**Deepfake Detection using Deep Neural Networks**

**Deepfake Detection using Deep Neural Networks**

Submitted in partial fulfillment of the requirements

of the degree of

**M. Tech. Computer Engineering**

By

**Runmay Rajendra Shirsat 60008240006**

Guide:

**Dr. Chetashshri Bhadane**



University of Mumbai

2024-2025

|  |
| --- |
|  |

**CERTIFICATE**

This is to certify that the mini project entitled **“Deepfake detection using Deep Neural Networks”** is a bonafide work of **“Runmay Rajendra Shirsat” (60008240006.)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of M. Tech. in Computer Engineering

**(Dr. Chetashshri Bhadane)**

**Internal Guide**

**(Dr. Meera Narvekar) (Dr. Hari Vasudevan)**

**Head of Department Principal**

**Project Report Approval for M.Tech.**

This project report entitled *Deepfake Detection Using Deep Neural Networks by Runmay Shirsat* is approved for the degree of ***M.Tech. in Computer Engineering.***

Examiners

1.---------------------------------------------

2.---------------------------------------------

Date:

Place:

**Declaration**

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

-----------------------------------------

(Signature)

**(Runmay Rajendra Shirsat 60008240006)**

Date:

**Abstract**

Deepfake technology poses significant challenges due to its ability to create hyper-realistic manipulated media, raising concerns in misinformation, cybersecurity, and privacy. This paper presents a hybrid model combining EfficientNet for feature extraction and Gated Recurrent Units for temporal sequence analysis. Using the Celeb-DF and DFDC datasets, the approach preprocesses videos by extracting frames and cropping facial regions to enhance detection accuracy.The preprocessing stage ensures the quality of input data by isolating facial features from video frames. EfficientNet extracts spatial details, while GRU layers capture temporal dependencies across frames. These steps allow the model to effectively identify deepfake patterns.A key focus is the comparative evaluation of the model’s performance on the Celeb-DF and DFDC datasets, which differ in complexity and manipulation techniques. The Celeb-DF dataset, with high-quality manipulations, challenges the model’s precision, while the diverse manipulations in the DFDC dataset test its adaptability. This comparison highlights the model’s robustness and potential areas for optimization.

Experimental results demonstrate that the hybrid model achieves high accuracy across both datasets, emphasizing the importance of combining spatial and temporal features for effective deepfake detection. The study provides a foundation for real-time detection systems and future enhancements, including optimizing preprocessing steps and expanding evaluations to additional datasets.

**Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Contents** | **Page No.** |
| **1** | **INTRODUCTION** | **1 - 4** |
|  | **1.1 Description** | **1** |
|  | **1.2 Problem Formulation** | **1** |
|  | **1.3 Motivation** | **2** |
|  | **1.3 Proposed Solution** | **3** |
|  | **1.4 Scope of the project** | **3** |
| **2** | **REVIEW OF LITERATURE** | **5 - 8** |
| **3** | **SYSTEM ANALYSIS** | **9 - 11** |
|  | **3.1 Functional Requirements** | **9** |
|  | **3.2 Non-Functional Requirements** | **10** |
|  | **3.3 Specific Requirements** | **10** |
|  |  |  |
| **4** | **ANALYSIS MODELING** | **12 - 14** |
|  | **4.1 Functional Modeling** | **12** |
|  | **4.2 TimeLine Chart** | **14** |
| **5** | **DESIGN** | **15 - 21** |
|  | **5.1 Architectural Design** | **15** |
|  | **5.2 User Interface Design** | **21** |
| **6** | **IMPLEMENTATION** | **22 - 32** |
|  | **6.1 Module Implemented** | **22** |
|  | **6.2 Output** | **24** |
|  |  |  |
| **7** | **CONCLUSION** | **33** |

Appendix

Literature Cited

Acknowledgements

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Fig. No.** | **Figure Caption** | **Page No.** |
| Fig. 4.1 | Data Flow Diagram | 12 |
| Fig. 5.1 | Architectural Design Diagram | 15 |
| Fig. 5.1.1(a) | Celeb DF Dataset Example | 16 |
| Fig. 5.1.1(b) | DFDC Dataset Example | 17 |
| Fig. 5.1.2(a) | Extracted Frame | 17 |
| Fig. 5.1.2(b) | Face Detection | 18 |
| Fig. 5.1.2(b) | Face Detection | 18 |
| Fig. 5.2 | User Interface | 21 |
| Fig. 6.1.1 | Model Architecture | 22 |
| Fig. 6.2.1(a) | Celeb DF Confusion Matrix | 24 |
| Fig. 6.2.1(b) | DFDC Confusion Matrix | 25 |
| Fig. 6.2.2(a) | Celeb DF Classification Report | 26 |
| Fig. 6.2.2(b) | DFDC Classification Report | 26 |
| Fig. 6.2.3(a) | Celeb DF Training & Validation Losses Curve | 27 |
| Fig. 6.2.3(b) | DFDC Training & Validation Losses Curve | 28 |
| Fig. 6.2.4(a) | Celeb DF Training & Validation Acurracy Curve | 29 |
| Fig. 6.2.4(b) | DFDC Training & Validation Acurracy Curve | 29 |
| Fig. 6.2.5(a) | User Input | 30 |
| Fig. 6.2.5(b) | Prediction Result | 31 |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **Table No.** | **Table Title** | **Page No.** |
| i | Time Line Chart | 14 |

**List of Abbreviations**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Abbreviation** | **Expanded form** |
| i | GRU | Gated Recurrent Units |
| ii | DFDC | Deep Fake Detection Challenge |
| iii | CNN | Convolutional neural network |
| iv | RNN | Recurrent Neural Network |
| v | PSO | Particle Swarm Optimization |
| vi | GANs | Generative Adversarial Networks |

1. **Introduction**
   1. **Description**

Deepfakes refer to manipulated media primarily videos or audio where artificial intelligence and machine learning algorithms are used to create realistic but fabricated content. These alterations often involve replacing faces or voices in videos with those of different individuals. While deepfakes can have creative uses in entertainment, they pose significant threats to privacy, security, and public trust. They can be exploited for malicious purposes, including misinformation, defamation, and identity theft, leading to social, political, and economic consequences.

The aim of this project is to detect deepfake videos using advanced deep neural networks. Specifically, two widely used datasets Celeb-DF and DFDC are employed, each containing both original and manipulated videos. The project begins with preprocessing, which involves extracting individual frames from the videos and cropping faces from these frames to focus on relevant features. These pre-processed images are then fed into an EfficientNet + GRU model, combining the power of EfficientNet's feature extraction capabilities with the temporal learning ability of the GRU model to detect deepfakes effectively.

By leveraging state-of-the-art deep learning models, this project aims to contribute to the growing field of deepfake detection, helping to mitigate the risks posed by these malicious AI-generated media. The results of this research can have significant implications in fields like cybersecurity, digital forensics, and media integrity.

* 1. **Problem Formulation**

The rapid advancement of deepfake technology has raised concerns over the authenticity of digital media, posing significant risks to privacy, security, and trust in the media. The main challenge in this field is the ability to detect deepfake content accurately while minimizing computational resources and training time. This project addresses the need for a deepfake detection system that is both accurate and computationally efficient, capable of identifying

manipulated content in real-time.To solve this problem, the focus is on improving detection accuracy while reducing the computational cost involved in training the model. Previous models, including MesoNet, DFDC model, and 3D CNN, have shown promising results but also face challenges in terms of accuracy and training efficiency. By combining EfficientNet, a highly efficient CNN architecture, with a GRU this project aims to leverage EfficientNet's superior feature extraction capabilities alongside GRU's ability to capture temporal dependencies, making the system more effective at detecting deepfakes while reducing training time.

The goal is to build a robust deepfake detection system that can be deployed in real-time applications. The system must be capable of processing videos with high accuracy while minimizing computational overhead, making it suitable for use in a wide range of environments, from security systems to social media platforms. This project seeks to contribute to the growing field of deepfake detection by offering a more efficient and reliable solution.

* 1. **Motivation**

Deepfake videos pose significant ethical and security challenges, making reliable detection methods critical. Traditional techniques often fail to handle the sophistication of modern deepfakes, necessitating advanced solutions.

Existing Approaches and Drawbacks:

1. Traditional Methods: Rely on handcrafted features, but lack accuracy and generalizability for high-quality deepfakes.
2. Machine Learning: Requires manual feature engineering and struggles with spatial-temporal data.
3. CNNs Alone: Capture spatial features but fail to model temporal dependencies in videos.
4. RNNs Alone: Handle temporal data but lack spatial feature extraction and suffer from training inefficiencies.

Proposed Approach:

This project combines EfficientNet for high-quality spatial feature extraction and GRU for modeling temporal dependencies in video frames. This hybrid deep learning model ensures robust detection, validated across diverse datasets (Celeb-DF and DFDC), addressing the limitations of existing methods effectively.

* 1. **Proposed Solution**

The proposed solution uses a hybrid deep learning approach combining **EfficientNet** and **GRU** to detect deepfake videos effectively. **EfficientNet** is utilized for spatial feature extraction, analyzing intricate facial details within video frames to focus on high-quality features. This ensures the system can identify subtle manipulations in video content. Meanwhile, **GRU** handles temporal analysis, capturing relationships between frames to detect inconsistencies over time. This dual architecture ensures robust and accurate detection of deepfakes.

Unlike traditional approaches that rely on standalone convolutional networks or basic machine learning, which struggle with temporal dependencies and complex data, this solution integrates spatial and temporal analysis for superior performance. The hybrid model addresses the limitations of existing techniques by offering better generalization and adaptability, ensuring high accuracy across diverse datasets like Celeb-DF and DFDC.

Additionally, the project provides a **user-friendly web interface** where users can upload videos to verify their authenticity. This ensures accessibility for both technical and non-technical users. By making deepfake detection intuitive and reliable, the solution helps combat misinformation, detect manipulated media, and address ethical concerns. The project empowers users to ensure content credibility, contributing to safer and more trustworthy digital interactions.

* 1. **Scope of the project**

The project focuses on detecting deepfake videos using a hybrid deep learning model that combines EfficientNet and GRU for spatial and temporal analysis, respectively. The scope primarily targets face-swap manipulations, addressing the growing need for reliable tools in domains such as social media moderation, digital content verification, news authenticity checks, and forensic investigations. The project leverages two prominent datasets, Celeb-DF and DFDC, ensuring applicability across diverse video manipulation techniques.

The system is designed to process videos containing visible and detectable human faces. It may face limitations when dealing with non-facial manipulations, such as audio spoofing or tampering with non-human subjects. Additionally, while the system achieves robust detection for face-based manipulations, its reliance on datasets with clear and high-quality facial data may reduce performance when analyzing low-resolution or heavily distorted videos.From a computational perspective, the project is constrained by hardware specifications, including 16GB RAM and 4GB VRAM, which limit its ability to handle extremely large-scale datasets or perform real-time detection at scale.

Despite these constraints, the system offers a user-friendly web interface, enabling both technical and non-technical users to upload videos and check their authenticity. Future scalability can include additional datasets, support for non-facial manipulations, and real-time processing enhancements, broadening its applicability and effectiveness.

1. **Literature Review**
   1. **Video deepfake detection using Particle Swarm Optimization**

Deepfake detection has gained prominence due to the misuse of AI-generated synthetic media, which poses risks to privacy and societal stability. Recent approaches leverage hybrid models like EfficientNet and GRU to capture spatial and temporal features for improved detection. EfficientNet’s visual feature extraction combined with GRU’s sequential data handling enhances the identification of deepfakes. Optimization techniques, such as Particle Swarm Optimization, further refine model performance by automating hyperparameter tuning. Datasets like Celeb-DFv2 and DFDC, containing diverse real and fake videos, support the evaluation of detection systems. Preprocessing pipelines involve extracting video frames, detecting and cropping faces, and resizing them for input. Using a PSO-optimized EfficientNet-GRU model, this study achieved superior accuracy, recall, and AUC scores across these datasets, demonstrating the effectiveness of integrating robust architectures with advanced optimization.

The EfficientNet-GRU model, enhanced by PSO, has proven effective for deepfake detection, setting a benchmark for accuracy and robustness. These findings highlight the importance of innovative hybrid models and optimization in combating the challenges posed by deepfakes.

* 1. **Deepfake Detection System Using Deep Neural Networks**

This research focuses on detecting deepfake images generated by StyleGAN using hybrid deep learning models. It employs a publicly available dataset comprising real and fake images derived from the FFHQ and StyleGAN datasets. The dataset includes 1,000 real and 1,000 fake high-resolution images, preprocessed through alignment, resizing, and data augmentation.Three hybrid models were explored: EfficientNetB4-LSTM, InceptionV3-LSTM, and InceptionResNetV2-LSTM. These models leverage CNN architectures for spatial feature extraction and LSTM for temporal classification. Training and validation were conducted using Google Colab with GPUs, optimizing performance through Adam and binary cross-entropy loss functions.

The results showed EfficientNetB4-LSTM achieving the highest accuracy of 98%, followed by InceptionResNetV2-LSTM (97%) and InceptionV3-LSTM (96%). EfficientNetB4-LSTM also had the lowest training time, making it the most efficient model. Limitations included smaller dataset size and reduced image resolution due to computational constraints.The study demonstrates the effectiveness of hybrid CNN-LSTM architectures for deepfake detection. EfficientNetB4-LSTM emerged as the best-performing model, balancing high accuracy and lower training time. Future work involves scaling the dataset, using higher-resolution images, and optimizing model performance with enhanced computational resources.

* 1. **An Efficient Deepfake Video Detection Approach with Combination of EfficientNet and Xception Models Using Deep Learning**

The study "Deepfake Detection Using EfficientNet and XceptionNet" investigates the capabilities of advanced convolutional neural networks (CNNs) in detecting deepfake videos, which pose a growing challenge to digital media integrity. The research employs EfficientNet-B4 and XceptionNet models, both renowned for their superior performance in image classification. EfficientNet-B4 is designed to optimize accuracy while maintaining computational efficiency, and XceptionNet utilizes depthwise separable convolutions to improve feature extraction. The datasets used are FaceForensics++ and Celeb-DF (v2), both critical in benchmarking deepfake detection efforts. The preprocessing involved extracting frames and isolating facial regions to concentrate the models on crucial features. Performance evaluation metrics included log loss and Area Under the Curve, where EfficientNet-B4 achieved an impressive AUC of 0.99 and a log loss of 0.02, outperforming XceptionNet, which recorded an AUC of 0.98 and a log loss of 0.04.

The study concludes that EfficientNet-B4 is more accurate and computationally efficient compared to XceptionNet. However, it emphasizes the need for continuous updates to detection algorithms to keep up with the advancing techniques in deepfake creation. This research highlights the importance of leveraging state-of-the-art CNN architectures for robust deepfake detection systems.

* 1. **Detecting Deepfakes With ResNext and LSTM: An Enhanced Feature Extraction and Classification Framework**

The detection of deepfake videos has gained significant attention due to the increasing prevalence of generative models like GANs . Various detection approaches focus on identifying artifacts introduced during the creation of deepfake content. For example, ResNext, a Convolutional Neural Network, is employed to extract frame-level features, identifying warping artifacts and resolution inconsistencies. Temporal inconsistencies are analyzed using Long Short-Term Memory networks, providing robust video classification results. Studies also highlight the absence of physiological signals, such as eye blinking or heart rate, as critical indicators of deepfakes.

The combined ResNext and LSTM model achieves high accuracy in distinguishing real and fake videos. Specifically, the model demonstrated a detection accuracy of over 95% on benchmark datasets like FaceForensics++ and other curated datasets containing 50% real and 50% manipulated videos. This methodology involves splitting videos into frames, detecting faces, and performing sequential analysis to classify one-second video segments with precision. Additional methods, such as capsule networks and conditional adversarial networks, have also shown robust results across diverse datasets, with high resilience to varying video qualities and attack methods.These advancements highlight the potential of deep learning-based methods in addressing the challenges posed by deepfake technologies

* 1. **Fake Video Detection Model Using Hybrid Deep Learning Techniques**

Deepfake detection has become a critical research area to combat the misuse of AI-generated content in fake news, scams, and misinformation. Advanced deep learning techniques play a significant role in addressing these challenges by identifying anomalies in facial features, motion, and audio-visual consistency.Among the models used, ResNext and LSTM-based architectures stand out for their effectiveness. ResNext, a Convolutional Neural Network, extracts detailed features from video frames, focusing on artifacts introduced by manipulations. LSTM a Recurrent Neural Network , processes temporal inconsistencies between consecutive frames, improving the classification of videos as real or fake. The combined ResNext-LSTM approach achieved over 95% accuracy on datasets like FaceForensics++ and Kaggle's Deepfake Detection Challenge data, highlighting its robustness.

Other models, such as Modified AlexNet and Deep InceptionNet, have also been explored. These models leverage efficient architectures for feature extraction and classification, achieving competitive results on publicly available datasets. Techniques like eye-blinking detection and physiological signal analysis further enhance detection capabilities by exploiting deepfake creation limitations.Overall, deep learning-based methods provide reliable and scalable solutions for deepfake detection. Continuous advancements in model architectures and datasets ensure improved accuracy and robustness against evolving deepfake generation techniques.

1. **SYSTEM ANALYSIS**
   1. **Functional Requirement**

The deepfake detection system is designed to identify manipulated videos using deep neural networks. Below are the key functional requirements of the system:

3.1.1 Input Handling

* The system shall accept video files in common formats such as MP4, AVI, and MOV.
* Users shall be able to upload videos through a Jupyter Notebook interface.

3.1.2 Preprocessing

* The system shall extract frames from the uploaded video.
* The system shall perform face detection and crop the faces from the extracted frames.
* A total of 10 cropped face images shall be extracted per video.

3.1.3 Model Processing

* The system shall use EfficientNet for extracting spatial features from the cropped face images.
* The system shall use GRU to analyze the temporal sequence of features from the frames.
* The model shall classify the video as either "manipulated" or "original."

3.1.4 Prediction Output

* The system shall output a binary classification ("manipulated" or "original").
* The classification result shall be displayed clearly in the Jupyter Notebook interface.

3.1.5 Video Format Handling

* The system shall detect manipulations involving face swaps in the video content.
  1. **Non-Function Requirement**

Non-functional requirements define the overall system performance, usability, and constraints that affect the system's operation. The key non-functional requirements for the deepfake detection system are as follows:

3.2.1 Performance

* The system shall process each video within a reasonable time frame.

3.2.2 Usability

* The system shall provide an intuitive Jupyter Notebook interface for interaction.

3.2.3 Reliability

* The system shall maintain a detection accuracy of at least 90% on the test dataset.

3.2.4 Scalability

* The system shall support future enhancements, such as adding new video formats or additional neural network architectures.
  1. **Specific Requirements**

The specific requirements for the deepfake detection system are as follows:

* + 1. Model Specification

Model Specifications:

* EfficientNetGRU Architecture: The model utilizes EfficientNet as the backbone for feature extraction, followed by a GRU layer for temporal analysis of video frames. The final output is passed through a fully connected layer to classify the video as either "real" or "fake".
* Input Format: The system processes video frames, extracting a fixed number of 10 images per video. These images are resized to 224x224 pixels, normalized, and converted into tensors.
* Sequence Length: The system processes videos by extracting sequences of 10 frames, ensuring temporal relationships are captured through the GRU layer.
* Output Format: The model outputs a classification label: either "real" (0) or "manipulated" (1).
  + 1. Hardware & Software Requirments

Hardware and Software Requirements:

* Hardware:
* 16GB RAM
* 4GB VRAM (GPU)
* 500GB SSD
* Software:
* PyTorch for model development and training
* torchvision for pre-trained models and image transformations
* scikit-learn for model evaluation metrics (accuracy, classification report)
* PIL for image handling
  + 1. Preprocessing & Traning
* Preprocessing: Videos are split into frames, and 10 cropped images are extracted from each video. These images are resized to 224x224 pixels, normalized, and transformed into tensors before being fed into the model.
* Training: The model is trained using the Adam optimizer with a learning rate of 1e-4 and Cross-Entropy Loss as the loss function. A batch size of 8 is used during training.
* Validation and Evaluation: The model is evaluated on a validation set, with classification performance assessed through accuracy scores and a detailed classification report.

1. **ANALYSIS MODELING**
   1. **Functional Modeling**

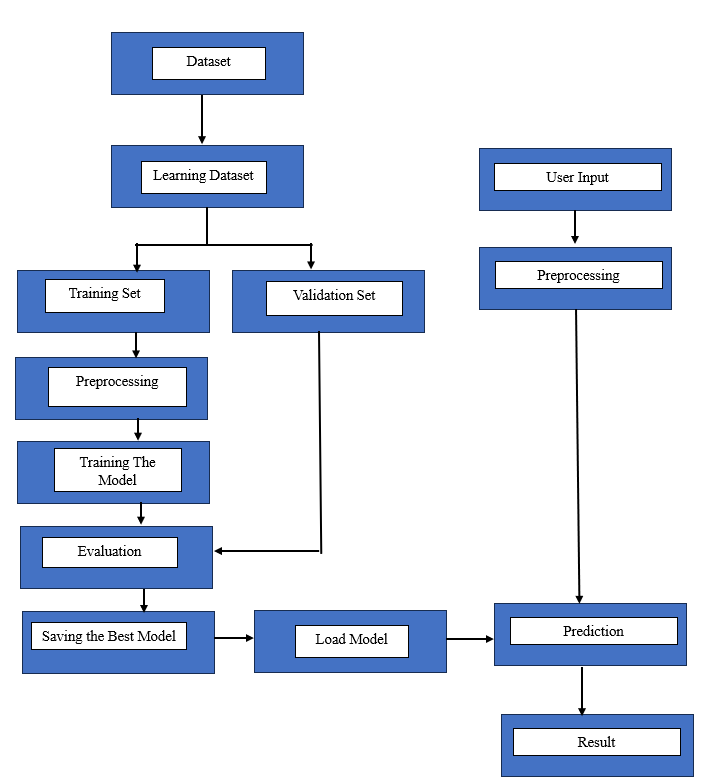


Fig. 4.1 Data Flow Diagram

* + 1. Dataset
  + This represents the initial collection of data used for model development. The dataset is the source from which learning and validation datasets are derived.
    1. Learning Dataset

The dataset is split into two subsets:

* + Training Set: Used to train the model.
  + Validation Set: Used to validate the model’s performance during training to prevent overfitting and ensure generalization.
    1. Training Set Process
* Preprocessing: The training data undergoes preprocessing, which may include data cleaning, normalization, augmentation, etc.
* Training the Model: The preprocessed training data is fed into the model to learn patterns and relationships.
* Evaluation: The model's performance is evaluated using metrics such as accuracy, loss, etc., on the training and validation sets.
* Saving the Best Model: After training, the best-performing model is saved for future use.
  + 1. User Input
* Represents real-time or batch input data from end-users that need to be processed by the trained model.
  + 1. Preprocessing of user input
* The user-provided data is preprocessed in the same manner as the training data to ensure consistency and proper model input format.
  + 1. Model Loading & Prediction
* Load Model: The saved model is loaded for inference.
* Prediction: The pre-processed user input is passed through the model to generate predictions or classifications.
* Result: The prediction results are generated and presented to the user, typically in an easily interpretable format.
  1. **Timeline Chart**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | | | | | | | | | | |
| MONTH | SEPTEMBER | | OCTOBER | | | | | NOVEMBER | | | | DECEMBER | | | | January | | | |
| NO. OF WEEKS | W3 | W4 | | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 | W1 | W2 | W3 | W4 |
| TASK |  | | | | | | | | | | | | | | | | | | |
| 1.SELECTING TOPIC |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2.PREPARATION |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.DATASET |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4.TRAIN MODEL |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5.USER INTERFACE |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6.REPORT MAKING |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. **DESIGN**
   1. **Architectural Design**

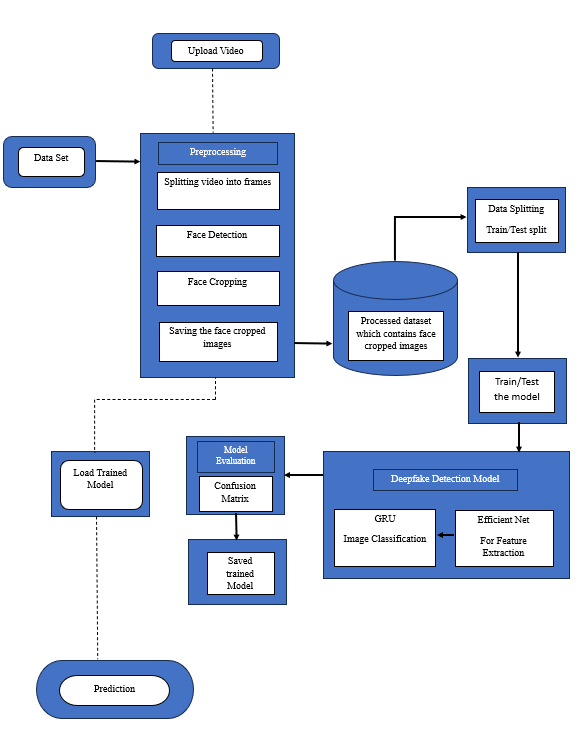


Fig. 5.1 Architectural Design Diagram

The architectural design of the deepfake detection system outlines the flow of data and the interaction between various components. Below are the primary components:

5.1.1. Input: Dataset of Manipulated/Original Videos

The system utilizes two datasets:

* Celeb-DF:
  + Original Videos: 590 high-quality videos featuring celebrities of diverse ages, ethnicities, and genders, sourced from YouTube.
  + DeepFake Videos: 5,639 videos generated using advanced face-swapping techniques, closely resembling real-world deepfake content.
  + 500 original and 500 manipulated videos for training purposes. And 60 videos were taken for testing.



Fig. 5.1.1(a) Celeb DF Dataset Example

* DFDC (Deepfake Detection Challenge):
  + Actors: Features videos from 3,426 paid actors, ensuring a wide range of appearances and behaviors.
  + Videos: Over 100,000 clips produced using various deepfake generation techniques, including GAN-based and non-learned methods.
  + 300 original and 300 manipulated videos were taken for traning. And 60 for testing.



Fig. 5.1.1(b) DFDC Dataset Example

5.1.2. Preprocessing

1. Frame Extraction:



Fig. 5.1.2(a) Extracted Frame

* 1. Videos are processed frame by frame using OpenCV's VideoCapture module.
  2. To ensure computational efficiency, only a fixed number of frames (num\_images, set to 10) are selected for further processing.
  3. The interval between frames is determined dynamically by dividing the total frames by the desired number of images, ensuring even sampling across the video.

1. Face Detection and Cropping:



Fig. 5.1.2(b) Face Detection

* 1. Faces are detected in each selected frame using the Haar cascade classifier for frontal faces (haarcascade\_frontalface\_default.xml) from OpenCV.
  2. Detected faces are cropped based on the bounding box coordinates provided by the face detector. If multiple faces are detected in a frame, only the first detected face is processed to maintain consistency.
  3. To ensure reliable detection, parameters like scaleFactor, minNeighbors, and minSize are carefully tuned to balance accuracy and speed.

1. Image Resizing:

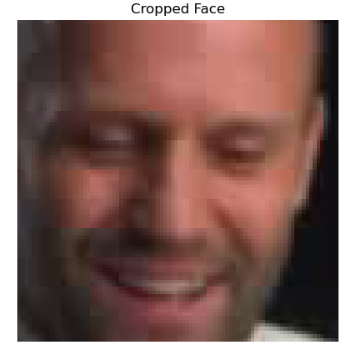


Fig. 5.1.2(c) Cropped Face

* 1. The cropped face images are resized to a uniform dimension (224x224 pixels) using OpenCV's resize function. This size aligns with the input requirements of the EfficientNet model used in the project.

1. Data Organization:
   1. Cropped and resized images are stored in a structured format under separate directories for original and manipulated videos.
   2. Each video's frames are saved in subdirectories named after the corresponding video file.
2. Error Handling:
   1. If no face is detected in a frame, the frame is skipped, ensuring only valid face crops are saved.
   2. The implementation ensures that the Haar cascade is properly loaded, and exceptions are raised if the XML file is missing or corrupted.

5.1.3. Processed Dataset Loading

The preprocessed dataset of cropped face images is loaded into the system, maintaining the class labels (original or manipulated).

5.1.4. Data Splitting

* The data is divided into training and validation sets, with 20% of the data reserved for validation.
* For testing, 50 videos from each class (original and manipulated) are processed into cropped face images.

5.1.5. Deepfake Detection Model

The model consists of two primary components:

a. EfficientNet: Serves as the feature extraction backbone, extracting spatial features from each image frame.

b. GRU: Processes the temporal dependencies of the extracted features across multiple frames to classify the video as "original" or "manipulated."

5.1.6. Model Evaluation

* The model’s performance is evaluated using metrics such as accuracy and a confusion matrix.
* The confusion matrix highlights the true positives, true negatives, false positives, and false negatives.

5.1.7. Saving the Model

The trained model is saved for future use during real-time prediction.

5.1.8. User Input and Prediction

* Users upload a video through the Jupyter Notebook interface.
* The video undergoes preprocessing, including frame extraction and face cropping.
* The pre-processed frames are passed through the saved model for prediction.
* The system outputs whether the video is "original" or "manipulated."
  1. **User Interface Design**

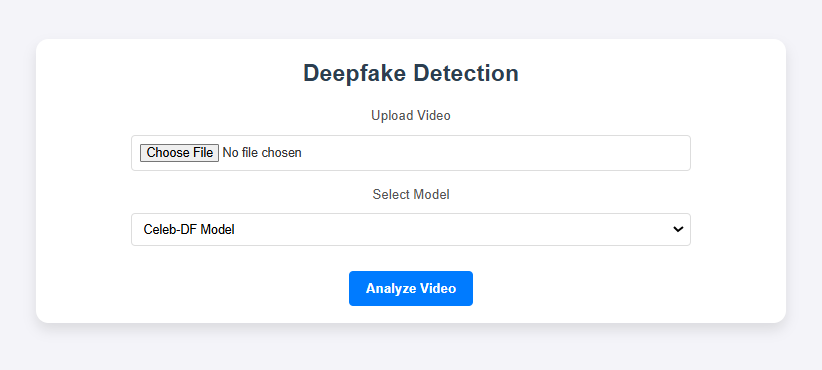


Fig. 5.2 User Interface

The Deepfake Detection system is designed with a user-friendly and intuitive interface, utilizing HTML and CSS for a clean and visually appealing layout. These frontend technologies work together to create an accessible platform for deepfake detection, ensuring a seamless experience for users.At the top of the page, the title "Deepfake Detection" clearly signifies the purpose of the application. Below the title, users can find the "Upload Video" section, which includes a file input field and a "Choose File" button. This button enables users to easily browse and select a video from their device that they wish to analyze.

Once a video is uploaded, users can choose between two deepfake detection models using a dropdown menu. The "Celeb-DF Model" is tailored for detecting deepfakes involving celebrity faces, trained on the Celeb-DF dataset. Alternatively, the "DFDC Model," based on the Deep Fake Detection Challenge (DFDC) dataset, offers a broader detection capability for various types of deepfake videos.At the bottom of the interface, the "Analyze Video" button initiates the analysis process once the video and model are selected. The backend of the system is powered by Flask, which handles the requests and processes the uploaded videos, providing quick results. This combination of technologies ensures efficient and accessible deepfake detection for users.

1. **IMPLEMENTATION**
   1. **Model Implementation**
      1. Model Architecture

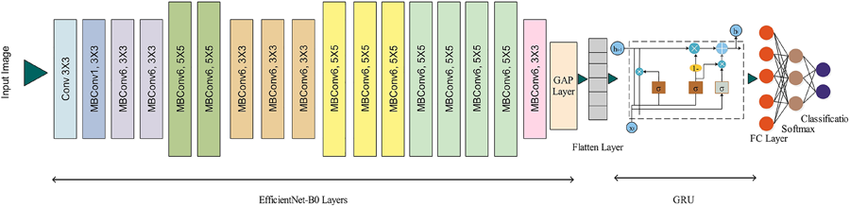


Fig. 6.1.1 Model Architecture

The deepfake detection model is built using a combination of EfficientNet and GRU, designed to handle both the spatial and temporal aspects of video data.

* EfficientNet Backbone: We use the pre-trained EfficientNetB0 model as the feature extractor. EfficientNet was selected due to its high performance and efficiency, providing a good balance between accuracy and computational resources. The feature extraction is performed by removing the final classification layers, and the remaining layers are used to extract relevant features from each frame.
* GRU Layer: The extracted features from each frame are then processed sequentially by a bidirectional GRU layer. The bidirectional nature of the GRU allows the model to capture temporal dependencies in both forward and backward directions, improving its ability to detect deepfake patterns across consecutive frames.
* Fully Connected Layer: After processing the sequence of frame features with the GRU, the hidden states are concatenated and passed through a fully connected layer to produce the final output, which is a binary classification: "original" (0) or "manipulated" (1).
  + 1. Preprocessing Pipeline

The preprocessing pipeline involves extracting frames from the input videos, detecting faces, and resizing them before feeding them into the model.

* Frame Extraction: Frames are extracted from each video, and a fixed number of frames (10) are selected per video sequence. The frame extraction process ensures that the model receives consistent input for each video.
* Face Detection and Cropping: Faces in the frames are detected using a Haar cascade classifier. Once detected, the faces are cropped from the frames and resized to a target dimension of 224x224 pixels to maintain consistency across the dataset.
* Data Loading: A custom VideoFrameDataset class is implemented to handle loading, transformation, and batching of the data. Each frame is transformed using a standard pipeline of resizing, normalization, and conversion to tensors.
  + 1. Model Training

The model was trained using the following parameters:

* Optimizer: The Adam optimizer was used for training, with a learning rate of 1e-4.
* Loss Function: Cross-entropy loss was used for the binary classification task.
* Batch Size: A batch size of 8 was used for training, which balances computational efficiency and model performance.
* Training Time: Each epoch took approximately 10 minutes, and the model was trained for 5 epochs, resulting in a total training time of 50 minutes.

The model was trained on a machine with 16GB of RAM, 4GB of VRAM, and a 500GB SSD.

* + 1. Evaluation

The model was evaluated on a validation set, and metrics such as accuracy, precision, recall, and F1-score were calculated. The evaluation was done using a classification report to assess the model's performance in detecting both "original" and "manipulated" videos.

* 1. **Output**

The implemented deepfake detection model outputs a binary classification for each video, indicating whether it is **real** (label: 0) or **manipulated** (label: 1). The prediction is based on a sequence of 10 frames extracted from the video, which are processed through the EfficientNet-GRU architecture. The output is generated as a probability distribution over the two classes, with the final classification determined by the class with the higher probability.

* + 1. Confusion Matrix

Celeb-DF Dataset:

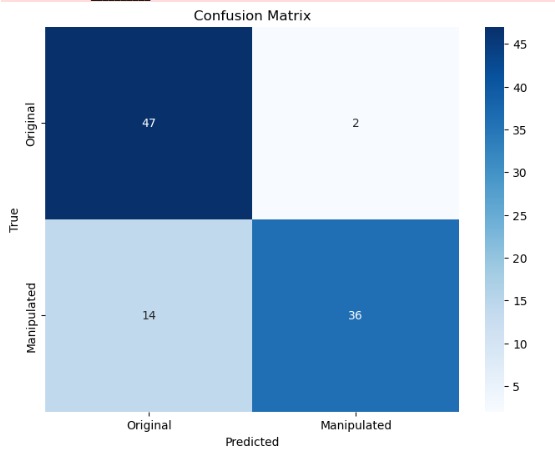


Fig. 6.2.1(a) Celeb DF Confusion Matrix

DFDC Dataset:

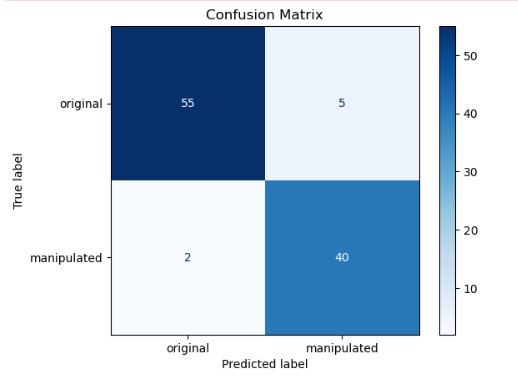


Fig. 6.2.1(b) DFDC Confusion Matrix

On the Celeb-DF dataset, the model exhibited a high degree of accuracy in identifying both genuine and manipulated videos, with low rates of false positives and false negatives. Similarly, on the DFDC dataset, the model demonstrated strong performance, although a slightly higher number of false negatives were observed for the manipulated class compared to Celeb-DF, potentially indicating a slightly lower sensitivity to certain types of manipulations within the DFDC dataset.

* + 1. Classification Report

The classification report summarizes the model’s performance in terms of precision, recall, and F1-score for both classes. The results are presented separately for the **Celeb-DF** and **DFDC** datasets:

Celeb-DF Dataset:

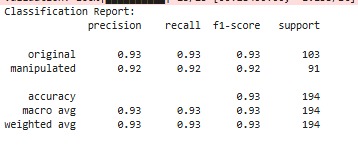


Fig. 6.2.2(a) Celeb DF Classification Report

DFDC Dataset:

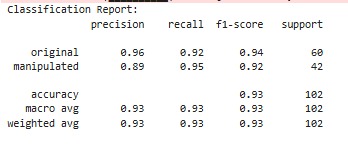


Fig. 6.2.2(b) DFDC Classification Report

* Overall Accuracy: Both datasets show comparable accuracy (around 0.93), indicating good overall performance.
* Precision & Recall:
  + Celeb-DF generally shows slightly better precision for the manipulated class.
  + DFDC generally shows slightly better recall for the manipulated class.

F1-score: Both datasets show comparable F1-scores for both classes, indicating a balanced trade-off between precision and recall.

* + 1. Training & Validation Loss

The training and validation loss for both datasets were recorded over 5 epochs. The loss curves demonstrate the model’s learning progress, with validation loss stabilizing as training progresses.

Celeb-DF Dataset Loss:



Fig. 6.2.3(a) Celeb DF Training & Validation Losses Curve

* Training and validation loss curves show a good balance with the validation loss stabilizing, suggesting minimal overfitting.
* DFDC Dataset Loss:

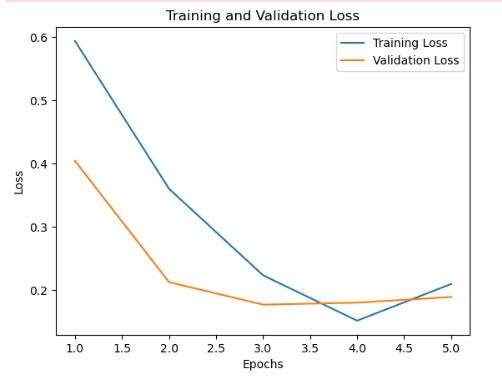


Fig. 6.2.3(b) DFDC Training & Validation Losses Curve

* The validation loss curve shows a slight increase towards the end, indicating a higher risk of overfitting.
  + 1. Training & Validation Accuracy

The model achieved consistent improvements in accuracy during training, with high validation accuracy indicating good generalization.

Celeb-DF Dataset Accuracy:

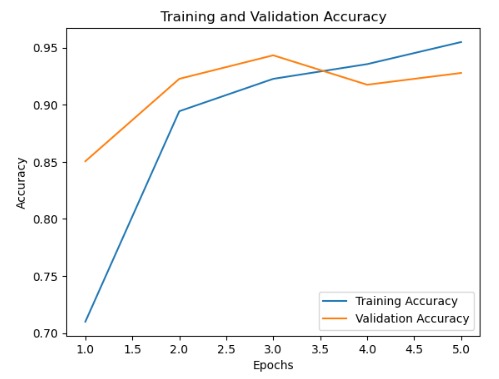


Fig. 6.2.4(a) Celeb DF Training & Validation Acurracy Curve

* Both training and validation accuracy increase steadily and remain close together, suggesting good generalization with minimal overfitting..

DFDC Dataset Accuracy:

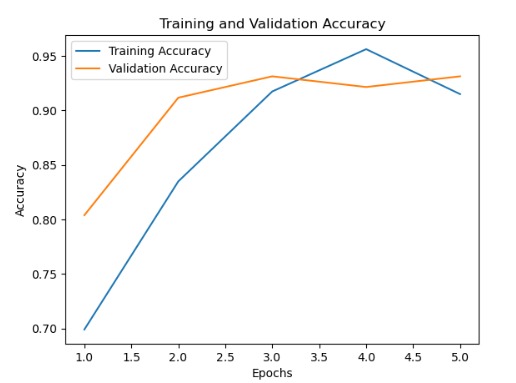


Fig. 6.2.4(b) DFDC Training & Validation Acurracy Curve

* Training accuracy continues to increase, while validation accuracy plateaus or slightly decreases, indicating a higher risk of overfitting.
  + 1. Webpage Results
       1. Input Page

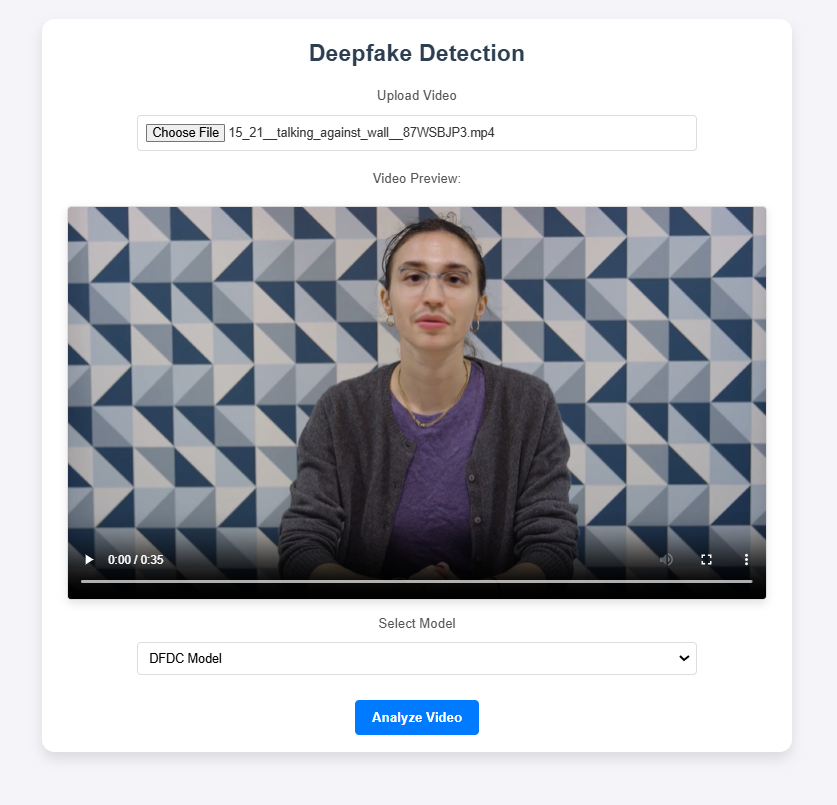


Fig. 6.2.5(a) User Input

* Upload Video Section: The user is prompted to choose a video file to upload.
* Video Preview: A preview of the video is shown, featuring a person speaking against a geometric-patterned wall.
* Model Selection: The user can select the detection model, with the "DFDC Model" pre-selected.
* Analyze Button: A blue button labeled "Analyze Video" allows the user to begin the deepfake analysis process.
  + - 1. Prediction Page

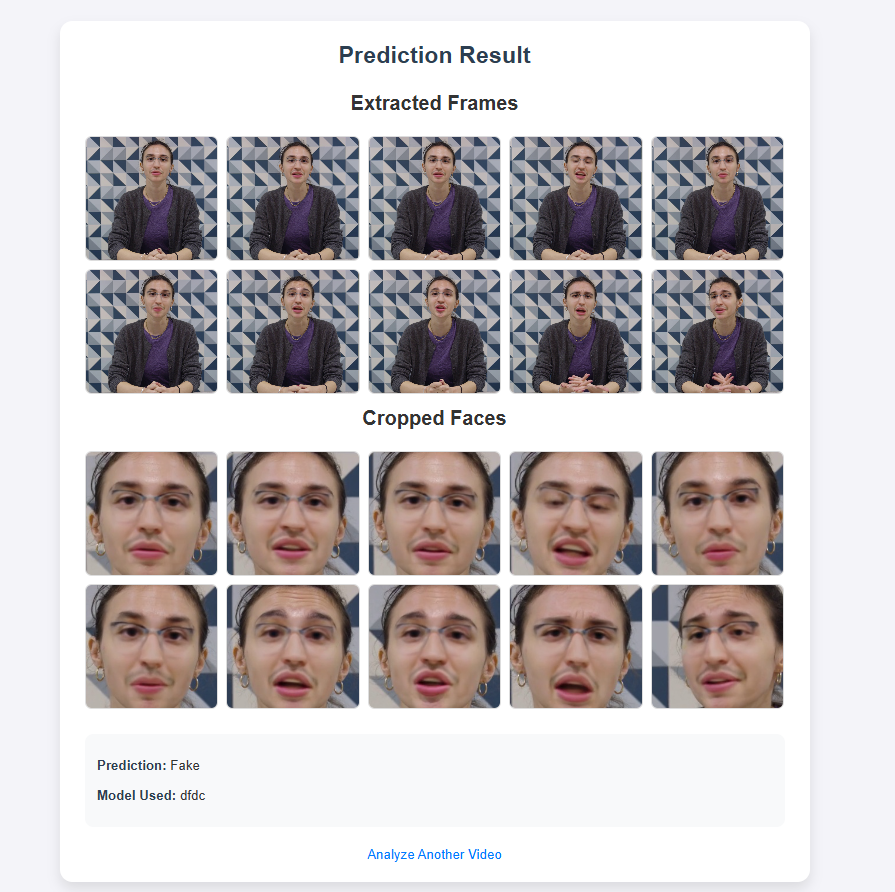


Fig. 6.2.5((b) Prediction Result

Image Description: Prediction Result  
The image showcases the results of a deepfake detection system presented on a webpage. The interface consists of two sections:

1. Extracted Frames: A grid displaying still frames extracted from the input video. The frames show a person seated against a geometric-patterned background, maintaining a consistent posture and appearance across the frames.
2. Cropped Faces: Another grid displaying close-up images of the detected face in each frame. These cropped images highlight subtle distortions, particularly around facial features, such as the mouth and eyes, which are indicative of potential deepfake artifacts.

Below the grids, a prediction outcome is presented:

* Prediction: Fake
* Model Used: dfdc

A button labeled "Analyze Another Video" is available for users to input a new video for analysis.

1. **CONCLUSION**

This project successfully developed a deepfake detection system using deep neural networks, integrating EfficientNet and GRU for spatial and temporal feature extraction. Trained and evaluated on the Celeb-DF and DFDC datasets, the model achieved high accuracy, reaching 0.9549 on Celeb-DF and 0.9314 on DFDC, with robust performance in distinguishing real and fake videos.The preprocessing pipeline efficiently handled frame extraction, face detection, and resizing, ensuring high-quality input for training. Evaluation metrics, including the confusion matrix and classification report, demonstrated low rates of false positives and false negatives, confirming the model's reliability.

**REFERENCES**

1. Cunha, L., Zhang, L., Sowan, B. *et al.* Video deepfake detection using Particle Swarm Optimization improved deep neural networks. *Neural Comput & Applic* **36**, 8417–8453 (2024).
2. M. L. Saini, A. Patnaik, Mahadev, D. C. Sati and R. Kumar, "Deepfake Detection System Using Deep Neural Networks," 2024 2nd International Conference on Computer, Communication and Control (IC4), Indore, India, 2024, pp. 1-5, doi: 10.1109/IC457434.2024.10486659. keywords: {Deep learning;Deepfakes;Visualization;Video games;Streaming media;Vectors;Convolutional neural networks;Deepfake;Fake contents;Deep Neural Networks;Generative Adversarial Networks (GANs);Anvil Platform},
3. S. AtaŞ, İ. İlhan and M. KarakÖse, "An Efficient Deepfake Video Detection Approach with Combination of EfficientNet and Xception Models Using Deep Learning," 2022 26th International Conference on Information Technology (IT), Zabljak, Montenegro, 2022, pp. 1-4, doi: 10.1109/IT54280.2022.9743542. keywords: {Deep learning;Anxiety disorders;Companies;Data models;Forgery;Information technology;Artificial intelligence},
4. R. Taviti, S. Taviti, P. A. Reddy, N. R. Sankar, T. Veneela and P. B. Goud, "Detecting Deepfakes With ResNext and LSTM: An Enhanced Feature Extraction and Classification Framework," 2023 International Conference on Signal Processing, Computation, Electronics, Power and Telecommunication (IConSCEPT), Karaikal, India, 2023, pp. 1-5, doi: 10.1109/IConSCEPT57958.2023.10170580. keywords: {Deepfakes;Visualization;Machine learning algorithms;Finance;Signal processing;Media;Feature extraction;Deepfakes;ResNext;LSTM;CNN},
5. O. S. A. Aboosh, A. N. Hassan and D. K. Sheet, "Fake Video Detection Model Using Hybrid Deep Learning Techniques," 2023 6th International Conference on Information and Communications Technology (ICOIACT), Yogyakarta, Indonesia, 2023, pp. 499-504, doi: 10.1109/ICOIACT59844.2023.10455952. keywords: {Deep learning;Graphics;Deepfakes;Image forensics;Recurrent neural networks;Logic gates;Feature extraction;DeepFake;FaceSwap;Face2Face;Gated Recurrent Unit;InceptionV3;Recurrent Neural Network},

**Appendix**

**A. Datasets Used**

* **Celeb-DF**:
  + **Original Videos**: 590 videos featuring diverse subjects.
  + **DeepFake Videos**: 5,639 videos generated using advanced face-swapping techniques.
  + Dataset used: 500 original and 500 manipulated videos for training, 60 for testing.
* **DFDC (Deepfake Detection Challenge)**:
  + Videos sourced from 3,426 actors, including over 100,000 clips.
  + Dataset used: 300 original and 300 manipulated videos for training, 60 for testing.

**B. Tools and Frameworks**

Programming Language: Python

* Libraries and Frameworks:
  + PyTorch (Model Development and Training)
  + torchvision (Pre-trained Models and Image Transformations)
  + scikit-learn (Evaluation Metrics)
  + PIL (Image Handling)
* Development Environment: Jupyter Notebook
* Visualization Tools: Matplotlib, Seaborn

**C. Hardware Configuration**

* RAM: 16GB
* VRAM: 4GB
* Storage: 500GB SSD

**D. Additional Observations**

* The Celeb-DF dataset posed challenges due to its high-quality manipulations, requiring the model to focus on finer details.
* DFDC offered diverse manipulation types, providing a robust test of model adaptability.

**ACKNOWLEDGEMENT**

I would like to express my heartfelt gratitude to everyone who supported me throughout this project.

First and foremost, I extend my deepest thanks to my project guide, Dr. Chetashshri Bhadane, for their invaluable guidance, insightful feedback, and unwavering encouragement, which were instrumental in completing this work successfully.

I am also profoundly grateful to Dr. Meera Narvekar, the Head of the Department of Computer Engineering, for providing the necessary resources and creating a conducive environment that enabled me to carry out this project effectively.

My sincere appreciation goes to my classmates and friends for their constructive suggestions, collaboration, and support at various stages of this project. Their input has greatly enhanced the quality of my work.

Lastly, I want to express my deep gratitude to my family and close friends for their constant support, understanding, and motivation throughout this journey. Their belief in me has been a source of strength and inspiration.

Thank you all for making this endeavor a fulfilling and rewarding experience.