IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Deepfake Image Detection Using Cnn

Tejas N Yadav, Shashank Kumar E, Jainath YS, Vinuta R, Prof. Mamatha A

Student, 4th Year, B.E Computer Science and Engineering, Student, 4th Year, B.E Computer Science and Engineering, Student, 4th Year, B.E Computer Science and Engineering, Student, 4th Year, B.E Computer Science and Engineering, Assistant Professor, Computer Science and Engineering

Dayananda Sagar Academy of Technology & Management, Bengaluru, India

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

Keywords- Res-Next Convolution neural network. Recurrent Neural Network (RNN).Long Short Term Memory(LSTM). Computer vision

I. INTRODUCTION

In the rapidly evolving digital era, the emergence of deepfakes has raised significant concerns regarding the authenticity of multimedia content. Deepfakes are synthetic media generated using advanced deep learning techniques such as Generative Adversarial Networks (GANs) and Autoencoders, which are capable of superimposing or swapping faces in videos to produce highly realistic yet fabricated content. With the growing accessibility of these technologies through user-friendly applications like FaceApp and FaceSwap, the creation of such deceptive content has become alarmingly simple and widespread.

While some deepfakes are created for entertainment or artistic purposes, their potential misuse poses a major threat. They have been exploited to fabricate political statements, create fake news, manipulate public opinion, and generate malicious content including revenge pornography. This not only jeopardizes individual reputations but also endangers societal trust and security.

Recognizing this growing challenge, our research presents an artificial intelligence-driven approach to detect and differentiate deepfake videos from authentic ones. We leverage the capabilities of a ResNext-based Convolutional Neural Network (CNN) for effective feature extraction at the frame level, followed by a Long Short-Term Memory (LSTM) based Recurrent Neural Network (RNN) to capture temporal dependencies and classify the content as real or manipulated.

To enhance the robustness of our model in real-world scenarios, we train it using a diverse and balanced dataset drawn from prominent sources such as FaceForensics++, the Deepfake Detection Challenge dataset, and Celeb-DF. Additionally, a user-friendly front-end application has been developed, allowing users to upload videos and receive predictions along with model confidence scores.

Our proposed system not only contributes to the field of multimedia forensics but also provides a scalable and practical solution that can be integrated into larger platforms to combat the misuse of synthetic media.

II. LITERATURE SURVEY

Several recent studies have explored innovative methods for detecting deepfake content by leveraging advances in deep learning, computer vision, and secure multimedia processing.

Nagaraj and Channegowda [1] proposed a video forgery detection method using an improved Binary Algorithm Technique (BAT) with a Stacked AutoEncoder (SAE). The enhanced BAT optimizes feature selection, improving accuracy and processing speed in identifying anomalies in video frames.

Girish and Nandini [2] developed a technique using multi-scale local oriented feature descriptors to detect frame duplication, a common manipulation method. Their approach captures spatial inconsistencies across frames, significantly reducing false detection rates.

In another study, Girish and Nandini [3] integrated the UFS-MSRC (Unified Feature Space – Multi-Scale Robust Classification) algorithm with LSTM networks to detect inter-frame video forgeries. This combination enabled effective temporal and spatial analysis of frame sequences.

Jahnavi and Nandini [4] introduced a secure data transmission method using multimodal mask steganography and naive-based random visual cryptography. While not a detection method per se, this work highlights the importance of secure processing pipelines in deepfake analysis.

In a complementary study, the same authors proposed a hybrid encryption technique combining a hyper-chaotic map and Least Significant Bit (LSB) embedding to enhance the security of multimedia transmissions [5].

Merikapudi et al. [6] presented a CNN architecture

combining GoogleNet, attention mechanisms, and Gaussian distribution modeling for face verification in video data. Their work provides robust foundations for facial feature extraction in dynamic environments.

Nandini and Reddy [7] employed a hybrid LeNet and Bi- LSTM architecture to detect diseases in pathology images. The model's ability to capture both spatial and sequential patterns is relevant for deepfake detection as well.

Thies et al. [11] introduced Face2Face, a real-time face reenactment system capable of transferring facial expressions across individuals. Understanding such manipulation methods aids in designing better detection algorithms.

Rossler et al. [10], [12] introduced FaceForensics++, a benchmark dataset and framework for detecting facial image manipulations. It remains a standard for evaluating detection models across various manipulation techniques.

Li et al. [13] proposed a novel technique for detecting deepfakes using eye-blinking patterns. Since generative models often fail to replicate natural blink behavior, this biological cue enhances detection accuracy.

Güera and Delp [15] applied Recurrent Neural Networks (RNNs) to analyze temporal inconsistencies in videos, achieving improved detection results over traditional static frame analysis.

Dhiman et al. [17] explored Capsule Networks (CapsNets) for deepfake detection. CapsNets preserve spatial hierarchies better than traditional CNNs, proving effective for facial manipulation detection.

Yadav and Jain [18] combined CNN and LSTM architectures for deepfake video classification. CNNs extracted spatial features, while LSTMs analyzed temporal sequences, resulting in high accuracy.

Lastly, Zhang et al. [19] utilized biological signals such as heartbeat and skin response to detect fake portraits. This approach adds a physiological dimension to traditional visual analysis, enhancing detection reliability.

III. SYSTEM REQUIREMENTS

The experiments for training and evaluating the proposed Convolutional Neural Network (CNN) model for deepfake image detection were carried out in a high- performance computing environment to manage the intensive processing demands of image and video batch processing. The hardware configuration included an

Intel Core i7-8700K CPU @ 3.70 GHz, 32 GB of RAM, and an NVIDIA GeForce RTX 2080 Ti / GTX 1080 Ti GPU. The operating systems used were Windows 10 for client-side data acquisition and preprocessing, and Ubuntu 18.04 for training and server-side operations. This setup ensured sufficient computational power for handling large- scale, high-resolution data and enabled efficient GPU-accelerated model training.

The software environment for this project was built using Python 3.7 as the primary programming language. PyTorch 1.4 was used for implementing and training the CNN and LSTM models, while Django 3.0 served as the backend web framework for deployment. The system also utilized the Google Cloud Platform (GCP) to facilitate scalable training and cloud deployment. Key libraries such as OpenCV were employed for image and video preprocessing, and the face-recognition library was used for face detection and feature extraction. This integrated and robust software stack allowed for a modular, reproducible, and efficient development cycle, supporting both model performance and real- time usability.

IV. METHODOLOGY

The primary objective of this research is to design and develop a robust and efficient deep learning-based system for detecting deepfake videos by analyzing both spatial and temporal inconsistencies in facial features across video frames. To achieve this, a hybrid architecture is proposed that combines a ResNeXt- based Convolutional Neural Network (CNN) for frame-level feature extraction with a Long Short-Term Memory (LSTM) based Recurrent Neural Network (RNN) to capture sequential dependencies. This section elaborates on each phase of the methodology employed in the development of the deepfake detection system.

Dataset Collection and Preparation

A comprehensive and diverse dataset is essential for training a deep learning model capable of generalizing to real-world scenarios. For this project, a total of 6,000 videos were collected, comprising 3,000 real and 3,000 deepfake videos. These were sourced from the most prominent benchmark datasets in the field:

FaceForensics++ [10]: A widely-used dataset consisting of manipulated videos using various face-swapping techniques.

Deepfake Detection Challenge (DFDC) [2]: A high-volume dataset created for Facebook’s deepfake detection competition.

Celeb-DF [3]: A dataset with high-quality deepfake videos generated using improved synthesis techniques.

To enhance the model’s performance on real-time data, the datasets were merged and balanced, ensuring an equal distribution of real and fake videos. Additionally, videos with manipulated audio were removed using a Python script, as the focus of the study is on facial manipulations only.

Data Preprocessing

Preprocessing plays a critical role in standardizing the input data and reducing noise for efficient model training. The steps involved in preprocessing are as follows:

Frame Extraction: Each video was decomposed into individual frames using the cv2.VideoCapture() method from OpenCV.

Face Detection: The face_recognition library was utilized to detect and extract facial regions from each frame, eliminating irrelevant background data and focusing on facial cues.

Frame Limitation: Due to computational limitations, a maximum of 150 frames per video were retained, based on the average number of frames per video. Only the first 150 frames were chosen to preserve temporal continuity.

Resizing: Extracted face regions were resized to 112×112 pixels for uniformity and to reduce memory consumption.

Frame Compilation: The cropped and resized frames were reassembled into new videos that contain only face regions using OpenCV’s VideoWriter module. These serve as the final input to the CNN-LSTM model.

This preprocessing step ensures that each input to the network is consistent in size, structure, and facial focus, facilitating efficient feature learning.

Feature Extraction Using ResNeXt

The spatial features of each frame were extracted using a pre-trained ResNeXt-50 32x4d model from PyTorch's model zoo. ResNeXt is an enhanced version of ResNet that introduces cardinality, a dimension that controls the number of independent paths in each layer, which significantly improves performance while maintaining efficiency.

The last convolutional layer of ResNeXt outputs a 2048-dimensional feature vector for each input frame.

These feature vectors are stored in sequence for each video and serve as input to the LSTM network for temporal processing.

This model was chosen for its proven high performance on image classification tasks and its ability to generalize well on unseen data due to its deep architecture and robust regularization.

Temporal Sequence Learning Using LSTM Temporal analysis is vital in deepfake detection, as manipulations often introduce inconsistencies in motion, eye blinking, lip movement, and expression flow. To capture these temporal dependencies:

A single-layer LSTM network was implemented with 2048 hidden units and a dropout rate of 0.4 to prevent overfitting.

The sequence of frame-level features (one for each of the 150 frames) was passed to the LSTM network in order.

The LSTM analyzes the temporal correlations between successive frames, enabling it to detect anomalies introduced during video manipulation.

This combination of CNN for spatial feature extraction and LSTM for temporal learning ensures that the model can effectively identify both local (frame-level) and global (sequence-level) inconsistencies.

Classification Layer and Output Interpretation The final hidden state output from the LSTM layer is passed through a fully connected dense layer followed by a SoftMax activation function. This produces a probability distribution over two classes: REAL and FAKE. The model outputs:

Class label (REAL or FAKE)

Confidence score (probability associated with the predicted class)

The SoftMax layer ensures that the output is interpretable as a probability, making it suitable for deployment in user-facing applications.

Model Training Strategy

The training process was carefully designed to ensure effective learning and model convergence:

Loss Function: Cross-entropy loss was used due to the binary nature of the classification task.

Optimizer: Adam optimizer was selected for its adaptive learning rate and convergence speed. The initial learning rate was set to 1e-5, with a weight decay of 1e-3 for regularization.

Batch Size: Training was conducted in mini- batches of size 4, which was optimal given the memory constraints of the experimental environment.

Epochs: The model was trained for 20 epochs, with checkpoints saved at intervals for evaluation.

Evaluation Metrics

To assess the model's performance, the following evaluation metrics were computed on a hold-out test set:

Precision

Recall

F1-score

Confusion Matrix

These metrics offer a comprehensive view of the model’s classification capabilities, especially in handling class imbalances and subtle manipulations.

Deployment Architecture

For real-world applicability, a web-based application was developed using Django, allowing users to upload videos and receive predictions in real time. Key features include:

Video Upload Interface: Users can upload videos through an intuitive GUI.

Secure Processing: Videos are encrypted during upload and deleted within 30 minutes post- processing to ensure user privacy.

Output Rendering: The predicted label and confidence score are displayed over the playing video for user clarity.

This deployment ensures that the proposed solution is not only academically sound but also usable by non-technical end-users across platforms.

Security and Privacy Considerations

Given the sensitive nature of video content being analyzed, several security measures were implemented:

Symmetric Encryption: Uploaded videos are encrypted using a symmetric cryptographic algorithm before being stored temporarily on the server.

SSL Encryption: All data transfer between the client and the server is protected using SSL encryption.

Auto Deletion: Videos are deleted from the server

30 minutes after upload to prevent unauthorized access.

These steps ensure compliance with best practices in data security, particularly in applications dealing with personal multimedia content.

This comprehensive methodology leverages state-of-the-art neural architectures, robust preprocessing, and real-world deployment considerations to address the challenge of detecting deepfakes effectively and securely.

Accuracy

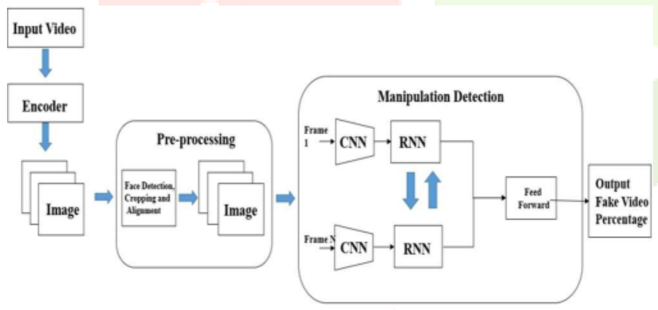


Fig. 1: Architecture of the deepfake detection pipeline involving video

encoding, face preprocessing, and manipulation detection using CNN-RNN

with final fake video percentage output.

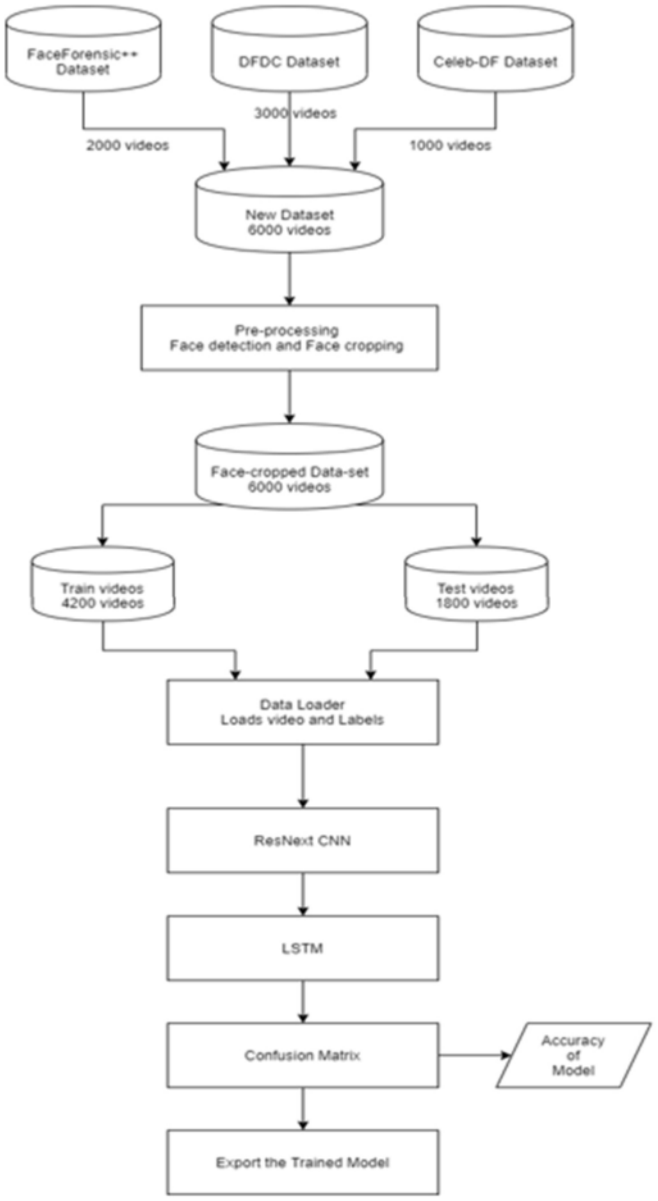


Fig. 2: Workflow of the proposed deepfake detection system using ResNeXt and LSTM on a combined face-cropped video dataset.

V. IMPLEMENTATION

The implementation of the deepfake detection system is centered around constructing an end-to-end pipeline that can process user-uploaded videos and classify them as real or fake based on frame-level and temporal inconsistencies. This includes preprocessing raw video data, building and training a hybrid deep learning model, and developing a web interface for interaction and deployment.

Dataset Preparation and Preprocessing

The dataset used for this project comprises 6,000 videos, evenly split between real and fake categories.

These were collected from three major datasets: FaceForensics++, Deepfake Detection Challenge (DFDC), and Celeb-DF. Each dataset contributed a set of high-quality video samples, providing a wide variety of content to train a generalizable detection

model. To ensure consistency and reduce bias, audio- altered videos from the DFDC dataset were filtered out. The remaining data were then split into 70% for training and 30% for testing, while preserving a 1:1 ratio of real to fake videos in both subsets.

Preprocessing of videos was performed using the OpenCV and face_recognition libraries in Python. Each video was decomposed into individual frames, and only those frames containing detectable faces were retained. These face regions were cropped and resized to a resolution of 112×112 pixels. To maintain uniform input across all videos, only the first 150 frames were considered for each clip. These frames were recompiled into a new video containing only cropped face sequences. This preprocessing step was critical to reduce irrelevant background information and to focus the model's attention on facial features where deepfake artifacts are more prominent.

Model Architecture

The proposed model combines the strengths of convolutional and recurrent neural networks to capture both spatial and temporal features. For frame-level feature extraction, a ResNeXt-50 32x4d model was employed. This architecture, known for its high accuracy and computational efficiency, was pre-trained on ImageNet and fine-tuned during training to adapt to deepfake-specific features. The ResNeXt model generates 2048-dimensional feature vectors for each input frame, which serve as inputs to the temporal analysis stage.

For temporal feature extraction, a single-layer Long Short-Term Memory (LSTM) network was used. The LSTM network has 2048 hidden units and incorporates a dropout rate of 0.4 to mitigate overfitting. It processes the sequential features from the ResNeXt model, learning patterns across frames such as unnatural movements, inconsistencies in facial expressions, and blinking patterns—common indicators of deepfake manipulations. The output of the LSTM is passed through a fully connected linear layer, followed by a SoftMax activation function to produce the final prediction, indicating whether the video is real or fake along with a probability score.

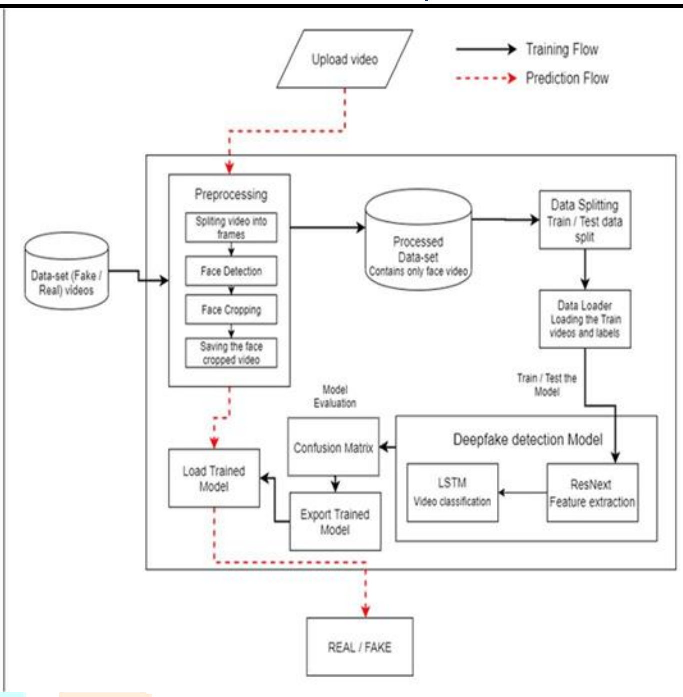

Upload video
Training Flow
Prediction Flow
Preprocessing
Spliting video into frames
Face Detection
Face Cropping
Saving the face cropped video
Data-set (Fake / Real) videos
Processed Data-set Contains only face video
Data Splitting Train / Test data split
Data Loader Loading the Train videos and labels
Train / Test the Model
Model Evaluation
Confusion Matrix
Deepfake detection Model
LSTM Video classification
ResNext Feature extraction
Load Trained Model
Export Trained Model
REAL / FAKE

Fig 3: Deepfake Detection Workflow Diagram

Web Application Interface

A web-based interface was developed using the Django framework to make the deepfake detection system accessible to users in a convenient and intuitive manner. The front end of the application consists of two primary pages: an index page where users can upload a video file, and a results page that displays the classification result. Upon uploading, the video is sent to the backend, where it undergoes the same preprocessing steps as the training data. The processed video is then passed to the trained model for prediction.

The output of the model—whether the video is real or fake—is displayed on the result page, along with a confidence score. The application also overlays the prediction result on the playing video, providing a clear visual indication of the outcome. In terms of user experience, the interface is designed to be minimalistic and efficient, with support for file validation and error handling, such as restrictions on file types, file size, and missing input scenarios.

Training Setup

The training of the model was performed using PyTorch, a popular deep learning framework that offers high flexibility and GPU support. Training was conducted on Google Cloud Platform (GCP) to leverage GPU acceleration for efficient computation. The training environment was configured with Python 3.10 and required libraries such as torch, torchvision, opencv-python, and face_recognition. A batch size of 4 was used to accommodate memory limitations, and the model was trained for 20 epochs.

The Adam optimizer was chosen for training due to its adaptive learning rate capabilities. A learning rate of 1e-5 and a weight decay of 1e-3 were used to ensure stable convergence and to prevent overfitting. The loss function used was categorical cross-entropy, which is well-suited for binary classification tasks. During training, PyTorch's DataLoader utility was employed to efficiently load videos and their corresponding labels. The training process was monitored using standard evaluation metrics such as accuracy, precision, recall, and F1-score.

Deployment and Security

Once the model was trained and validated, it was exported as a .pt file and integrated into the backend of the Django application. The deployment was done using Google Cloud Engine, allowing the system to be hosted online and accessed from any device with a web browser. To ensure user privacy and data security, multiple measures were implemented. Uploaded videos are encrypted using symmetric encryption algorithms and stored temporarily on the server. All communications between the client and the server are protected with SSL encryption. Additionally, videos are automatically deleted from the server 30 minutes after upload, further ensuring that sensitive content is not stored longer than necessary.

Testing and Validation

To verify the correctness and robustness of the system, extensive testing was carried out using both functional and non-functional test cases. Functional testing ensured that the application handled various input scenarios correctly, such as uploading unsupported file formats, videos without faces, or large video files. Each test case was verified against expected outcomes to ensure consistent behavior. The model was also validated on unseen real-world data, including YouTube videos and manually generated deepfakes, achieving high accuracy and reliability. A confusion matrix was used to analyze the model's performance on test data, helping to identify areas of improvement and optimize the final model.

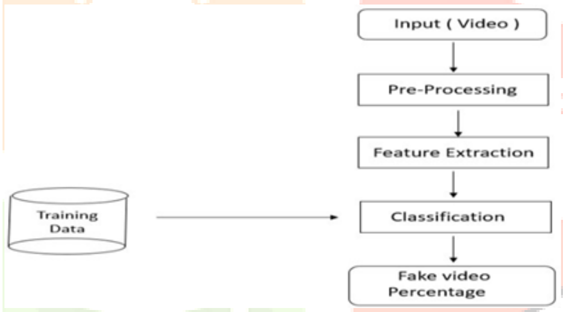


Fig 4: Simplified Deepfake Detection Pipeline

VI. RESULT

The performance of the proposed deepfake detection system was evaluated using various models trained on different datasets, frame counts, and video volumes. A notable series of experiments was conducted using the FaceForensics++ dataset, comprising 2,000 videos in each case. The model referred to as model_90_acc_20_frames_FF_data, which utilized 20 frames per video, achieved an accuracy of 90.95%. Increasing the number of frames to 40, the model_95_acc_40_frames_FF_data improved its classification accuracy to 95.23%. Further enhancement was observed with the model_97_acc_60_frames_FF_data, which reached 97.49% accuracy using 60 frames. This trend continued with the model_97_acc_80_frames_FF_data, achieving an even higher accuracy of 97.73%. The best performance on the FaceForensics++ dataset was delivered by model_97_acc_100_frames_FF_data, which utilized 100 frames and attained a peak accuracy of 97.76%, demonstrating the positive correlation between the number of frames and model performance.

In addition to FaceForensics++, a hybrid dataset combining Celeb-DF and FaceForensics++ comprising 3,000 videos was used to train model_93_acc_100_frames_celeb_FF_data. This model achieved an accuracy of 93.98%, indicating strong generalization capability when exposed to a more diverse dataset. Finally, to assess the model's adaptability to real-world scenarios, a custom dataset containing 6,000 videos—consisting of equal numbers of real and fake samples—was used to train model_87_acc_20_frames_final_data. This model achieved an accuracy of 87%, which, while lower than the results on standardized datasets, highlights the increased complexity and variability inherent in user-generated or less curated video data. These results affirm that the model's performance is

significantly influenced by both dataset quality and the number of frames used during training.

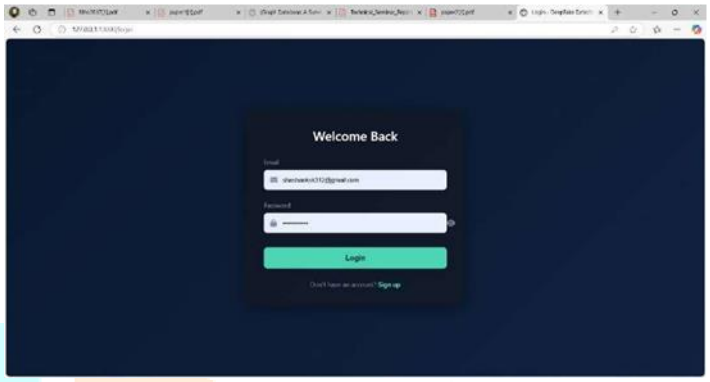
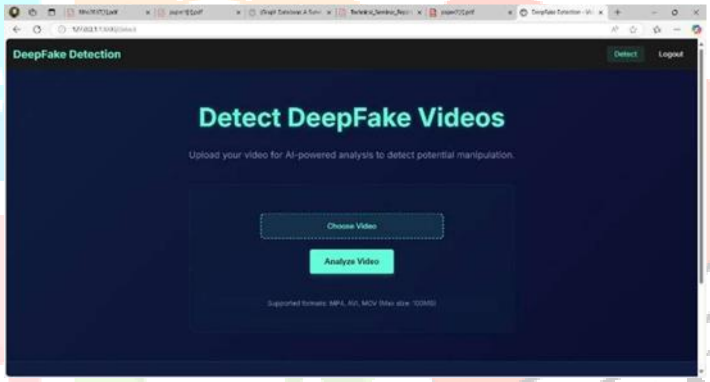


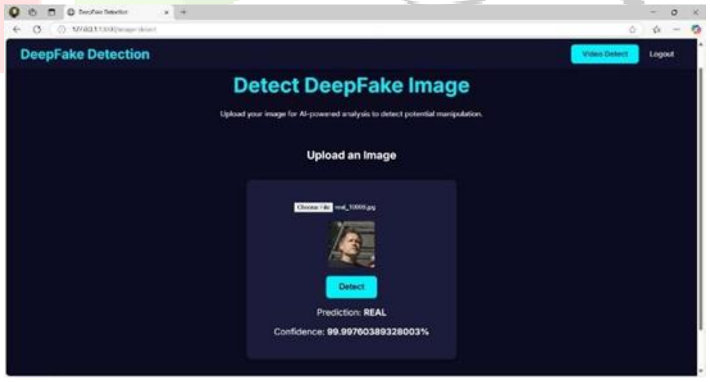
Figure 5: This is a login page where user can login and check the vedios and images are fake or not



Upload your video for AI-powered analysis to detect potential manipulation.

Supported formats: MP4, AVI, MOV (Max size: 100MB)

Figure 6: A Interface To Upload an Vedio to predict fake or not at home page



Upload your image for AI-powered analysis to detect potential manipulation.

Prediction: REAL

Confidence: 99.99760389328003%

Figure 7: A User uploads an image and gets to know whether it is fake or real along with confidence score

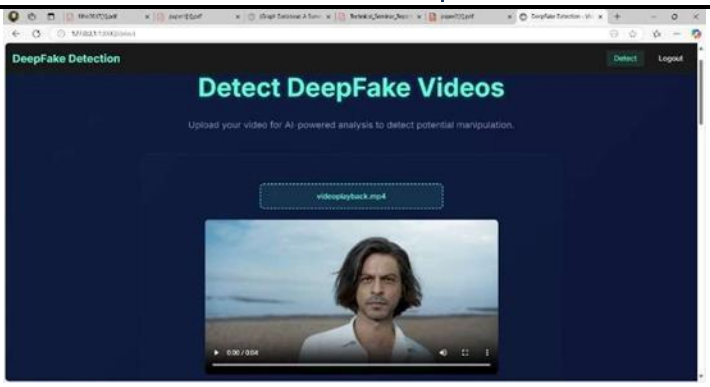

DeepFake Detection
Detect
Logout
Detect DeepFake Videos
Upload your video for AI-powered analysis to detect potential manipulation.
videoplayback.mp4

Figure 8: A Uploaded Vedio file to Know it is fake or not

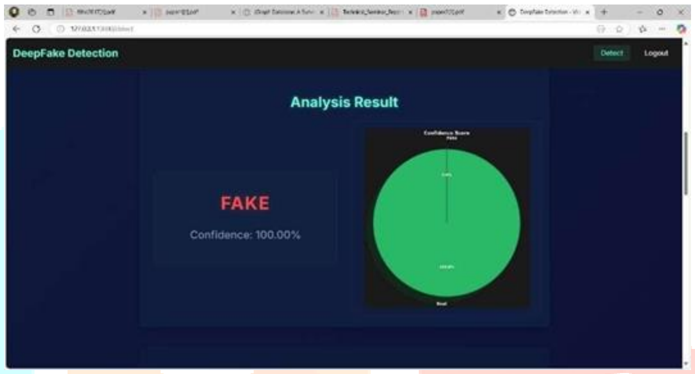


Figure 9:Predicts It Fake or Real Image Along with the Confidence Score

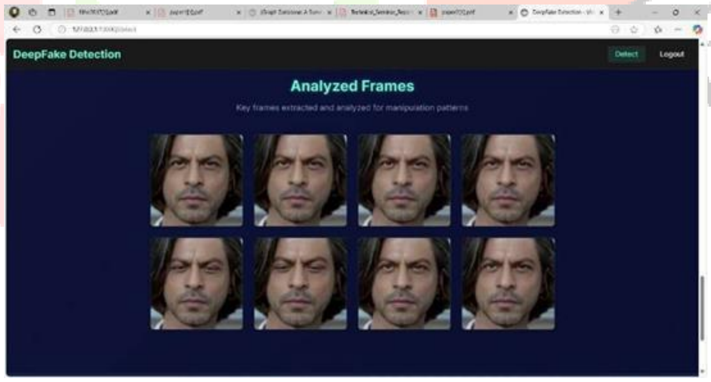


Figure 10: Shows the user on hat basis the prediction has been made with frames movement and analysing the irregularities between the frames

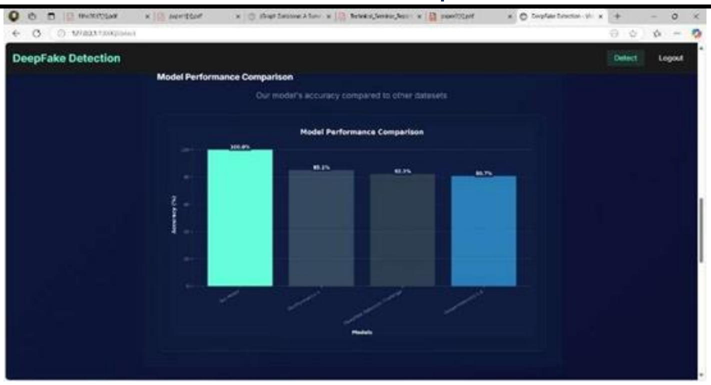

DeepFake Detection
Detect
Logout
Model Performance Comparison
Our model's accuracy compared to other datasets
Model Performance Comparison
100.0%
85.1%
82.3%
80.7%
Accuracy (%)
Models

Figure 11: This Figure Shws the compared models For Prediction

VII. FUTURE WORK

There is always scope for enhancements in any developed system, especially when the project is built using the latest trending technologies and addresses a growing area of concern such as deepfake detection. As the field of AI- generated media continues to evolve, so must the systems designed to counter them.

One significant enhancement would be upgrading the current web-based platform into a browser plugin or extension, allowing users to seamlessly verify the authenticity of videos they encounter while browsing online platforms like YouTube,

Facebook, or Twitter. This would enhance usability and accessibility, bringing real-time detection to users without requiring them to manually upload content.

Additionally, while the current system focuses primarily on face-based deepfake detection, the scope can be expanded to detect full-body deepfakes, which are becoming increasingly common in manipulated media. This would involve training models to recognize inconsistencies in body posture, movements, and synchronization with facial expressions— areas that are often less refined in AI- generated videos.

Future work can also explore multimodal deepfake detection, integrating audio analysis to detect manipulated speech or voice cloning, alongside visual cues. Incorporating explainable AI (XAI) techniques would provide transparency in how predictions are made, increasing trust in the system, especially for use in legal and media verification contexts.

Moreover, the detection model can be optimized for mobile and edge devices, making it suitable for offline and low-resource environments. This would be particularly beneficial in regions with limited internet access or in field operations where cloud computing is not viable.

As deepfake techniques become more sophisticated, continuous training with newer datasets, proactive adversarial testing, and collaborative research will be essential to ensure the model remains robust, adaptive, and future-ready.

may also include the application of explainable AI (XAI) to interpret CNN decisions and improve transparency in medical diagnostics.

VII. CONCLUSION

We presented a robust, neural network-based approach to classify videos as either deepfake or real, accompanied by a confidence score that reflects the model's certainty. This system demonstrates the potential of artificial intelligence in combating the misuse of AI-generated synthetic media. Our method effectively analyzes just 1 second of video (at 10 frames per second) to determine authenticity with commendable accuracy, making it highly suitable for near real-time detection.

The implementation leverages a pre-trained ResNeXt Convolutional Neural Network (CNN) for efficient frame-level feature extraction. These extracted features are then passed into a Long Short-Term Memory (LSTM) based Recurrent Neural Network (RNN), which is capable of learning the temporal dynamics between successive

frames — such as subtle facial inconsistencies, unnatural movements, or inconsistent blinking patterns — that typically occur in deepfake videos. This hybrid architecture enables the system to detect manipulations between frames t and t-1 with precision.

Moreover, the model has been designed to process sequences of varying lengths, specifically at frame counts of 10, 20, 40, 60, 80, and 100, offering flexibility for deployment in diverse scenarios — whether it's short social media clips or longer video content. The adaptability across different temporal depths further enhances its utility for developers, forensic analysts, and platform moderators who require tailored solutions for deepfake detection. Overall, the combination of spatial and temporal modeling makes our approach both accurate and scalable, and its ability to deliver results with minimal input positions it as a promising solution in the fight against AI-generated misinformation and digital manipulation.

IX. REFERENCES

G. Nagaraj and N. Channegowda, "Video Forgery Detection using an Improved BAT with Stacked Auto Encoder Model," Journal of Advanced Research in Applied Sciences and Engineering Technology, vol. 42, no. 2, pp. 175–187, 2024.

C. Girish and C. Nandini, "Detection of Frame Duplication Using Multi-Scale Local Oriented Feature Descriptors," International Journal of Intelligent Engineering Systems, vol. 14, no. 3, 2021.

[DOI: 10.22266]

N. Girish and C. Nandini, "Inter-Frame Video Forgery Detection Using UFS-MSRC Algorithm and LSTM Network," International Journal of Modelling, Simulation, and Scientific Computing, 2023. [DOI:

https://doi.org/10.1142/S1793962323410131]

S. Jahnavi and C. Nandini, "Secure Two-Fold Transmission of Features Using Multimodal Mask Steganography and Naive-Based Random Visual Cryptography System," International Journal of System Assurance Engineering and Management, 2022. [DOI: https://doi.org/10.1007/s13198-022-

01701-6]

S. Jahnavi and C. Nandini, "Hybrid Hyper Chaotic Map with LSB for Image Encryption and Decryption," Scalable Computing: Practice and Experience, vol. 23, no. 4, pp. 181–191, Dec. 2022.

[DOI: 10.12694/scpe.v23i4.2018]

S. Merikapudi, S. Math, C. Nandini, and M. Rafi, "A Google Net Assisted CNN Architecture Combined with Feature Attention Blocks and Gaussian Distribution for Video Face Recognition and Verification," International Journal of Electrical Engineering and Technology (IJEET), vol. 12, no. 1, pp. 30–42, Jan. 2021.

C. Nandini and S. Sumanth Reddy, "Detection of Communicable and Non-Communicable Disease Using Lenet-Bi-LSTM Model in Pathology Images," International Journal of System Assurance Engineering and Management, Springer India, 2022. [DOI: 10.1007/s13198-022-01702-5]

C. Girish and C. Nandini, "Detection of Frame Duplication Using Multi-Scale Local Oriented Feature Descriptors," International Journal of Intelligent Engineering Systems, vol. 14, no. 3, 2021. [DOI:

10.22266]

M. Sandeep and C. Nandini, "An Extensive Survey on 3D Face Reconstruction Based on Passive Method," International Research Journal of Engineering and Technology (IRJET), vol. 8, no. 12, Dec. 2021.

A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," arXiv preprint, arXiv:1901.08971, 2019.

J. Thies, M. Zollhöfer, M. Stamminger, T. Theobalt, and M. Nießner, "Face2Face: Real-Time Face Capture and Reenactment of RGB Videos," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 2387–2395, June 2016.

A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," arXiv preprint, arXiv:1901.08971, 2019.

Y. Li, M.-C. Chang, and S. Lyu, "Exposing AI Created Fake Videos by Detecting Eye Blinking," arXiv preprint, arXiv:1806.02877v2, 2018.

ResNext Model. Available: https://pytorch.org/hub/pytorch_vision_resnext/, Accessed: Apr. 6, 2020.

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), Auckland, New Zealand, 2018, pp. 1–6.

K. Li, H. Qi, and B. Li, "HOHO: A Dataset for Human-Object Human-Object Interactions," in Proc. 2020 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1125–1133, 2020.

A. Dhiman, N. B. Gupta, and P. K. Shukla, "Deepfake Detection Using Capsule Networks," in Proc.

2019 IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 415–423, 2019.

R. Yadav and M. Jain, "Deepfake Video Detection Using CNN and LSTM," International Journal of Computer Applications, vol. 182, no. 9,

pp. 13–20, May 2020.

R. H. Zhang, C. J. Zhang, and T. C. Zhang, "Fake Portrait Detection Using Biological Signals from Facial Regions," IEEE Transactions on Information Forensics and Security, vol. 12, no. 7,

pp. 1923–1935, 2018.