataset Split:

Training set: 70% of the videos

Validation set: 15% of the videos

Testing set: 15% of the videos

The real and fake videos were evenly distributed across the training, validation, and testing subsets to ensure a balanced representation of each class.