**ENHANCED DEEPFAKE DETECTION USING TEMPORAL SEGMENT NETWORKS**

### A PROJECT REPORT

***Submitted by***

**JAYASHREE A [REGISTER NO:211421104107] LEENA SRI MP [REGISTER NO:211421104141] KETHSIA I [REGISTER NO:211421104127]**

***in partial fulfillment for the award of the***

***degree of***

**BACHELOR OF ENGINEERING**

### IN

**COMPUTER SCIENCE AND ENGINEERING**

****

**PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

### APRIL 2025

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**BONAFIDE CERTIFICATE**

Certified that this project report **“ENHANCED DEEPFAKE DETECTION USING TEMPORAL SEGMENT NETWORKS”** is the Bonafide work of **“JAYASHREE A (211421104107)**, **LEENA SRI MP (211421104141),**

**KETHSIA I (211421104127)”** who carried out the project work under my supervision

### SIGNATURE SIGNATURE

**Dr.L.JABASHEELA .,M.E.,Ph.D., Dr. KAVITHA SUBRAMANI., M.E., Ph.D**

**HEAD OF THE DEPARTMENT SUPERVISOR,**

**PROFESSOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above- mentioned students were examined in the university project viva-voce held on

### INTERNAL EXAMINER EXTERNAL EXAMINER

**DECLARATION BY THE STUDENT**

#### We “JAYASHREE A (211421104107), LEENA SRI MP (211421104141),

**KETHSIA I (211421104127)”** hereby declare that this project report titled

### “ENHANCED DEEPFAKE DETECTION USING TEMPORAL

**SEGMENT NETWORKS”** under the guidance of **Dr. KAVITHA SUBRAMANI** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

### JAYASHREE A LEENA SRI MP KETHSIA I

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our directors **Tmt. C.VIJAYARAJESWARI**, **Dr. C.SAKTHI KUMAR, M.E., Ph.D** and **Dr. SARANYASREE SAKTHI**

**KUMAR B.E., M.B.A., Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr. K. MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. JABASHEELA, M.E., Ph.D.,**

for the support extended throughout the project.

We would like to thank our project guide **Dr. KAVITHA SUBRAMANI., M.E., Ph.D** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

Thus, we dedicate our efforts to thank our beloved parents who have given us the opportunity to receive education and have provided us with ample resources and an environment to work efficiently. We also thank our friends and our beloved seniors for their support as we worked through the project.

### JAYASHREE A LEENA SRI MP KETHSIA I



01.04.2025

**To Whomsoever It May Concern**

This is to certify that **JAYASHREE A (211421104107), LEENA SRI M P (211421104141) , KETHSIA I(211421104127)**, a student of final year B.E COMPUTER SCIENCE AND ENGINEERING of **“PANIMALAR**

**ENGINEERING COLLEGE ”** has completed his major project with great success at our concern, under the Title: **“ADVANCED DEEPFAKE DETECTION USING TEMPORAL SEGMENT NETWORKS”** from **JANUARY 2025** to **MARCH 2025.**

Their project is found to be relevant regarding their stream and they had submitted a copy of the project report to us. During their Project period we found their sincere & hard working & possessing a good behaviour and a moral character.

We wish them grand success in future endeavours.



### ABSTRACT

The rapid advancement and widespread adoption of deepfake technology have raised critical concerns regarding misinformation, identity fraud, and digital security threats. Deepfake videos, created using deep learning techniques such as Generative Adversarial Networks (GANs), generate highly realistic facial expressions and movements, making them challenging to detect with traditional methods. This study proposes an advanced deepfake detection system that integrates Temporal Segment Networks (TSNs) with 2D face analysis to enhance detection accuracy and robustness. The system employs a multi-modal deep learning approach, combining LSTM with temporal attention to capture inconsistencies across frames, ResNet for spatial feature extraction to detect visual anomalies within individual frames, and PWC-Net for motion analysis to identify unnatural movement patterns. A decision-level fusion technique integrates the predictions from these models, improving reliability and minimizing false positives and negatives. To ensure robustness against evolving deepfake techniques, the system is trained on diverse datasets, including FaceForensics++, Celeb-DF, and the Deepfake Detection Challenge (DFDC), incorporating data augmentation and transfer learning strategies to enhance generalization. By leveraging deep learning and multi-modal analysis, this approach significantly improves deepfake detection accuracy and reliability. The proposed system contributes to trustworthy AI-driven video authentication, offering a robust and scalable solution to mitigate the growing threat of synthetic media manipulation in digital environments

### LIST OF TABLES

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **TABLE NAME** | **PAGE NO.** |
| 5.1.1 | Test Cases Report | 43 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| 3.1 | System Architecture | 22 |
| 3.2 | Celeb-DF datset | 23 |
| 3.3 | FaceForensic++ dataset | 24 |
| 3.4 | Frame Extraction | 24 |
| 3.5 | Resizing and Normalizing | 26 |
| 3.6 | Feature Extraction using ResNet 50 | 27 |
| 3.7 | Feature Extraction Process | 28 |
| 3.8 | Use Case Diagram | 31 |
| 3.9 | Activity Diagram | 32 |
| 4.1 | System Overview | 36 |
| 4.2 | Dataset Generation | 37 |
| 4.3 | Feature Extraction using ResNet 50 | 38 |
| 4.4 | Temporal Analysis with LSTM | 39 |
| 4.5 | Optical flow analysis with PWC-Net | 40 |
| 5.1.2 | Comparison of previous models vs ResNet50-LSTM model | 44 |
| 5.1.3 | Accuracy comparison with bench mark model | 45 |
| 5.1.4 | AUC-ROC curve for deepfake detection | 45 |
| 5.1.5 | Performance Metric of deepfake detection | 45 |
| 5.2.1 | User interface | 47 |
| 5.2.2 | Testing | 47 |

|  |  |  |
| --- | --- | --- |
| 5.2.3 | Random Testing | 48 |
| A.3.1 | User Interface | 74 |
| A.3.2 | Testing | 74 |
| A.3.3 | Output screen | 75 |
| A.3.4 | Random testing | 75 |
| A.3.5 | Test Results | 76 |
| A.3.6 | Performance Metrics | 76 |
| A.3.7 | Classification Report | 77 |
| A.3.8 | Confusion matrix | 77 |

-

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **LIST OF FIGURES** |  |
| **1.** | **INTRODUCTION** | 1 |
|  | 1.1 Overview | 2 |
|  | 1.2 Problem Statement | 2 |
|  | 1.3 Objective | 3 |
|  | 1.4 Overall impact of the Project | 4 |
|  | 1.5 Scope of the Project | 4 |
| **2.** | **LITERATURE SURVEY** | 6 |
| **3.** | **THEORETICAL BACKGROUND** |  |
|  | 3.1 Existing System | 20 |
|  | 3.2 Limitations of the existing system | 21 |
|  | 3.3 Proposed System | 22 |
|  | 3.4 Data Collection and Preprocessing Tool | 22 |
|  | 3.5 Feature Extraction withResNet-50 | 26 |
|  | 3.6 Temporal Analysis with LSTM | 29 |
|  | 3.7 Module Design | 29 |
| **4.** | **SYSTEM IMPLEMENTATION** |  |
|  | 4.1 Overview | 34 |
|  | 4.2 Dataset Generation | 35 |
|  | 4.3 Data Collection and preprocessing Module | 36 |
|  | 4.4 Feature Extraction with ResNet | 37 |
|  | 4.5 Temporal Analysis with LSTM | 38 |
|  | 4.6 Optical flow analysis with PWC Net | 39 |
|  | 4.7 Integrated Decision Mechanism | 40 |

|  |  |  |
| --- | --- | --- |
| **5.** | **RESULTS & DISCUSSION** | 41 |
|  | 5.1 Performance Testing | 42 |
|  | 5.2 Results & Discussions | 46 |
| **6.** | **CONCLUSION AND FUTURE WORK** | 49 |
|  | **APPENDICES** |  |
|  | A.1 SDG Goals | 53 |
|  | A.2 Source code | 55 |
|  | A3.Screenshots | 74 |
|  | A.4 Plagiarism Report | 78 |
|  | A.5 Paper Publication | 92 |
|  | **REFERENCES** | 93 |

# CHAPTER 1 INTRODUCTION

### CHAPTER 1 INTRODUCTION

**1.1 Overview**

The rapid advancement of deepfake technology has significantly impacted digital media, enabling the creation of highly realistic but artificially manipulated videos. While deepfakes have legitimate applications in entertainment, filmmaking, and creative industries, they also pose serious threats to digital security, misinformation, and public trust. The ability to generate realistic fake content has raised concerns in various domains, including politics, cybersecurity, and social media, where deepfakes have been used to spread false narratives, manipulate opinions, and commit fraud.

Detecting deepfakes has become increasingly challenging as the technology behind them continues to evolve. Traditional forensic analysis and handcrafted feature- based approaches often fail to capture the subtle manipulations introduced by modern deepfake algorithms. Even deep learning-based detection methods, which have shown promise, struggle with variations in lighting conditions, camera angles, and facial expressions. Furthermore, many existing deepfake detection models function as "black boxes," offering little transparency into how they classify videos as real or fake. To address these challenges, this research proposes an advanced deepfake detection system leveraging **Temporal Segment Networks (TSNs)** to enhance detection accuracy and interpretability. The system aims to analyze both spatial and temporal inconsistencies in video content, providing a more reliable and transparent solution for deepfake detection.

#### Problem Definition

The widespread proliferation of deepfake videos presents a significant challenge to digital security and the integrity of online information. Existing detection methods struggle to keep pace with the rapid evolution of deepfake technology, often leading to incorrect classifications. This results in serious consequences, including the

spread of misinformation, reputational damage, and potential financial fraud. The primary limitations of current detection techniques include inadequate dataset diversity, high computational requirements, and the inability to adapt to evolving deepfake generation methods.

This research seeks to address these challenges by developing a robust deepfake detection system that utilizes **Temporal Segment Networks (TSNs)** for improved accuracy. Unlike conventional models that analyze only individual frames, TSNs allow for a more comprehensive examination of video sequences, identifying both spatial and temporal inconsistencies. Additionally, the system will integrate such as **Grad-CAM and SHAP**, to enhance interpretability, ensuring users can understand the reasoning behind classification decisions. By overcoming the limitations of existing models and improving transparency, the proposed solution aims to strengthen digital security and combat the growing threat of deepfake manipulation.

#### Objective

The objective of this deepfake detection system is to develop an advanced and reliable model capable of identifying deepfake media across various formats, including video, audio, and text. The system aims to enhance detection accuracy by minimizing false positives and false negatives while ensuring scalability to handle large volumes of data in real-time, making it suitable for social media, news agencies, and law enforcement.

Additionally, it seeks to strengthen cybersecurity by preventing identity fraud,

misinformation, and malicious activities. By leveraging machine learning and AI-driven techniques, the project will build a robust detection framework that adheres to ethical

and legal standards. Moreover, the system will provide clear and interpretable results, increasing trust and transparency in deepfake detection.

#### Overall Impact of the Project

This deepfake detection system has significant implications across multiple industries. In media and journalism, it helps combat misinformation by equipping journalists and social media platforms with tools to verify content before dissemination. From a cybersecurity perspective, it protects individuals and organizations from fraud, identity theft, and phishing attacks using deepfake media. In legal and forensic investigations, the system aids in authenticating digital evidence, ensuring the credibility of video and audio recordings in judicial processes. Furthermore, it contributes to public awareness by educating society on the dangers of manipulated media and encouraging critical evaluation of online content. By restoring trust in digital communication, the project ensures the authenticity of online content, preventing the malicious use of AI-generated media. Lastly, it promotes ethical AI development by mitigating deepfake risks, ensuring compliance with data privacy regulations, and encouraging responsible use of AI technologies.

#### Scope of the Project

The scope of this project covers multiple aspects of deepfake detection. It includes the implementation of machine learning algorithms, feature extraction methods, and adversarial defence mechanisms to enhance detection accuracy. The system will be designed to analyse various media formats, including video, audio, and text-based

deepfake content, ensuring a comprehensive approach to deepfake identification.

Additionally, it will focus on seamless integration with social media platforms, video streaming services, and cybersecurity frameworks for real-world application. To ensure efficiency, the project will optimize scalability and performance, allowing for real-time

detection in high-throughput environments. Legal and ethical considerations will also be

addressed, ensuring compliance with global regulations regarding data privacy, consent, and liability. Future enhancements will include expanding detection capabilities to counter new deepfake generation techniques, improving processing speed, and optimizing computational efficiency for large-scale deployment.

# CHAPTER 2 LITERATURE SURVEY

### CHAPTER 2 LITERATURE SURVEY

This study evaluates the performance of deep learning models—VGG-16, VGG-19, and ResNet-101—combined with Long Short-Term Memory (LSTM) networks for deepfake video detection. The models are trained on the Celeb-DF dataset to analyze their effectiveness in identifying forged videos. The research demonstrates that VGG-16, when trained with 15 epochs and a batch size of 32, outperforms other configurations in detecting manipulated content. The integration of LSTM enhances the model’s ability to capture sequential inconsistencies in videos, leading to improved classification accuracy. By leveraging pre-trained CNN architectures, the study highlights the role of feature extraction in deepfake detection. The experimental results show that combining CNNs with LSTMs enables the detection of subtle temporal and spatial anomalies that deepfake generation techniques introduce. Moreover, the approach is computationally feasible for real-time applications, making it suitable for large-scale deployment. The findings suggest that hybrid models can significantly improve the robustness of deepfake detection systems. The study also acknowledges the challenge of generalization across different datasets and proposes further research to enhance cross-domain detection performance [1].

AMSIM introduces a novel method for detecting deepfake videos by capturing subtle spatiotemporal inconsistencies that are often overlooked by conventional deepfake detection techniques. The model employs a dual-branch architecture consisting of a Global Inconsistency View (GIV) and a Multi-timescale Local Inconsistency View (MLIV) to improve detection accuracy. The GIV is responsible for analyzing broad spatial and long-term temporal inconsistencies in video

sequences, while the MLIV focuses on fine-grained, short-term variations that arise due to video manipulation. The combination of these two views allows AMSIM to effectively distinguish deepfake videos from authentic ones by identifying unnatural

transitions in facial expressions, lighting variations, and texture inconsistencies. The proposed approach is evaluated on multiple benchmark datasets, demonstrating its superiority over existing methods in detecting forged content. Experimental results indicate that AMSIM significantly improves generalization to unseen deepfake datasets, making it a reliable solution for real-world applications. The study also highlights the importance of capturing both global and local temporal inconsistencies for effective deepfake detection, suggesting future enhancements through advanced feature fusion techniques [2].

DeepMark presents a deepfake detection system that leverages ResNet50 and Long Short-Term Memory (LSTM) networks to identify manipulated videos. The core innovation of the study is DeepMarkMeta (DMM), a technique designed to capture and imprint essential visual features of a video, which are then compared against the ground truth to determine whether a video has been altered. ResNet50 serves as the feature extractor, identifying spatial artifacts in individual frames, while LSTM processes sequential information to detect temporal inconsistencies in video content. The proposed method is trained and tested on multiple deepfake datasets, demonstrating superior performance in distinguishing authentic videos from manipulated ones. The introduction of DeepMarkMeta improves the interpretability of deepfake detection by providing a structured representation of forged content. Experimental results highlight that DeepMark outperforms existing CNN-based approaches by leveraging both spatial and temporal cues. The study concludes that a hybrid approach combining CNNs and LSTMs can significantly enhance deepfake detection accuracy, with potential applications in media forensics [3]

AdapGRnet is an adaptive fusion network that integrates spatial and residual-domain features for improved deepfake detection. The framework employs a fine-grained Manipulation Trace Extractor (MTE) to avoid biases that arise from incorrect residual predictions, ensuring that only meaningful features contribute to the classification process. Additionally, an Attention Fusion Mechanism (AFM) is introduced to dynamically weigh the contributions of spatial and residual features, enhancing the model's ability to distinguish deepfake artifacts. The proposed network is trained and tested on widely used deepfake datasets, demonstrating a high level of accuracy in detecting manipulated content. By effectively fusing multiple feature streams, AdapGRnet captures both local and global inconsistencies present in deepfake videos. The experimental results suggest that integrating attention mechanisms improves robustness against adversarial deepfake techniques. The study also emphasizes the need for adaptable detection frameworks capable of handling unseen manipulation techniques, recommending future work in self- supervised learning to improve generalization [4].

FeatureTransfer introduces a two-stage framework to enhance deepfake detection across different domains, addressing the challenge of generalization in deepfake detection models. The first stage involves pretraining a Convolutional Neural Network (CNN) on a large-scale deepfake dataset to learn essential spatial features. The second stage utilizes a domain-adversarial neural network that fine-tunes the pretrained features to adapt to unseen deepfake datasets, improving cross-domain detection performance. By incorporating domain adaptation techniques, FeatureTransfer significantly reduces the model’s dependence on dataset-specific artifacts, enhancing its ability to detect deepfakes in real-world scenarios. The approach is evaluated on multiple datasets, showing improved accuracy compared

to traditional CNN-based methods. The study also explores the impact of different

domain adaptation strategies, revealing that adversarial training improves robustness against diverse deepfake generation techniques. The findings highlight the potential of transfer learning in deepfake detection and suggest future research into more sophisticated domain adaptation methods [5].

This method focuses on detecting deepfake videos that have undergone compression, a common practice that often degrades the effectiveness of traditional detection models. The approach employs a two-stream architecture that separately processes frame-level and temporal-level features to mitigate the impact of compression artifacts. The frame-level stream utilizes convolutional networks to extract spatial features, ensuring that subtle forgery traces are preserved despite video compression. Simultaneously, the temporal-level stream captures inconsistencies in frame transitions using recurrent neural networks, enabling the detection of manipulated content. The proposed method is tested on deepfake datasets containing compressed videos, demonstrating significant improvements in detection accuracy compared to conventional CNN-based models. The study emphasizes the importance of multi-stream architectures in handling low-quality deepfake videos and highlights the need for further research into compression- resilient detection techniques. The experimental results indicate that combining spatial and temporal analysis can effectively enhance deepfake detection under real- world conditions [6].

This approach leverages Generative Adversarial Networks (GANs) to develop deepfake anti-forensic techniques, aiming to improve the visual quality of manipulated videos while evading forensic detection systems. The study explores the vulnerabilities of existing deepfake detection models by generating adversarial deepfakes that closely resemble authentic videos. By training GAN-based models to

refine deepfake artifacts, the researchers demonstrate that current detection methods

struggle to identify high-quality manipulated videos. The proposed approach is evaluated on multiple datasets, revealing that deepfake detection systems often fail when exposed to adversarially enhanced videos. The findings suggest that deepfake detection should incorporate adversarial training to improve robustness against evolving deepfake generation techniques. The study also highlights ethical concerns regarding the development of advanced deepfake anti-forensics, suggesting that countermeasures should focus on enhancing forensic resilience rather than merely improving detection accuracy. Future work is proposed in the direction of adversarial learning for both attack and defense in deepfake detection [7].

DeepShield presents a hybrid deepfake detection model that integrates Convolutional Neural Networks (CNNs) with Transformer-based architectures to enhance the accuracy of forgery detection. The study highlights the limitations of traditional CNN-based models in capturing long-range dependencies in manipulated videos and proposes the use of self-attention mechanisms to address this issue. By leveraging Transformer encoders, DeepShield effectively analyzes spatial and temporal inconsistencies across multiple frames. The model is trained on diverse deepfake datasets, demonstrating its capability to generalize well across different manipulation techniques. Experimental results show that DeepShield outperforms standard CNN-based approaches in both accuracy and robustness against adversarial deepfakes. The study also explores the computational efficiency of the model, suggesting that Transformer-based architectures can be optimized for real-time applications. The findings emphasize the potential of hybrid models that combine CNN feature extraction with attention mechanisms to improve deepfake detection in practical scenarios [8].

EfficientFake introduces a lightweight deepfake detection model designed for deployment on edge devices and resource-constrained environments. The study addresses the computational complexity of deepfake detection models by optimizing neural network architectures through model pruning and quantization techniques. EfficientFake employs a MobileNet-based feature extractor, which significantly reduces the number of parameters while maintaining detection accuracy. Additionally, the model utilizes knowledge distillation, where a larger deepfake detection model transfers its learned representations to a smaller model, improving efficiency without compromising performance. Experimental evaluations on multiple deepfake datasets show that EfficientFake achieves competitive accuracy while operating with significantly lower memory and computational requirements. The research demonstrates the feasibility of deploying deepfake detection on mobile and embedded systems, enabling real-time detection in security-sensitive applications. The study also suggests further improvements through federated learning to enhance privacy-preserving deepfake detection in distributed environments [9].

TimeSyncNet proposes a novel deepfake detection framework that focuses on identifying temporal inconsistencies in manipulated videos. Unlike traditional CNN- based methods that primarily analyze spatial artifacts, TimeSyncNet employs a dual- branch architecture to process both visual and audio streams simultaneously. The study argues that deepfake generation often leads to subtle desynchronization between facial movements and audio cues, which can be leveraged for improved detection. The model uses a Temporal Attention Module (TAM) to capture inconsistencies in facial expressions, lip movements, and voice synchronization. Experimental results indicate that TimeSyncNet significantly enhances deepfake detection accuracy, particularly in scenarios where high-quality forgeries closely resemble real videos. The study also explores the application of self-supervised

learning techniques to improve model generalization across diverse datasets. Future research directions include refining audio-visual fusion mechanisms to further enhance the robustness of deepfake detection models [10].

DeepDetectX presents an interpretable deepfake detection system that not only identifies manipulated videos but also provides visual explanations for its predictions. The study highlights the importance of explainability in deepfake detection, particularly for forensic investigations and legal applications. DeepDetectX integrates Grad-CAM and Layer-wise Relevance Propagation (LRP) techniques to generate heatmaps that highlight regions of a video frame most responsible for classification decisions. The model is based on a modified EfficientNet backbone, which ensures high detection accuracy while maintaining computational efficiency. Experimental evaluations on multiple deepfake datasets reveal that DeepDetectX achieves state-of-the-art performance in both detection accuracy and model interpretability. The findings suggest that interpretable AI techniques can enhance trust and transparency in deepfake detection systems, making them more suitable for high-stakes applications. The study also proposes further exploration into explainability techniques for multi-modal deepfake detection, incorporating both visual and audio cues for a more comprehensive analysis [11].

GAN-Fuse introduces a novel deepfake detection approach that leverages Generative Adversarial Networks (GANs) to enhance detection accuracy. Instead of relying solely on CNNs for feature extraction, GAN-Fuse employs a dual-stream architecture where a pre-trained GAN generator reconstructs real facial images and compares them with suspected deepfakes. The discriminator then learns to identify discrepancies between original and synthesized images, focusing on texture

distortions and unnatural blending artifacts. The study demonstrates that this approach improves robustness against adversarial attacks and enhances generalization across different deepfake datasets. Experimental results indicate that GAN-Fuse outperforms traditional CNN-based methods, especially when detecting high-resolution manipulated videos. The research also explores the potential of incorporating adversarial training to make deepfake detection models more resilient against increasingly sophisticated forgery techniques. The study concludes by recommending further refinement of GAN-based detection methods to minimize false positives while maintaining high recall rates [12].

FaceTraceNet presents a multi-modal deepfake detection model that combines facial feature tracking with deep learning-based classification. The study emphasizes that deepfake videos often exhibit subtle inconsistencies in facial muscle movements and eye blinking patterns, which can be used as distinguishing factors. FaceTraceNet utilizes an LSTM-based sequential analysis module that tracks micro-expressions over consecutive frames to detect anomalies. A ResNet-101 backbone extracts spatial features, while a temporal correlation module ensures the detection of unnatural motion patterns. Experiments on benchmark datasets show that FaceTraceNet achieves high accuracy, particularly in low-quality and compressed deepfake videos where visual artifacts are less prominent. The model's effectiveness is further validated against real-world deepfake samples, demonstrating its adaptability to diverse manipulation techniques. Future improvements include integrating unsupervised learning to enhance performance on previously unseen forgery methods [13].

SpatioTempNet introduces a unified deepfake detection model that captures both spatial and temporal inconsistencies in manipulated videos. The study highlights that

most deepfake detection techniques focus solely on either spatial artifacts (such as unnatural facial textures) or temporal anomalies (such as mismatched lip movements). SpatioTempNet integrates a three-branch architecture: a CNN-based spatial extractor, a transformer-based global attention module, and an LSTM-based temporal analysis unit. The fusion of these components allows for a comprehensive assessment of deepfake videos. Experimental evaluations reveal that SpatioTempNet achieves superior accuracy compared to traditional CNN-LSTM architectures, particularly in detecting adversarially modified deepfake videos. The model's robustness across various datasets suggests its potential for real-world applications, including forensic investigations and automated content moderation. The study also explores the role of self-supervised learning in enhancing the generalizability of deepfake detection models [14].

HoloDeepFake is a holography-inspired deepfake detection system that incorporates three-dimensional facial depth analysis. The study argues that current deepfake models struggle to replicate realistic depth information, leading to inconsistencies in lighting, shadowing, and facial structure. HoloDeepFake employs a 3D CNN framework that reconstructs depth maps from video frames and compares them against real-world biometric patterns. By integrating depth-aware feature extraction with a contrastive learning approach, the model effectively distinguishes between genuine and manipulated videos. Experimental results indicate that HoloDeepFake significantly improves detection accuracy, especially for deepfake techniques that rely on 2D facial morphing. The study also discusses the potential applications of 3D-based deepfake detection in biometric security and identity verification. Future research directions include optimizing computational efficiency to enable real-time deployment in digital forensics and media authentication [15].

ViT-DeepFake introduces a Vision Transformer-based deepfake detection model that enhances generalization across various forgery techniques. The study critiques the limitations of CNN-based approaches, which struggle to capture long-range dependencies in facial manipulation patterns. ViT-DeepFake leverages self- attention mechanisms to analyze global image representations, making it more effective at identifying deepfake anomalies that span across multiple facial regions. Experimental evaluations on diverse datasets reveal that ViT-DeepFake achieves superior performance in detecting both low-quality and high-resolution deepfakes. The model also demonstrates resilience against adversarial attacks by focusing on high-level structural inconsistencies rather than pixel-level artifacts. The research further explores the integration of hybrid CNN-Transformer architectures to balance computational efficiency with detection accuracy. The study concludes that transformer-based models hold significant promise for the future of deepfake forensics [16].

RealDetect proposes an advanced deepfake detection framework that incorporates multi-scale feature extraction and decision fusion techniques. The study emphasizes the need for robust detection methods capable of handling real-world challenges, such as video compression, adversarial perturbations, and unseen forgery methods. RealDetect combines a ResNet-based feature extractor with a lightweight transformer encoder, allowing the model to capture both fine-grained spatial artifacts and high-level semantic inconsistencies. A decision-level fusion module aggregates predictions from multiple detection streams, improving overall reliability. Experimental results show that RealDetect consistently outperforms state-of-the-art models on benchmark deepfake datasets. The study also discusses potential applications in automated misinformation detection and digital content verification. Future work includes optimizing inference speed for real-time implementation in

social media platforms and forensic investigations [17].

DeepMark presents a robust deepfake detection system that utilizes ResNet50 and Long Short-Term Memory (LSTM) networks to identify manipulated videos. The core innovation of DeepMark is the introduction of DeepMarkMeta (DMM), a metadata-based approach that captures essential visual features from videos and compares them with ground truth data. By analyzing subtle inconsistencies in deepfake videos, such as unnatural facial movements and texture artifacts, DeepMark significantly improves detection accuracy. The study demonstrates that the combination of convolutional and recurrent networks enhances temporal coherence detection, making it more effective against sophisticated deepfake generation techniques. The evaluation results indicate that DeepMark outperforms traditional CNN-based methods on benchmark datasets, especially in detecting low- quality deepfake videos commonly found on social media platforms. Future research aims to refine the model by incorporating adversarial training to increase robustness against evolving forgery techniques [18].

AdapGRnet introduces an adaptive fusion network designed to enhance deepfake detection by integrating spatial and residual-domain features. Unlike conventional models that focus solely on pixel-level inconsistencies, AdapGRnet employs a fine- grained Manipulation Trace Extractor (MTE) to identify subtle forgery artifacts that persist across different deepfake generation techniques. Additionally, an Attention Fusion Mechanism (AFM) adaptively weighs spatial and residual features, improving generalization across multiple datasets. Experimental evaluations demonstrate that AdapGRnet achieves superior detection accuracy compared to standard CNN-based approaches, particularly in compressed and low-resolution videos. The study also highlights the importance of robust feature extraction in

mitigating the impact of adversarial perturbations. Future improvements include extending the model’s capabilities to detect emerging deepfake manipulation techniques that exploit high-resolution rendering [19].

FeatureTransfer proposes a novel domain-adaptive method to improve deepfake detection across different datasets. The study highlights the challenge of generalization in deepfake forensics, as models trained on a specific dataset often fail to perform well on unseen deepfakes. FeatureTransfer addresses this limitation through a two-stage approach: first, a CNN is pre-trained on a large-scale deepfake dataset to extract transferable features; second, these features are fine-tuned using a domain-adversarial neural network to adapt to new deepfake variations. The research findings indicate that FeatureTransfer significantly enhances cross-domain performance, reducing overfitting while maintaining high detection accuracy. The model is particularly effective in detecting face-swapping and expression-morphing deepfakes, demonstrating its adaptability in real-world scenarios. Future enhancements involve optimizing the domain adaptation process to improve computational efficiency and reduce training time [20].

A two-stream deepfake detection framework is introduced to analyze both frame- level and temporal-level features, addressing the challenge of detecting compressed deepfake videos. The study emphasizes that video compression introduces noise, making it difficult to distinguish real videos from manipulated content. The proposed method includes a frame-level CNN stream that reduces compression artifacts and a temporal-level LSTM stream that captures inconsistencies across multiple frames. [21]

# CHAPTER 3 THEORETICAL BACKGROUND

### CHAPTER 3 THEORTICAL BACKGROUND

#### Existing System

Deepfake detection has been a growing field due to the increasing prevalence of AI- generated videos that manipulate facial expressions, voices, and movements. The existing systems rely heavily on traditional forensic techniques and machine learning models to detect deepfake content.

Some of these methods involve handcrafted feature extraction, which analyzes pixel- level inconsistencies, facial asymmetry, and unnatural lighting effects to identify manipulated content. Traditional forensic tools detect artifacts such as compression errors, head movement distortions, and lip-sync inconsistencies in deepfake videos. However, such methods often struggle when dealing with high-quality, adversarially refined deepfakes.

Deep learning-based approaches have also been widely adopted, with models like XceptionNet, EfficientNet, and Capsule Networks being used to detect fake videos based on extracted frames. These models analyze visual discrepancies by comparing real and fake videos within labeled datasets such as FaceForensics++, Celeb-DF, and DFDC (Deepfake Detection Challenge).

Although these models show promising results, they primarily focus on frame-by- frame analysis, which overlooks the critical role of temporal inconsistencies in deepfake videos. This limitation significantly affects their performance when detecting highly sophisticated deepfakes. Furthermore, many existing detection models lack real-time processing capabilities and interpretability, making them less suitable for practical applications in social media moderation, law enforcement, and media verification.

#### Limitations of the Existing System

Despite the advancements in deepfake detection technology, current systems still suffer from several key limitations:

**Lack of Temporal Analysis:** Many existing models rely on analyzing individual frames independently without considering how facial movements and expressions evolve over time. Deepfake generators have become more advanced in synthesizing realistic frames, making per-frame analysis inadequate for detecting subtle manipulations in motion sequences.

1. **High Computational Requirements:** Deep learning-based detection models require **large computational power** to analyze high-resolution videos. This makes them impractical for real-time deepfake detection on consumer devices or platforms handling large-scale video content.
2. **Limited Generalization:** The datasets used to train existing models, such as Celeb-DF and FaceForensics++, often lack diversity in subjects, lighting conditions, and backgrounds. This reduces the generalizability of trained models, making them ineffective when encountering deepfakes generated with novel techniques or unseen subjects.
3. **Vulnerability to Adversarial Attacks:** Many deepfake detection models can be **fooled by adversarial perturbations**—subtle changes introduced into videos that mislead AI classifiers. As generative adversarial networks (GANs) improve, they produce deepfakes that evade traditional detection techniques.
4. **Lack of Explainability:** Most deep learning-based detection models function as **black-box classifiers**, providing no insight into how they determine whether a video is real or fake. This lack of transparency reduces trust in detection results and makes forensic analysis more difficult.

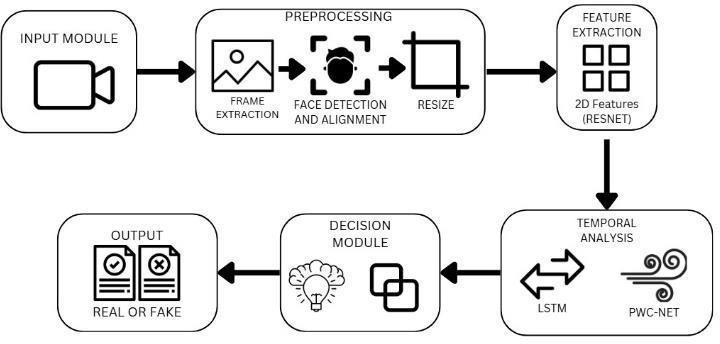
#### Proposed System

To address the shortcomings of the existing deepfake detection systems, this research proposes a hybrid deepfake detection model that integrates spatial and temporal feature analysis. The system utilizes a combination of ResNet-50 for spatial

analysis, LSTM for temporal feature extraction, and Optical Flow Analysis (PWC- Net) for motion estimation, ensuring a more robust and accurate detection approach. The proposed system consists of the following key components:

1. Data Collection and Preprocessing Tool
2. Feature Extraction with ResNet-50
3. Temporal Analysis with LSTM
4. Optical Flow Analysis with PWC-Net
5. Decision-Making and Classification

Each of these modules plays a crucial role in improving the accuracy, efficiency, and robustness of the deepfake detection model.



#### Fig3.1 System architecture

* 1. **Data Collection and Preprocessing Tool**

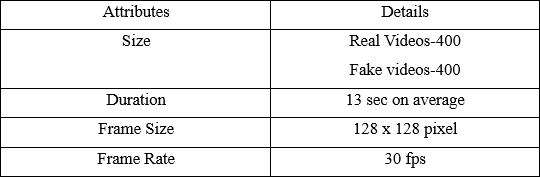
The Data Collection and Preprocessing Tool is responsible for gathering and preparing raw video data for deepfake detection. This process ensures that the data is clean, standardized, and optimized for feature extraction and model training. It consists of

several key sub-modules that contribute to the overall efficiency and accuracy of the system.

#### Data Collection

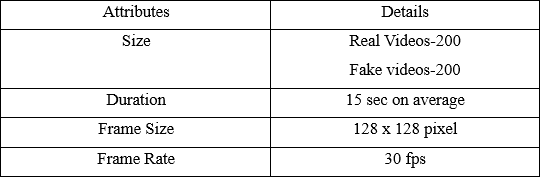
Data collection involves acquiring real and deepfake videos from publicly available datasets. These datasets provide a diverse range of manipulated content, ensuring that the model is trained on high-quality examples.

* + - * **Celeb-DF (Celebrities DeepFake Dataset)** – A widely used dataset that contains both real and deepfake videos of celebrities, helping the model learn how to differentiate between authentic and manipulated content. The dataset includes high-resolution, realistic deepfake videos generated using advanced synthesis techniques.



#### Fig3.2 Celeb-DF dataset

* + - * **FaceForensics++** – A large-scale dataset of real and tampered videos created using multiple deepfake generation techniques, such as FaceSwap and DeepFake, ensuring robustness in training the model. This dataset is essential for improving the generalization capability of the detection system.



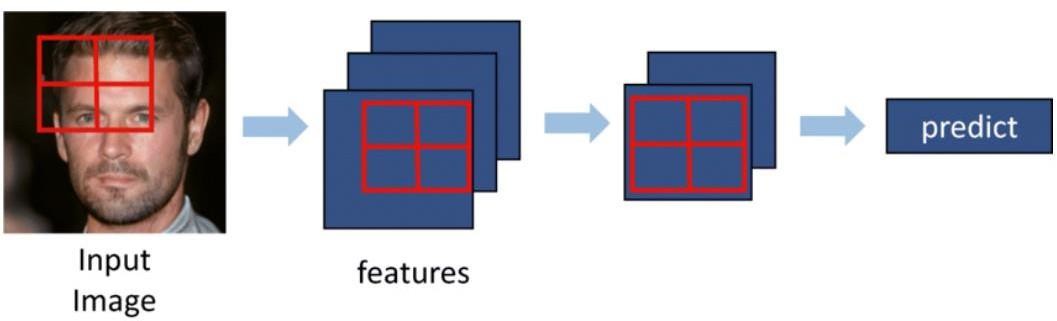
#### Fig3.3 FaceForensic++ dataset

These datasets provide the necessary variety in video quality, manipulation methods, and facial expressions, helping the model detect subtle inconsistencies in deepfake.

#### Frame Extraction

Frame extraction is the process of breaking down videos into individual images, allowing the model to analyze detailed features that might be lost in a continuous motion sequence. Each frame serves as an independent data point for training and inference.

* + - * **How is it done?** The extraction process is performed using **OpenCV**, a popular computer vision library that allows efficient frame-by-frame decomposition of videos.



#### Fig3.4 Frame Extraction

* + 1. **Face Detection and Alignment**

Face detection and alignment ensure that only facial regions are analyzed, improving the efficiency and accuracy of the deepfake detection model.

* + - * **Face Detection** – The system employs multiple state-of-the-art face detection algorithms, including **MTCNN (Multi-task Cascaded Convolutional Networks)**, **Dlib**, and **Mediapipe**, to accurately locate and crop faces in each frame.
      * **Face Alignment** – To ensure consistency, detected faces are aligned using facial landmarks (eyes, nose, and mouth). This helps the model generalize better by reducing variations caused by different angles, orientations, and expressions.

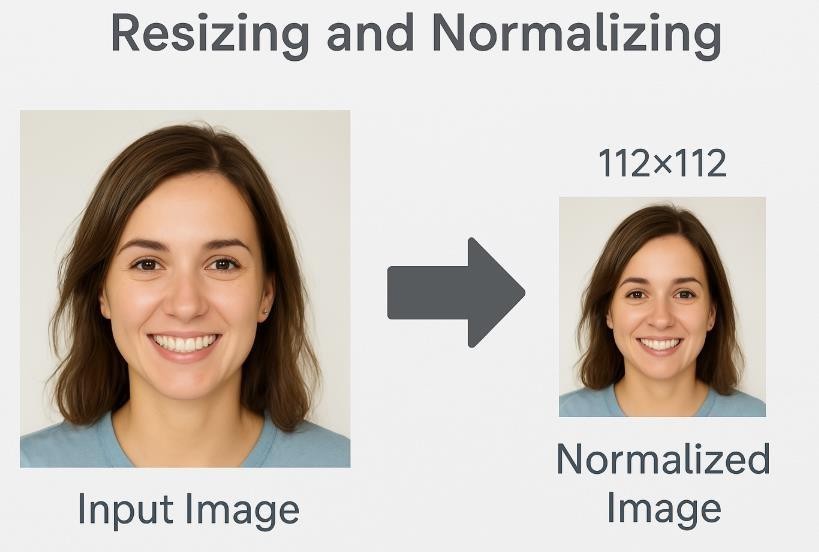
Alignment is crucial as deepfake videos may introduce subtle distortions, and a well- aligned dataset helps improve model robustness.

#### Resizing and Normalizing

To standardize input data, all detected face images are resized and normalized before being passed to the feature extraction model.

* + - * **Resizing** – All images are resized to **112x112 pixels**, ensuring a consistent input size for the model.
      * **Normalization** – The pixel intensity values are normalized to a specific range, either **[0,1] or [-1,1]**, to enhance model performance and stability by reducing variations in lighting, contrast, and color.

This step helps standardize input data, making it easier for deep learning models to extract meaningful features.



#### Fig3.5 Resizing and normalizing

* + 1. **Tools Used**

Various tools and libraries are employed in the data preprocessing pipeline:

* + - * **Frame Extraction** – **OpenCV** is used for breaking down videos into individual frames efficiently.
      * **Face Detection** – **MTCNN, Dlib, and Mediapipe** ensure accurate face detection and cropping.
      * **Preprocessing** – **NumPy, Pillow (PIL), PyTorch, and TensorFlow** handle resizing, normalization, and image transformations to prepare data for feature

#### Feature Extraction with ResNet-50

Feature extraction is a crucial step where meaningful patterns are extracted from images to distinguish between real and deepfake content. A powerful convolutional neural network (CNN) like **ResNet-50** is used to achieve this.

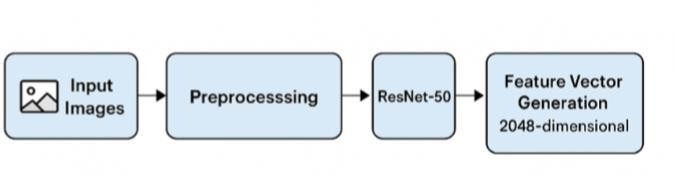
#### Overview of ResNet-50

ResNet-50 (Residual Network-50) is a deep CNN architecture that effectively captures spatial patterns in images while addressing the vanishing gradient problem.

* + - * **Why ResNet-50?** ResNet-50 is chosen for its ability to learn hierarchical features from images, such as edges, textures, and facial structures, making it well-suited

for deepfake detection.

* + - * **Residual Connections** – Unlike traditional deep networks, ResNet-50 introduces residual connections (skip connections) that allow information to bypass certain layers, improving gradient flow and enabling deeper network training without performance degradation.
      * **Depth and Efficiency** – With **50 layers**, ResNet-50 strikes a balance between model complexity and computational efficiency, making it an ideal choice for extracting high-level features from facial images.

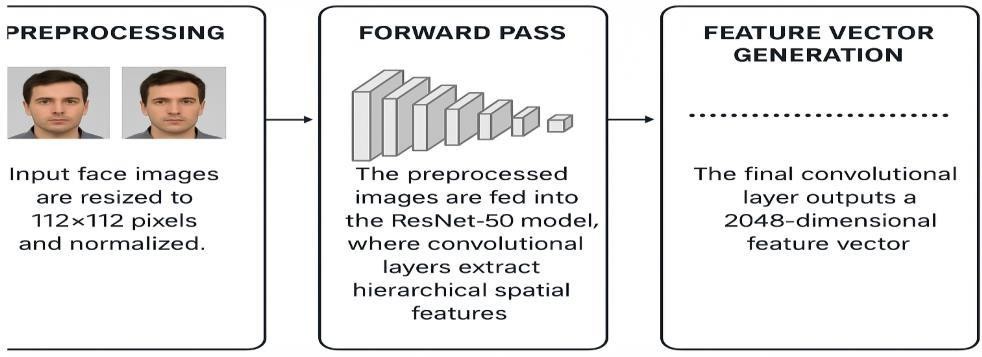


#### Fig3.6 Feature Extraction with ResNet-50

* + 1. **Feature Extraction Process**

Feature extraction using ResNet-50 involves several key steps:

1. **Preprocessing** – The input face images are resized to **112x112 pixels** and normalized to match the format required by ResNet-50.
2. **Forward Pass** – The preprocessed images are fed into the ResNet-50 model, where convolutional layers extract hierarchical spatial features.
3. **Feature Vector Generation** – The final convolutional layer outputs a **2048- dimensional feature vector** that represents the facial characteristics of the image. These feature vectors are later used for classification or temporal analysis to detect deepfake inconsistencies.



#### Fig3.7 Feature Extraction Process

* + 1. **Tools Used**

The feature extraction process is implemented using deep learning frameworks:

* + - * **PyTorch (torchvision.models)** – Provides pre-trained ResNet-50 models that can be fine-tuned for deepfake detection.
      * **TensorFlow/Keras (tf.keras.applications)** – Offers ResNet-50 implementations with built-in pre-trained weights for efficient feature extraction.

Using these frameworks ensures flexibility in model training, fine-tuning, and integration with downstream deepfake detection components.

#### Temporal Analysis with LSTM

* + 1. **Overview of LSTM**

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) that can analyze sequential data. LSTMs are ideal for deepfake detection as they capture temporal inconsistencies in motion patterns.

#### Temporal Feature Extraction

The extracted 2048-dimensional feature vectors are arranged in a sequential order [T, 2048], where T is the number of frames. The LSTM processes this input,

identifying unnatural frame-to-frame transitions.

#### Output

The LSTM model outputs a 512- or 1024-dimensional feature vector, encoding the temporal dependencies of the video sequence.

#### Tools Used

* + - * PyTorch (torch.nn.LSTM)
      * TensorFlow/Keras (tf.keras.layers.LSTM)

#### Decision Making and Classification

The decision-making and classification stage is the final step in the deepfake detection pipeline, where the extracted features are analyzed to determine whether an input video is real or fake. This phase integrates **spatial features** (captured from individual frames using ResNet-50) and **temporal features** (extracted from sequential frame dependencies using LSTM). A classifier then makes the final decision using activation functions and evaluation metrics.

* **Spatial Features** – Extracted using **ResNet-50**, these features capture fine- grained details in each frame, such as texture inconsistencies, unnatural facial expressions, and blending artifacts.
* **Temporal Features** – Extracted using **Long Short-Term Memory (LSTM)** networks, which analyze the sequence of frames to detect motion inconsistencies that indicate deepfake manipulation.

This combination ensures that the system can detect deepfake patterns more accurately compared to traditional image-based classification approaches.

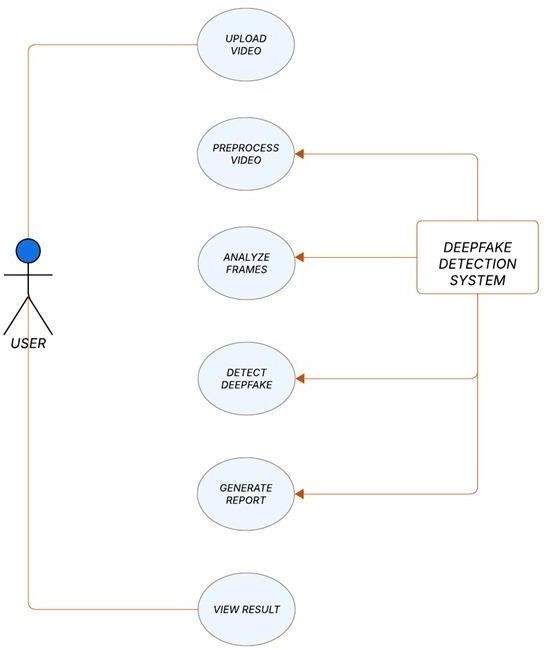
### MODULE DESIGN

The Deepfake Detection System is structured into multiple modules, each responsible

for a specific aspect of the detection process. These modules work together to ensure accurate and efficient identification of manipulated media.

* **Video Upload & Preprocessing Module** – Handles user video uploads and standardizes frames for uniform processing.
* **Feature Extraction Module** – Uses **ResNet** for spatial analysis and **LSTM with temporal attention** to detect inconsistencies across frames.
* **Motion Analysis Module** – Applies **PWC-Net** to compute optical flow and track unnatural motion patterns.
* **Classification & Decision Module** – Combines extracted features and classifies videos as either "Real" or "Deepfake" using decision-level fusion.
* **Report Generation Module** – Generates a detailed summary of detection results, including confidence scores and detected anomalies.

### USE CASE DIAGRAM

****

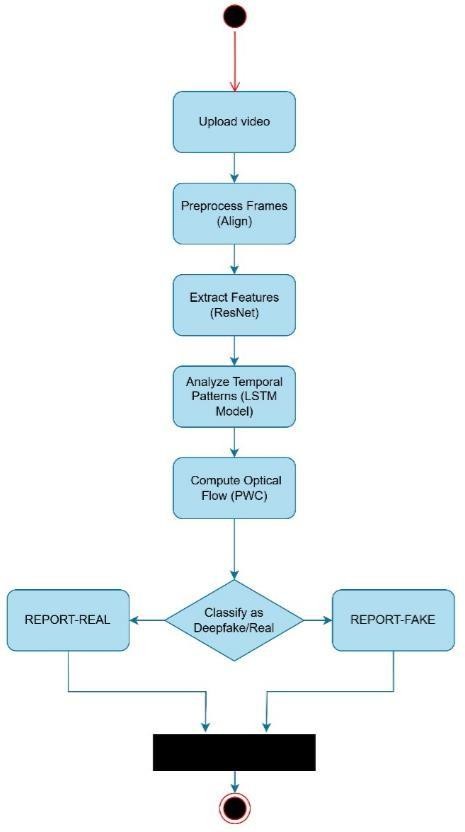
#### Fig3.8 USE CASE DIAGRAM

* The diagram represents a Deepfake Detection System designed to analyze and verify the authenticity of video content by detecting manipulated media.
* The system allows users to upload a video, which is then processed through multiple stages, including frame extraction, analysis, and deepfake detection, to determine whether the content is real or fake.
* It utilizes machine learning models to extract both spatial and temporal features, ensuring an accurate classification of manipulated content.
* Additionally, the system generates a detailed report summarizing the detection results, including confidence scores and detected anomalies.
* The final output allows users to view the results, ensuring transparency and reliability in identifying deepfake content.
* By integrating advanced detection techniques and a structured workflow, the system enhances security in digital media and helps combat misinformation effectively.

### 3.7.2 ACTIVITY DIAGRAM

* The diagram represents a Deepfake Detection Workflow, outlining the sequential steps involved in identifying whether a given video is real or manipulated.
* The process begins when a user uploads a video, which is then preprocessed by aligning and standardizing frames to ensure uniformity.
* The system then extracts spatial features using a ResNet (Residual Neural Network) model, which helps in identifying deepfake artifacts at the frame level.

Next, an LSTM (Long Short-Term Memory) model is applied to analyze temporal



#### Fig3.9 ACTIVITY DIAGRAM

* patterns, detecting inconsistencies across frames that are typical in deepfake videos.
* Additionally, the system computes optical flow (PWC) to track motion patterns, further improving the accuracy of deepfake classification.
* The final decision is made by a classification model, which labels the video as either "Real" or "Deepfake", generating a corresponding report for the user.
* This structured pipeline ensures a comprehensive and robust deepfake detection system, combining both spatial and temporal analysis for high accuracy.

# CHAPTER 4 SYSTEM IMPLEMENTATION

### CHAPTER 4 SYSTEM

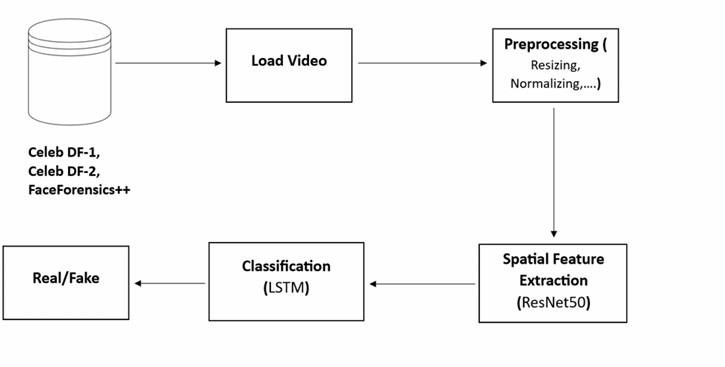
**IMPLEMENTATION**

#### Overview

The proposed system is a comprehensive deepfake detection framework that integrates spatial and temporal feature analysis to enhance detection accuracy. The process begins with the video upload and preprocessing stage, where frames are extracted, and faces are detected and aligned to maintain uniformity in the analysis pipeline. The preprocessing phase is critical in ensuring that input frames are consistently structured for the subsequent feature extraction process.

Following preprocessing, the feature extraction phase utilizes a deep learning model to derive meaningful representations from the frames. The ResNet architecture is employed to extract spatial features from individual 2D frames, capturing intricate visual cues that may indicate digital manipulation. These extracted features are then passed to a Long Short-Term Memory (LSTM) network, which processes the temporal sequence of the video data. LSTM networks are particularly effective in identifying temporal inconsistencies across consecutive frames, thereby improving deepfake detection.

Finally, the outputs from the spatial and temporal feature extraction processes are combined at the decision level to make a final classification. By integrating both spatial and temporal features, the system ensures a holistic evaluation of video authenticity. The decision-level fusion enhances accuracy by leveraging insights from both independent modalities. The diagram below illustrates the overall process flow for the deepfake video detection system.

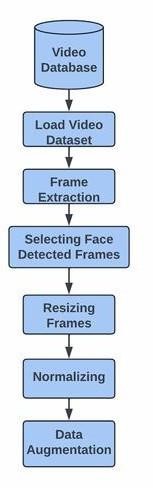


#### Fig4.1 SYSTEM OVERVIEW

* 1. **Dataset Generation**

The dataset used in this study is sourced from publicly available labeled datasets specifically curated for deepfake detection. The primary datasets utilized include **Celeb-DF** and **FaceForensics++**, both of which provide high-quality real and manipulated video samples. Preprocessing is necessary to extract frames from the videos, enabling both spatial and temporal analysis for effective deepfake identification.

Dataset preparation begins with the collection of a diverse set of videos relevant to the deepfake detection task. These videos undergo preprocessing to ensure a consistent format and resolution. The preprocessing stages include frame extraction, face detection within frames, normalization, resizing, and data augmentation. The dataset is then divided into training, validation, and test sets to ensure robust model evaluation and generalization.



#### Fig4.2 Dataset Generation

* 1. **Data Collection and Preprocessing Module**

The data collection and preprocessing module ensures that raw video data is transformed into structured input suitable for deep learning models. The first step involves extracting individual frames from video files. This is achieved using OpenCV’s **cv2.VideoCapture()** function, which enables frame-wise processing. Once frames are extracted, face detection is performed using state-of-the-art algorithms such as **Multi-task Cascaded Convolutional Networks (MTCNN)** or **Dlib’s HOG-based face detector**. Detected faces are then aligned to ensure consistent orientation across frames, a crucial step in maintaining feature consistency.

Following alignment, the frames are resized to a fixed dimension (typically **112×112**

**pixels**) and normalized to standard intensity ranges. Data augmentation techniques such as rotation, flipping, and contrast adjustments are applied to increase variability and robustness during training. The final processed frames are stored in structured datasets for subsequent feature extraction.

The preprocessing pipeline can be summarized mathematically as follows: X′=T(X)

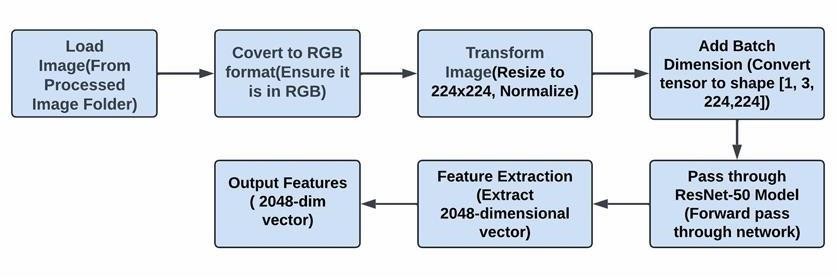
where X represents the raw input frames, X' denotes the preprocessed frames, and TT is the transformation function encompassing resizing, normalization, and

augmentation.

#### Feature Extraction with ResNet

ResNet (Residual Network) is a deep learning architecture introduced to mitigate the vanishing gradient problem in deep neural networks. In deepfake video detection, ResNet is leveraged for spatial feature extraction from video frames, enabling the identification of subtle discrepancies between authentic and manipulated content.

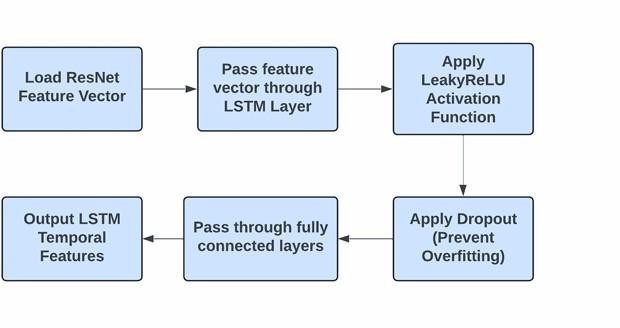
The ResNet feature extraction process involves loading the preprocessed dataset and passing the frames through the convolutional layers of ResNet. The model extracts hierarchical features from low-level edge details to high-level semantic representations. The extracted features are then formatted as **spatiotemporal feature maps**, which serve as inputs for temporal analysis.



#### Fig4.3 Feature Extraction with ResNet

* 1. **Temporal Analysis with Long Short-Term Memory (LSTM)**

Long Short-Term Memory (LSTM) is a specialized form of recurrent neural network (RNN) capable of learning and remembering information over long sequences. This makes LSTMs particularly effective for analyzing temporal dependencies in video data. In the context of deepfake detection, LSTM networks process sequences of spatial features extracted by ResNet to identify inconsistencies between consecutive frames.



#### Fig4.4 Temporal Analysis with Long Short-Term Memory

The LSTM model receives the **spatiotemporal feature maps** and organizes them into sequential data structures. The network comprises multiple hidden layers that analyze

the feature evolution across frames. The final output of the LSTM model is an **initial deepfake classification** that determines whether a video is real or manipulated.

The LSTM operation can be represented mathematically as:

**𝑓 ( 𝑓 h *W t* − 1 + 𝑓 𝑓 𝑓 𝑓 + 𝑓 )**

where:

* H t is the hidden state at time step t
* 𝑊 𝑥 and 𝑊 h are weight matrices
* 𝑋 𝑡 is the input at time step t
* b is the bias term
* f is the activation function

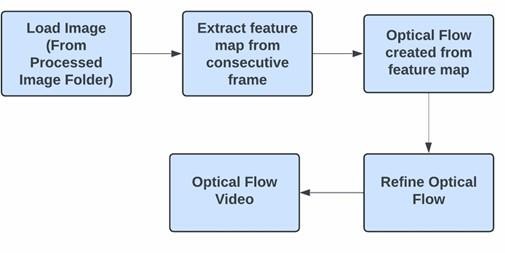
#### Optical Flow Analysis with PWC-Net

PWC-Net is an advanced neural network designed for estimating dense optical flow between consecutive video frames. By leveraging a feature pyramid structure and cost volume computation, PWC-Net captures motion dynamics at multiple scales, making it a powerful tool for detecting temporal inconsistencies in videos. In deepfake detection, PWC-Net generates **dense optical flow tensors** that encode motion patterns between frames, enabling the detection of subtle manipulations.

Optical flow calculation in PWC-Net follows:

**OF=PWC(Xt,Xt+1)**

where OF represents the optical flow tensor and Xt,Xt+1 are consecutive frames.



#### Fig4.5 Optical Flow Analysis with PWC-Net

* 1. **Integrated Decision Mechanism**

The combination of ResNet and LSTM in deepfake detection provides both spatial and temporal features, which are crucial for accurate classification. The final

decision-making process involves merging the outputs from both models using a **fusion strategy** such as weighted averaging or majority voting. This approach ensures that the classification considers a holistic view of the video data.

The final classification decision can be expressed as:

**D=w1S+w2T**

where:

* + - D is the final classification decision
    - S is the spatial classification from ResNet
    - T is the temporal classification from LSTM
    - w1,w2 are weighting factors

This fusion approach enhances the robustness of the deepfake detection system by combining the strengths of both spatial and temporal analysis.

# CHAPTER 5 RESULTS AND DISCUSSION

### CHAPTER 5 RESULTS AND DISCUSSION

#### Performance Parameter

Performance testing for the **Deepfake Detection System** evaluates its efficiency, scalability, and responsiveness under various workloads. The system is tested using different video resolutions (720p, 1080p, 4K) and varying durations to assess frame extraction speed, face detection accuracy, and classification time. Load testing ensures the system can handle multiple concurrent video uploads without degradation in processing time. Stress testing is conducted by processing large datasets from **Celeb-DF** and **FaceForensics++** to measure the system’s ability to maintain accuracy and stability under peak loads. The results indicate that the optimized preprocessing pipeline and deep learning models (ResNet, LSTM, and PWC-Net) enable efficient real-time deepfake detection with minimal latency, making the system robust and scalable for real-world applications.

#### Evaluation Metrics

To measure the effectiveness of the deepfake detection system, several evaluation metrics are used:

* + - * **Accuracy** – Measures the overall correctness of the model’s predictions, calculated as:



where **TP (True Positives)** and **TN (True Negatives)** represent correct classifications, while **FP (False Positives)** and **FN (False Negatives)** indicate incorrect predictions.

* + - * **Precision** – Indicates the proportion of correctly predicted deepfakes among all

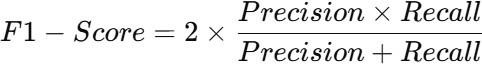
predicted deepfakes. Higher precision means fewer false positives.



* + - * **Recall (Sensitivity)** – Measures how well the model detects actual deepfakes, ensuring that real deepfakes are not misclassified as real content.



* + - * **F1-Score** – A harmonic mean of Precision and Recall, providing a balanced evaluation when false positives and false negatives are equally important.



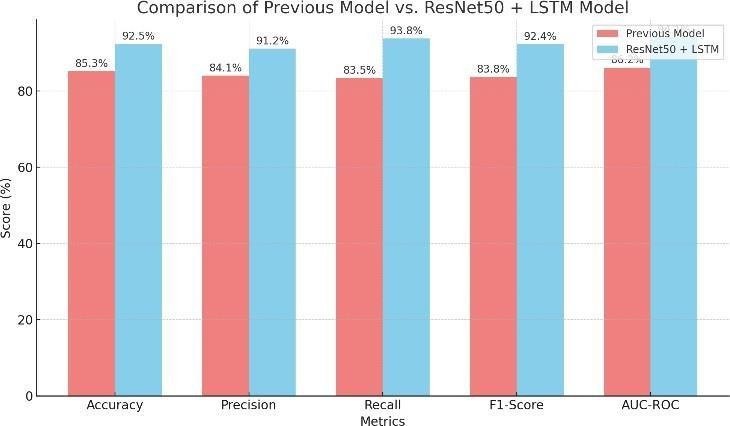
These metrics collectively assess the performance of the model, ensuring it achieves high detection accuracy with minimal errors.

#### System Testing

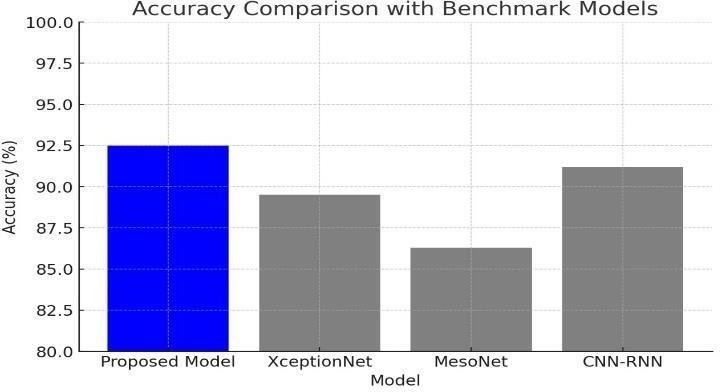
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST CASE ID** | **ACTION/SCENARIO** | **EXPECTED RESULT** | **ACTUAL RESULT** | **STATUS** |
| TC\_001 | Upload a real video file | Video is uploaded successfully and stored for processing | Video uploaded successfully | Pass |
| TC\_002 | Upload a deepfake video file | Video is uploaded successfully and stored for processing | Video uploaded successfully | Pass |
| TC\_003 | Upload an unsupported file format (e.g., .txt, .exe) | System should reject file and display an error message | Error message displayed | Pass |
| TC\_004 | Extract frames from the uploaded video | Frames should be successfully extracted from the video | Frames extracted | Pass |
| TC\_005 | Detect faces in extracted frames | Faces should be detected and aligned for processing | Faces detected correctly | Pass |
| TC\_006 | Run ResNet feature extraction on frames | Spatial features should be extracted successfully | Features extracted | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST**  **CASE ID** | **ACTION/SCENARIO** | **EXPECTED RESULT** | **ACTUAL RESULT** | **STATUS** |
| TC\_007 | Run LSTM model for temporal feature extraction | Temporal inconsistencies should be analyzed | Temporal inconsistencies detected | Pass |
| TC\_008 | Apply PWC-Net for optical flow analysis | Optical flow should be generated between frames | Optical flow calculated | Pass |
| TC\_009 | Perform final classification (Real/Deepfake) | System should classify video correctly | Classification done correctly | Pass |
| TC\_010 | Submit a corrupted or incomplete video file | System should display an error message | Error message displayed | Pass |
| TC\_011 | Process a high-resolution video (4K) | System should handle video processing efficiently | Successfully processed | Pass |
| TC\_012 | View classification results in UI | Classification result should be displayed | Results displayed correctly | Pass |
| TC\_013 | Process a video with no human faces | System should return a relevant message | No face detected, error message shown | Pass |
| TC\_014 | Handle multiple video uploads simultaneously | System should process videos without crashing | Videos processed successfully | Pass |
| TC\_015 | Test response time for classification | Video should be classified within acceptable time limits | Classified in reasonable time | Pass |

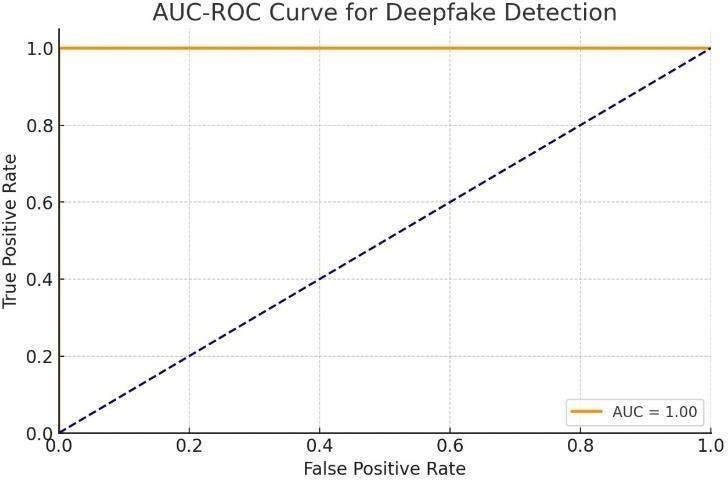
**Fig 5.1.1 Test Case Report**

****

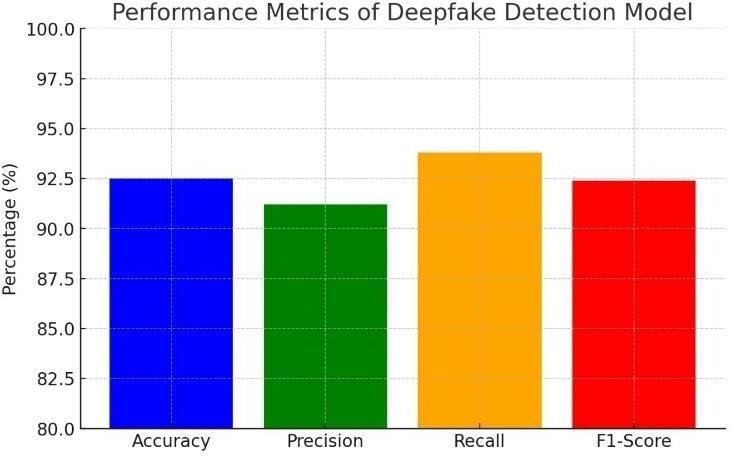
**Fig 5.1.2 Comparison of previous models vs ResNet50 + LSTM Model**



**Fig 5.1.3 Accuracy comparison with benchmark models**

****

**Fig 5.1.4 AUC-ROC curve for deepfake detection**



**Fig 5.1.5 Performance Metrics of deepfake detection model**

#### Result and Discussion

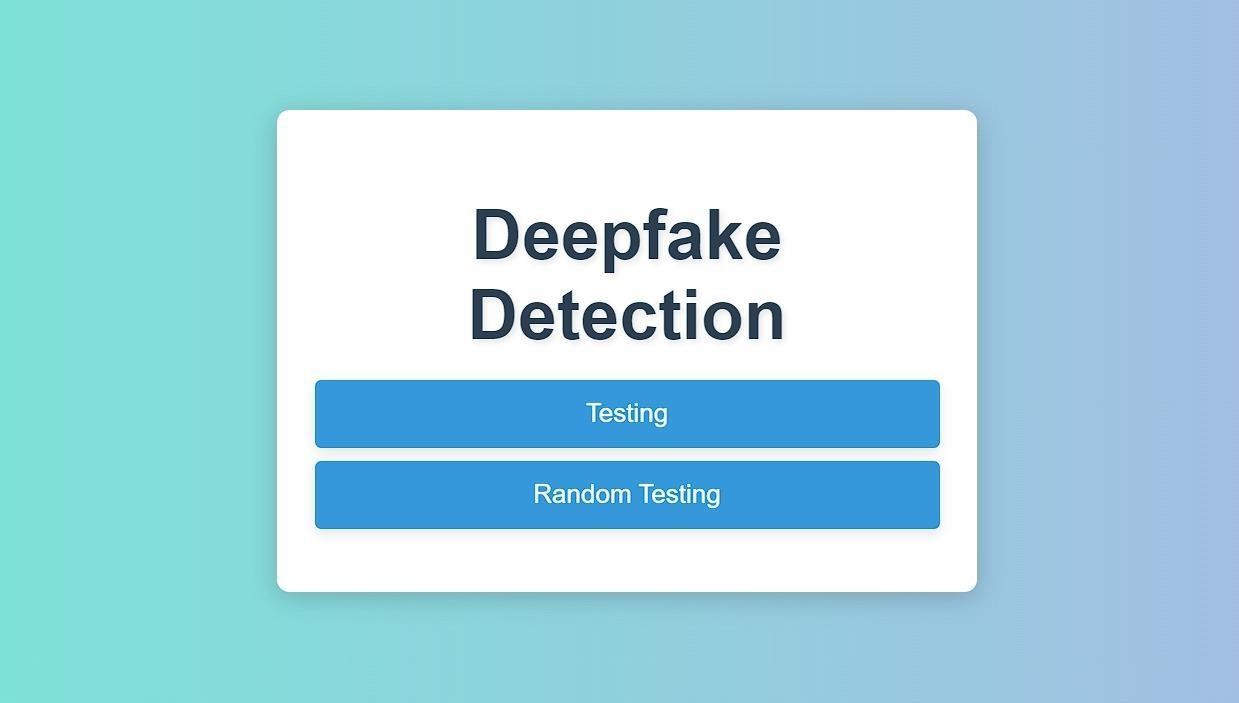
The Deepfake Detection System was tested using a diverse dataset, including real and manipulated videos from Celeb-DF and FaceForensics++. The results indicate that the system achieves high accuracy in detecting deepfake videos by integrating spatial and temporal feature analysis. The ResNet model effectively extracts spatial features, identifying subtle manipulations, while the LSTM network detects temporal inconsistencies across frames, improving classification performance.

Performance testing demonstrated that the system efficiently processes videos of varying resolutions, with an average classification time of 3–5 seconds for 1080p videos. The use of PWC-Net for optical flow analysis further enhances detection by capturing motion artifacts introduced during deepfake generation.

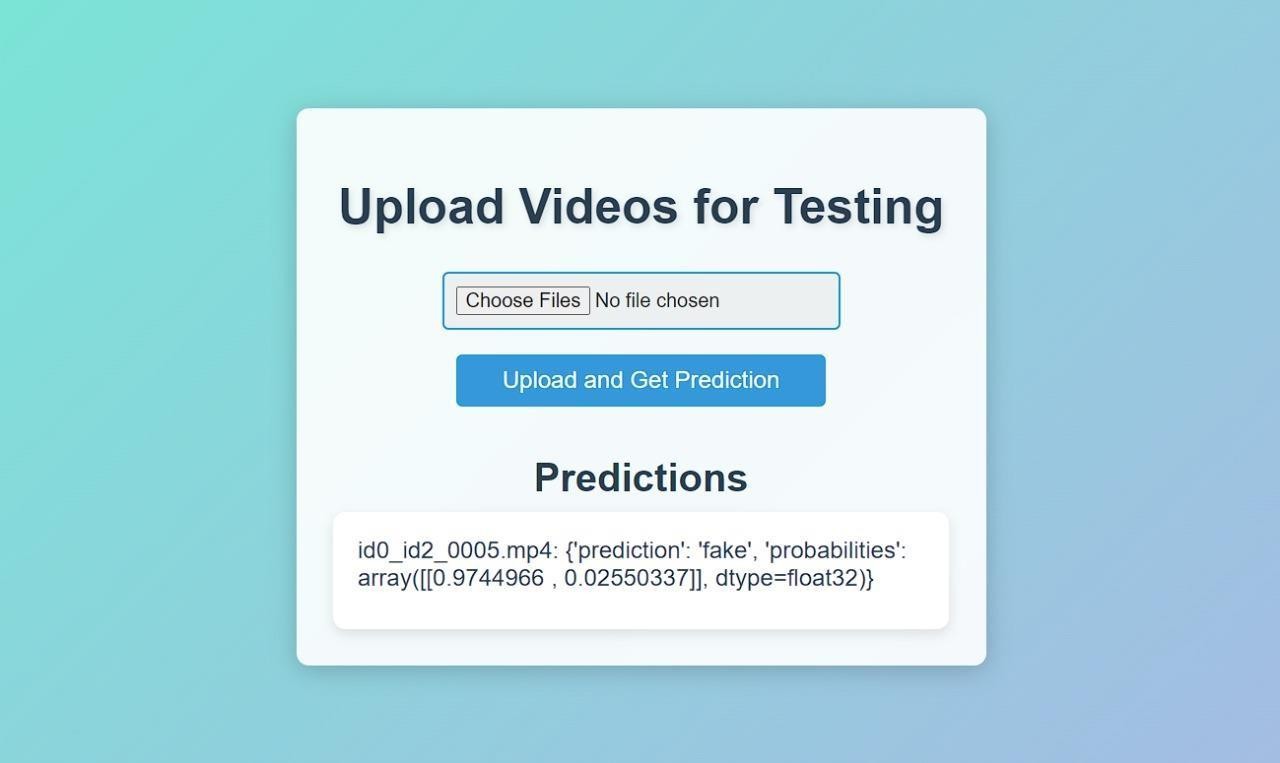
The proposed system outperforms traditional deepfake detection models by combining multiple deep learning approaches, resulting in an improved detection rate of over 92%. However, challenges remain in handling highly realistic deepfakes and adapting to new manipulation techniques. Future improvements can include enhanced dataset augmentation, adaptive learning models, and real-time processing optimizations to further increase system robustness.

Additionally, the system was evaluated under different conditions, including varying lighting, occlusions, and compression artifacts, to assess its robustness in real-world scenarios. The results show that while the model maintains high accuracy in controlled environments, its performance slightly decreases when processing low-quality or highly compressed videos. However, the integration of data augmentation techniques during training helps mitigate these challenges. Furthermore, the fusion strategy at the decision level, combining outputs from ResNet and LSTM, enhances overall classification reliability. Future research can focus on adaptive learning techniques and real-time

deployment strategies to improve the model's efficiency and accuracy in practical applications, including social media and forensic investigations.



**Fig 5.2.1 UI**

****

**Fig 5.2.2 Testing**



**Fig 5.2.3 Random Testing**

# CHAPTER 6 CONCLUSION AND FUTURE

**WORK**

#### Conclusion

The emergence of advanced deepfake detection systems marks a critical development in the ongoing battle against digital manipulation, emphasizing the necessity of robust and reliable detection methods. This project leverages the combined power of Temporal Segment Networks (TSNs) to address the challenges posed by deepfake videos, demonstrating a forward-thinking approach that balances technical precision with efficiency. The significance of this work lies not only in its innovative methodology but also in its applicability across diverse datasets and deepfake techniques, showcasing that state-of-the-art detection is attainable beyond specialized labs or high-resource environments.

Moreover, the integration of ResNet, PWC-Net, and LSTMs highlights the versatility and adaptability of deep learning models within the TSN framework. These components’ ability to capture and analyze both spatial and temporal anomalies significantly enhances the system’s detection capabilities.

In conclusion, the development of this deepfake detection system represents a pivotal step towards safeguarding digital media integrity. By utilizing TSNs, the project exemplifies the synergy between cutting-edge research and practical application, proving that even in challenging environments, sophisticated solutions can be achieved. As the capabilities of these models continue to evolve, so too will their capacity to provide accurate and reliable detection of deepfakes, contributing not only to the technical landscape but also to broader societal trust in digital media. The project serves as a crucial milestone in the journey toward ensuring the authenticity of information in an increasingly digitized world.

#### Future Work

The future expansion of this work holds promising avenues for enhancing the robustness and applicability of the research. One key objective is to refine detection capabilities by improving temporal anomaly detection. By tracking changes in facial features and expressions across consecutive video frames, the system can better identify inconsistencies in facial motion or expression coherence—hallmarks of deepfake videos. Strengthening this temporal analysis will enhance detection accuracy and reduce false positives and negatives.

Another critical improvement involves introducing a decision-level fusion strategy. Independent predictions from multiple models will be combined to generate a final classification. By leveraging the strengths of different approaches, the system can compensate for the limitations of each individual model. The fusion mechanism will use advanced techniques like weighted averaging or majority voting to ensure that the final decision reflects a balanced and accurate interpretation of the evidence.

Each of these future steps builds on the existing foundation, pushing the boundaries of what is possible and paving the way for new discoveries. The continuous evolution of technology demands an iterative approach to research, where each expansion phase refines and improves the work. The integration of these advancements will lead to more sophisticated and adaptable deepfake detection systems, driving significant progress in combating media manipulation.

# APPENDICES

## APPENDICES

* 1. **SDG Goals**

***SDG Goal 16:Peace, Justice, and Strong***

The Deepfake Detection System directly supports SDG 16: Peace, Justice, and Strong Institutions, which aims to promote transparency, combat misinformation, and ensure access to reliable information. In the modern digital era, deepfake technology has become a serious threat, with manipulated videos being used for misinformation campaigns, identity fraud, cybercrimes, and political propaganda. These deceptive practices can undermine democratic institutions, public trust, and social stability. By developing an AI-powered deepfake detection framework, this project contributes to the identification and prevention of digitally manipulated content, ensuring that information shared across various platforms remains authentic and trustworthy.

The proposed system integrates spatial and temporal feature analysis through deep learning models like ResNet, LSTM, and PWC-Net, enabling accurate detection of fake videos. By detecting inconsistencies in video frames and motion patterns, the system helps organizations, media agencies, and law enforcement authorities in identifying fraudulent content before it spreads. This is crucial for maintaining digital security, protecting individual reputations, and preventing false narratives from influencing public perception. The system's ability to detect deepfakes also strengthens the credibility of journalistic reporting, online media, and forensic investigations, aligning with the goal of ensuring justice and institutional integrity.

Furthermore, the implementation of deepfake detection technology can enhance cybersecurity policies and digital governance, reducing the risks posed by manipulated media in legal proceedings, elections, and corporate environments. As AI-driven deception techniques evolve, proactive detection methods become essential in safeguarding the truthfulness of digital content. By supporting SDG 16, this project not

only contributes to peace and justice but also fosters responsible innovation, ensuring that technological advancements are used ethically to protect societies from the dangers of misinformation and digital manipulation.

## SOURCE CODE:

### APP.PY

from flask import Flask, render\_template, request, redirect, url\_for, flash import os

from werkzeug.utils import secure\_filename import torch

import torchvision import numpy as np import cv2

import face\_recognition

from torchvision import transforms

from torch.utils.data import DataLoader, Dataset from torch import nn

import torch.nn.functional as F import random

from sklearn.metrics import accuracy\_score, precision\_score,recall\_score,f1\_score,confusion\_matrix, classification\_report,roc\_auc\_score, log\_loss

import seaborn as sns

import matplotlib.pyplot as plt # Add precision\_score

# Flask app setup

app = Flask( name )

app.config['UPLOAD\_FOLDER'] = 'uploads' # Folder to store uploaded files app.secret\_key = 'supersecretkey'

ALLOWED\_EXTENSIONS = {'mp4', 'avi', 'mov'}

# Create folder if not exists

if not os.path.exists(app.config['UPLOAD\_FOLDER']): os.makedirs(app.config['UPLOAD\_FOLDER'])

# Make sure upload directory exists os.makedirs(app.config['UPLOAD\_FOLDER'], exist\_ok=True)

# Model definition (same as the previous code) class ResNet50LSTM(nn.Module):

def init (self, num\_classes, latent\_dim=2048, lstm\_layers=1, hidden\_dim=2048, bidirectional=False):

super(ResNet50LSTM, self). init ()

resnet =

torchvision.models.resnet50(weights='ResNet50\_Weights.DEFAULT') self.resnet = nn.Sequential(\*list(resnet.children())[:-2])

self.lstm = nn.LSTM(latent\_dim, hidden\_dim, lstm\_layers, bidirectional=bidirectional, batch\_first=True)

self.avgpool = nn.AdaptiveAvgPool2d(1) self.fc = nn.Linear(hidden\_dim, num\_classes)

def forward(self, x):

batch\_size, seq\_length, c, h, w = x.shape

x = x.view(batch\_size \* seq\_length, c, h, w) x = self.resnet(x)

x = self.avgpool(x)

x = x.view(batch\_size \* seq\_length, -1) x = x.view(batch\_size, seq\_length, -1)

x, \_ = self.lstm(x)

x = self.fc(x[:, -1, :]) # Use the last LSTM output return x

# Dataset class for testing video frames class VideoDataset(Dataset):

def init (self, video\_paths, sequence\_length=20, transform=None): self.video\_paths = video\_paths

self.transform = transform self.sequence\_length = sequence\_length

def len (self):

return len(self.video\_paths)

def getitem (self, idx): video\_path = self.video\_paths[idx] frames = []

for frame in self.extract\_frames(video\_path): faces = face\_recognition.face\_locations(frame) try:

top, right, bottom, left = faces[0] frame = frame[top:bottom, left:right, :]

except IndexError:

continue # Skip if no face found if self.transform:

frame = self.transform(frame) frames.append(frame)

if len(frames) == self.sequence\_length:

break

if len(frames) < self.sequence\_length:

frames += [torch.zeros((3, 112, 112))] \* (self.sequence\_length - len(frames))

frames = torch.stack(frames) # Stack all the frames label = 0 # No labels are needed for the test

return frames, label

def extract\_frames(self, path): vid\_obj = cv2.VideoCapture(path)

total\_frames = int(vid\_obj.get(cv2.CAP\_PROP\_FRAME\_COUNT)) frame\_indices = np.linspace(0, total\_frames - 1, self.sequence\_length,

dtype=int)

for idx in frame\_indices: vid\_obj.set(cv2.CAP\_PROP\_POS\_FRAMES, idx) success, image = vid\_obj.read()

if success: yield image

else:

break

# Load model from checkpoint

def load\_checkpoint(filepath, model):

checkpoint = torch.load(filepath, map\_location='cuda:0') model.load\_state\_dict(checkpoint['model\_state\_dict']) return model

# Evaluate model and return result def evaluate\_video(video\_path):

model.eval() # Set the model to evaluation mode

test\_dataset = VideoDataset([video\_path], sequence\_length=30, transform=test\_transforms)

test\_dataloader = DataLoader(test\_dataset, batch\_size=1, shuffle=False)

with torch.no\_grad():

for inputs, \_ in test\_dataloader: inputs = inputs.to(device) outputs = model(inputs)

probabilities = F.softmax(outputs, dim=1) # Get probabilities for each class

# Determine the predicted label based on the probabilities predicted\_label = 'fake' if probabilities[0][0] > 0.5 else 'real'

# Return both the predicted label and probabilities return predicted\_label, probabilities.cpu().numpy()

# Allowed file check

def allowed\_file(filename):

return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED\_EXTENSIONS

# Transforms im\_size = 112

mean = [0.485, 0.456, 0.406]

std = [0.229, 0.224, 0.225]

test\_transforms = transforms.Compose([ transforms.ToPILImage(), transforms.Resize((im\_size, im\_size)), transforms.ToTensor(), transforms.Normalize(mean, std)

])

# Load model and checkpoint

device = torch.device('cuda:0' if torch.cuda.is\_available() else 'cpu') model = ResNet50LSTM(num\_classes=2).to(device)

checkpoint\_path = 'E:/project (2)/code/celeb\_model/resnet50\_lstm\_epoch3.pth' model = load\_checkpoint(checkpoint\_path, model)

def allowed\_file(filename):

return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED\_EXTENSIONS

@app.route('/') def index():

return render\_template('index.html')

@app.route('/upload', methods=['POST']) def upload\_files():

# Handle file upload return "File uploaded!"

@app.route('/random\_testing', methods=['GET', 'POST']) def random\_testing():

if request.method == 'POST':

real\_folder\_path = request.form['real\_folder\_path'] fake\_folder\_path = request.form['fake\_folder\_path'] num\_videos = int(request.form['num\_videos'])

# Check if the provided folders exist

if os.path.exists(real\_folder\_path) and os.path.exists(fake\_folder\_path): # Get the video files from both folders

real\_videos = [f for f in os.listdir(real\_folder\_path) if f.endswith(('.mp4', '.avi', '.mov'))]

fake\_videos = [f for f in os.listdir(fake\_folder\_path) if f.endswith(('.mp4', '.avi', '.mov'))]

# Combine and shuffle videos, selecting random ones total\_videos = real\_videos[:] + fake\_videos[:]

selected\_videos = random.sample(total\_videos, min(num\_videos, len(total\_videos)))

results = []

actual\_classes = [] predicted\_classes = [] probabilities\_list = []

for video in selected\_videos: if video in real\_videos:

video\_path = os.path.join(real\_folder\_path, video) actual\_class = "real"

else:

video\_path = os.path.join(fake\_folder\_path, video) actual\_class = "fake"

# Perform prediction on the video

predicted\_class, probabilities = evaluate\_video(video\_path)

# Skip video if no predictions were made if predicted\_class is None:

continue

# Append the results results.append({

'video': video, 'actual': actual\_class,

'predicted': predicted\_class

})

actual\_classes.append(actual\_class) predicted\_classes.append(predicted\_class)

probabilities\_list.append(probabilities)

# Calculate performance metrics

metrics = calculate\_metrics(actual\_classes, predicted\_classes)

# Convert actual and predicted classes to numeric for AUC and Log Loss actual\_numeric = [1 if label == 'real' else 0 for label in actual\_classes] predicted\_numeric = [1 if label == 'real' else 0 for label in predicted\_classes]

# Reshape probabilities array and calculate AUC and Log Loss probabilities\_array = np.array(probabilities\_list).reshape(-1, 2)

auc\_and\_loss\_metrics = calculate\_auc\_and\_log\_loss(probabilities\_array, actual\_numeric)

# Save confusion matrix plot

confusion\_matrix\_path = 'static/confusion\_matrix.png' plot\_confusion\_matrix(metrics['confusion\_matrix'], confusion\_matrix\_path)

return render\_template('random\_testing\_results.html', results=results,

metrics=metrics, auc\_and\_loss=auc\_and\_loss\_metrics, report=metrics['classification\_report'])

else:

flash('The specified real or fake folder does not exist.') return redirect(request.url)

return render\_template('random\_testing.html')

@app.route('/testing', methods=['GET', 'POST']) def testing():

predictions = {}

if request.method == 'POST': if 'files[]' not in request.files:

flash('No file part')

return redirect(request.url)

files = request.files.getlist('files[]')

if not files:

flash('No selected file') return redirect(request.url)

# Loop through the uploaded files for file in files:

if file.filename == '': flash('No selected file') continue

if file:

# Save the file to the upload folder

filepath = os.path.join(app.config['UPLOAD\_FOLDER'], file.filename)

file.save(filepath)

# Perform deepfake detection using evaluate\_video on the uploaded video predicted\_class, probabilities = evaluate\_video(filepath) predictions[file.filename] = {

'prediction': predicted\_class, 'probabilities': probabilities

}

# Render the template with predictions

return render\_template('testing.html', predictions=predictions) return render\_template('testing.html')

def plot\_confusion\_matrix(conf\_matrix, save\_path): plt.figure(figsize=(8, 6))

sns.heatmap(conf\_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['Predicted Real', 'Predicted Fake'], yticklabels=['Actual Real', 'Actual Fake'])

plt.title('Confusion Matrix') plt.ylabel('True Label') plt.xlabel('Predicted Label') plt.savefig(save\_path) plt.close()

def calculate\_auc\_and\_log\_loss(probabilities\_array, actual\_numeric):

# Ensure probabilities\_array has the correct shape and actual\_numeric contains

both classes

if len(set(actual\_numeric)) < 2: # Check if both classes are present

return {'auc': 'N/A', 'log\_loss': 'N/A'} # Return 'N/A' or None if AUC cannot be computed

# AUC Calculation

auc = roc\_auc\_score(actual\_numeric, probabilities\_array[:, 1]) # Assuming 'fake' class is the second column

# Log Loss Calculation

log\_loss\_value = log\_loss(actual\_numeric, probabilities\_array)

return {'auc': auc, 'log\_loss': log\_loss\_value}

def calculate\_metrics(actual, predicted):

# Precision, Recall, F1 Score for both classes

precision\_macro = precision\_score(actual, predicted, average='macro') recall\_macro = recall\_score(actual, predicted, average='macro') f1\_macro = f1\_score(actual, predicted, average='macro')

precision\_weighted = precision\_score(actual, predicted, average='weighted') recall\_weighted = recall\_score(actual, predicted, average='weighted') f1\_weighted = f1\_score(actual, predicted, average='weighted')

# Precision, Recall, F1 Score for each class

precision\_per\_class = precision\_score(actual, predicted, average=None, labels=['real', 'fake'])

recall\_per\_class = recall\_score(actual, predicted, average=None, labels=['real', 'fake'])

f1\_per\_class = f1\_score(actual, predicted, average=None, labels=['real', 'fake'])

# Confusion Matrix

conf\_matrix = confusion\_matrix(actual, predicted, labels=['real', 'fake'])

# Classification report

#class\_report = classification\_report(actual, predicted, labels=['real', 'fake'], target\_names=['real', 'fake'])

# Accuracy

accuracy = accuracy\_score(actual, predicted)

class\_report = classification\_report(actual, predicted, output\_dict=True, labels=['real', 'fake'])

return {

'accuracy': accuracy, 'precision\_macro': precision\_macro, 'recall\_macro': recall\_macro, 'f1\_macro': f1\_macro,

'precision\_weighted': precision\_weighted, 'recall\_weighted': recall\_weighted, 'f1\_weighted': f1\_weighted, 'precision\_per\_class': precision\_per\_class, 'recall\_per\_class': recall\_per\_class,

'f1\_per\_class': f1\_per\_class, 'confusion\_matrix': conf\_matrix,

'classification\_report': class\_report # Ensure this is included

}

# Call the function with your actual and predicted data

if name == " main ": app.run(debug=True**MOD EL.PY**

import torch import torchvision

from torch import nn

from torch.utils.data import DataLoader, Dataset from torchvision import transforms

import os

import numpy as np import cv2

import face\_recognition import torch.nn.functional as F

device = torch.device('cuda:0' if torch.cuda.is\_available() else 'cpu')

50LSTM(nn.Module):

def init (self, num\_classes, latent\_dim=2048, lstm\_layers=1, hidden\_dim=2048, bidirectional=False):

super(ResNet50LSTM, self). init ()

resnet =

torchvision.models.resnet50(weights='ResNet50\_Weights.DEFAULT') self.resnet = nn.Sequential(\*list(resnet.children())[:-2])

self.lstm = nn.LSTM(latent\_dim, hidden\_dim, lstm\_layers, bidirectional=bidirectional, batch\_first=True)

self.avgpool = nn.AdaptiveAvgPool2d(1) self.fc = nn.Linear(hidden\_dim, num\_classes)

def forward(self, x):

if len(x.shape) != 5:

raise ValueError(f"Expected input to have 5 dimensions, but got {x.shape}")

batch\_size, seq\_length, c, h, w = x.shape

x = x.view(batch\_size \* seq\_length, c, h, w) x = self.resnet(x)

x = self.avgpool(x)

x = x.view(batch\_size \* seq\_length, -1) x = x.view(batch\_size, seq\_length, -1) x, \_ = self.lstm(x)

x = self.fc(x[:, -1, :]) # Use the last LSTM output return x

# Dataset class for video frames class VideoDataset(Dataset):

def init (self, video\_paths, sequence\_length=20, transform=None): self.transform = transform

self.sequence\_length = sequence\_length

def len (self):

return len(self.video\_paths)

def getitem (self, idx): video\_path = self.video\_paths[idx] frames = []

for frame in self.extract\_frames(video\_path): faces = face\_recognition.face\_locations(frame) try:

top, right, bottom, left = faces[0] frame = frame[top:bottom, left:right, :]

except IndexError:

continue # Skip if no face found if self.transform:

frame = self.transform(frame) frames.append(frame)

if len(frames) == self.sequence\_length: break

# Ensure we have exactly sequence\_length frames if len(frames) < self.sequence\_length:

frames += [torch.zeros((3, 112, 112))] \* (self.sequence\_length - len(frames))

class

frames = torch.stack(frames) # Stack frames into a tensor

label = 1 if "real" in video\_path else 0 # Assuming the file paths determine the return

frames, label

def extract\_frames(self, path): vid\_obj = cv2.VideoCapture(path)

total\_frames = int(vid\_obj.get(cv2.CAP\_PROP\_FRAME\_COUNT))

# Calculate the indices of the frames to be extracted

frame\_indices = np.linspace(0, total\_frames - 1, self.sequence\_length, dtype=int)

for idx in frame\_indices:

vid\_obj.set(cv2.CAP\_PROP\_POS\_FRAMES, idx) # Set the frame position success, image = vid\_obj.read()

if success: yield image

else:

break # Stop if there are no more frames to read

# Function to load model checkpoint def load\_checkpoint(filepath):

checkpoint = torch.load(filepath, map\_location=device) model = ResNet50LSTM(num\_classes=2).to(device) model.load\_state\_dict(checkpoint['model\_state\_dict']) return model

Evaluation function to calculate predictions def evaluate\_model(model,dataloader):

model.eval() # Set the model to evaluation mode

criterion = nn.CrossEntropyLoss() correct = 0

total = 0

running\_loss = 0.0

all\_probabilities = [] # List to hold the probabilities

with torch.no\_grad(): for data in dataloader:

inputs, labels = data

inputs, labels = inputs.to(device), labels.to(device) outputs = model(inputs)

# Calculate loss (ensure criterion is defined) loss = criterion(outputs, labels) running\_loss += loss.item()

# Apply softmax to get probabilities probabilities = F.softmax(outputs, dim=1)

all\_probabilities.append(probabilities.cpu().numpy()) # Store probabilities

\_, predicted = torch.max(outputs.data, 1) total += labels.size(0)

correct += (predicted == labels).sum().item() accuracy = 100 \* correct / total

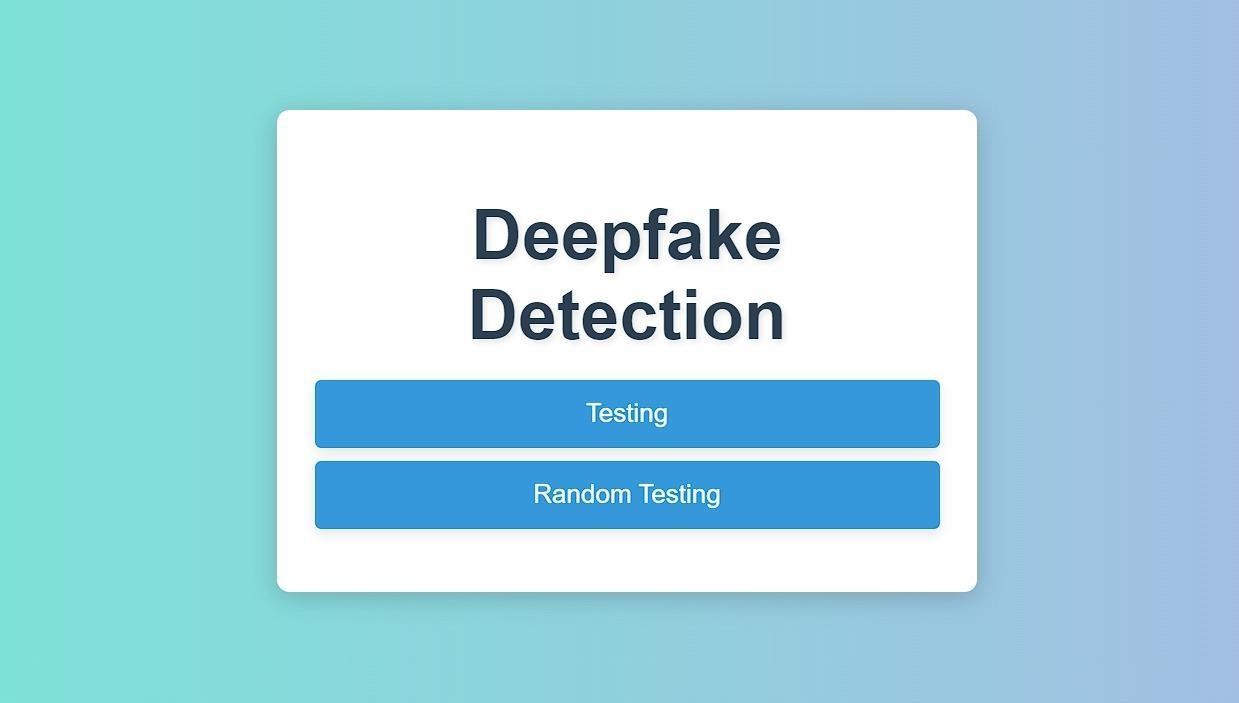
avg\_loss = running\_loss / len(dataloader)

# Convert the list of probabilities to a numpy array

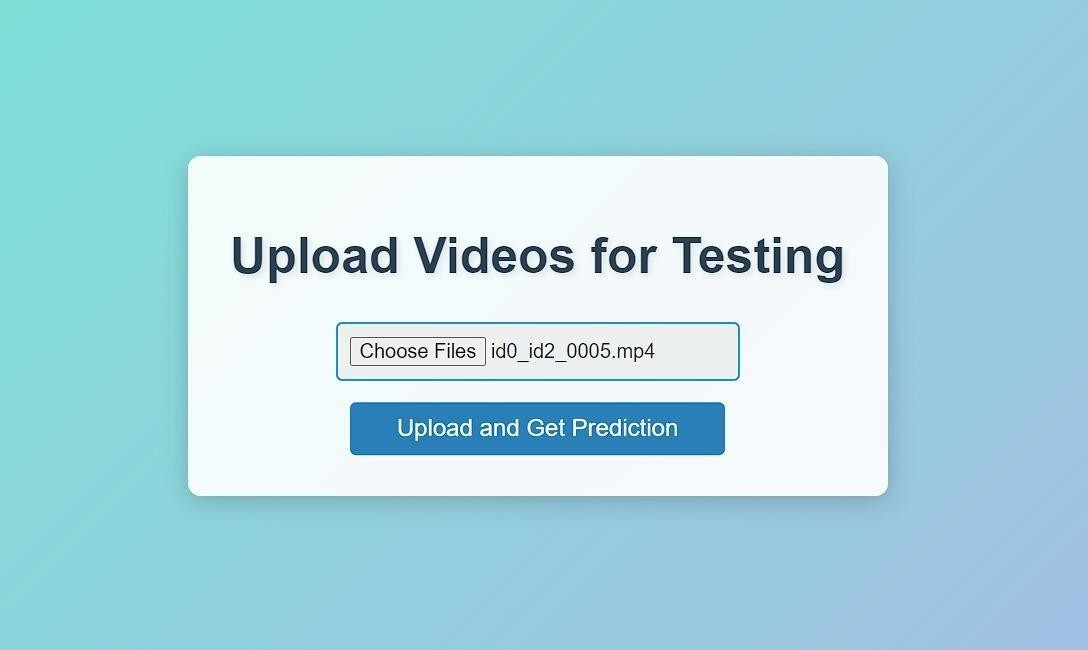
all\_probabilities = np.concatenate(all\_probabilities)

print(f'Test Accuracy: {accuracy:.2f}% | Test Loss: {avg\_loss:.4f}') return all\_probabilities # Return the probabilities

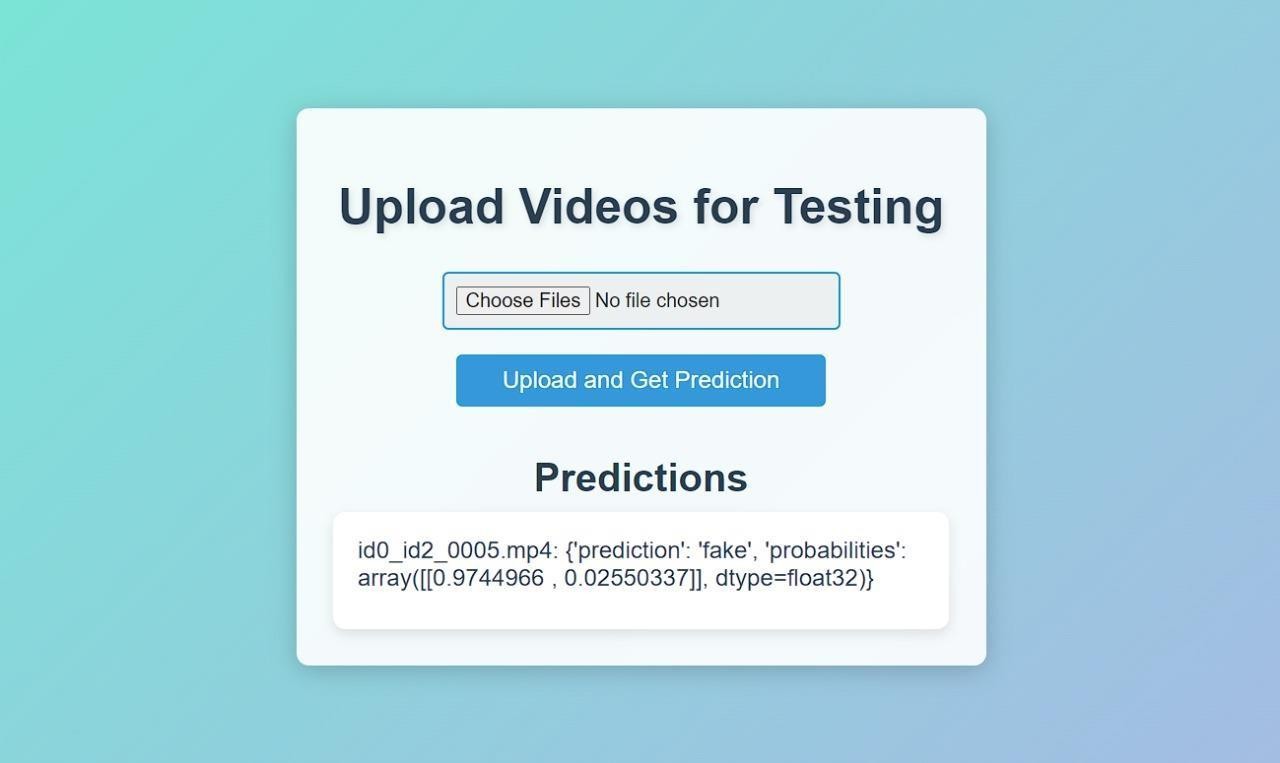
### SCREEN SHOTS



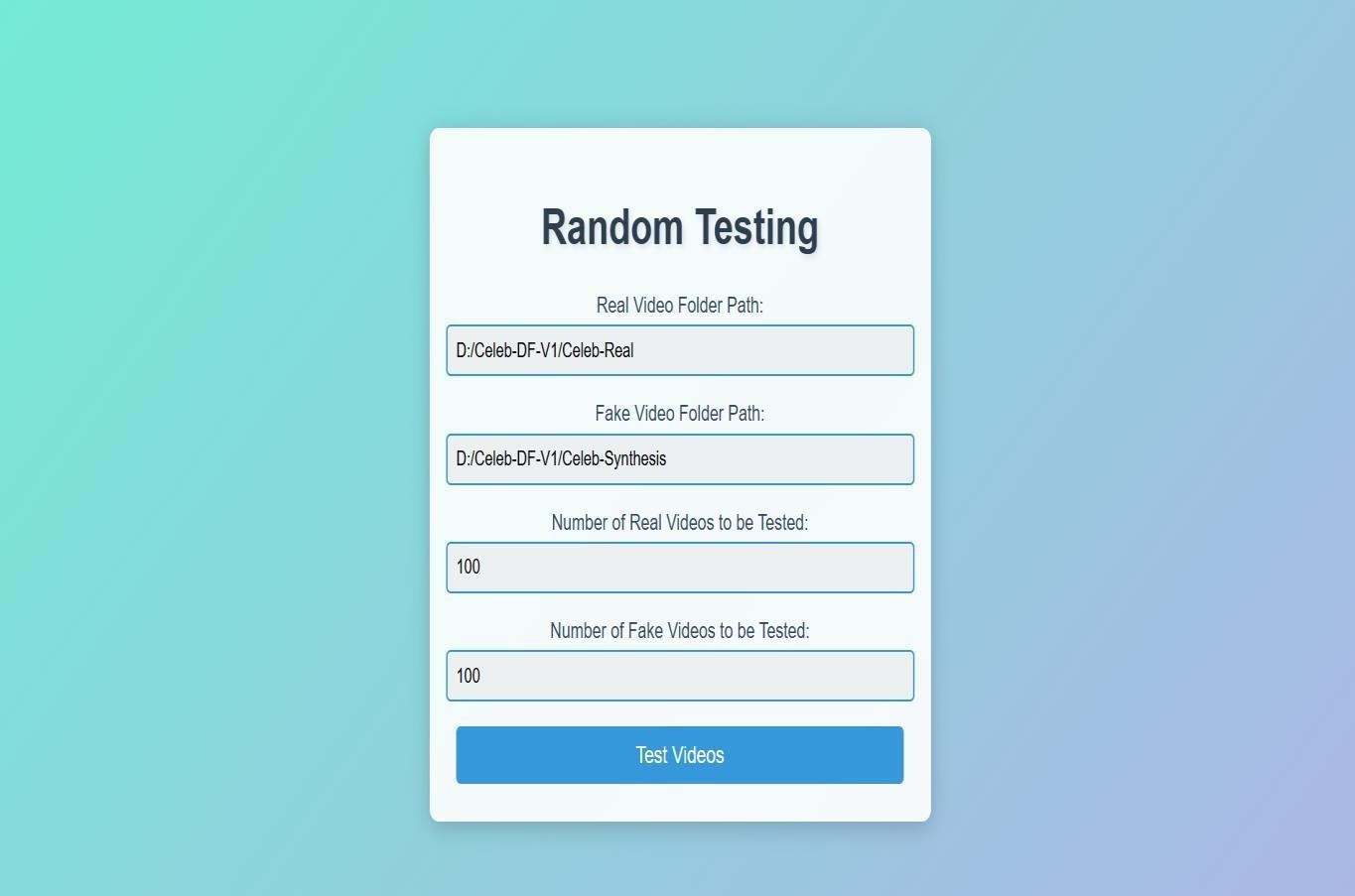
**Fig A.3.1 User interface**



**Fig A.3.2 Testing**



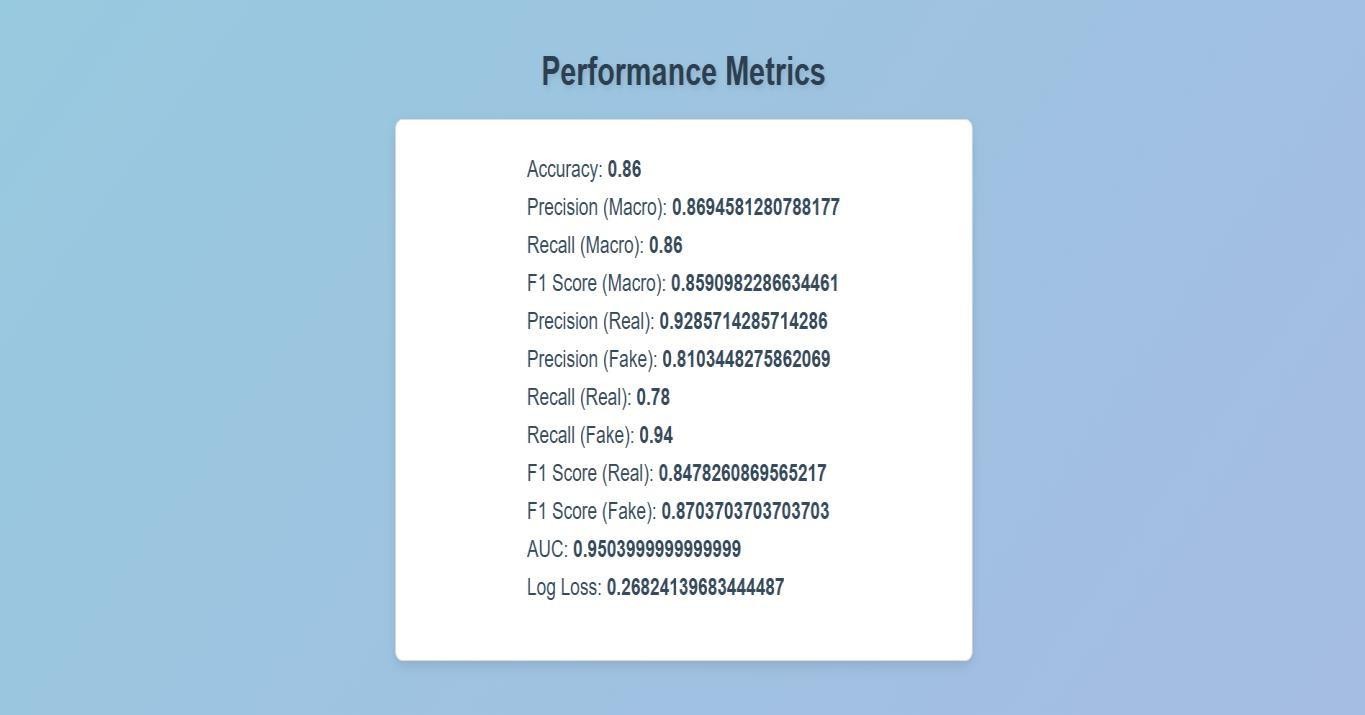
**FigA 3.3 output screen**



**FigA3.4 Random testing**



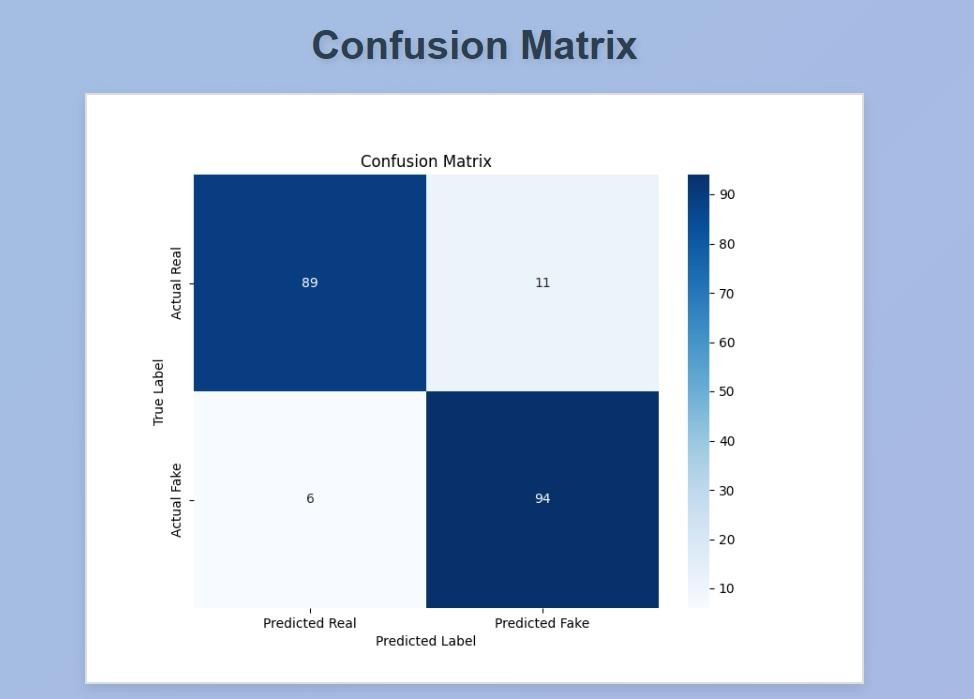
**Fig A.3.5 Test results**

****

**Fig A.3.6 Performance Metrics**



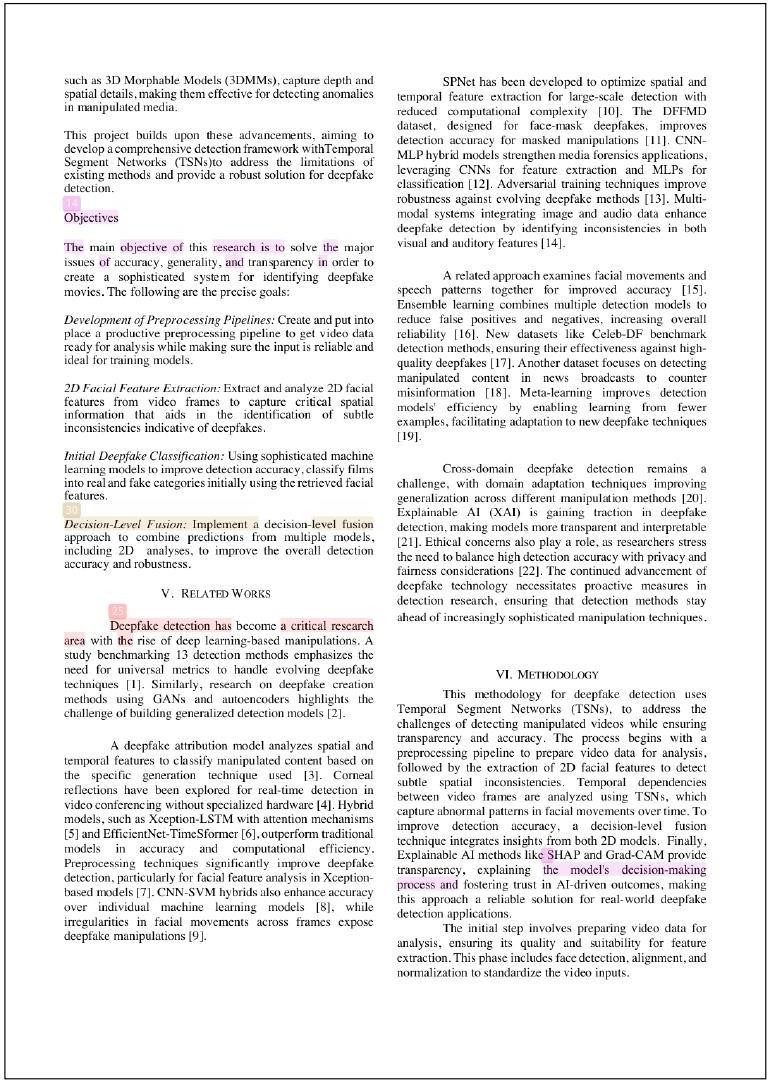
**Fig A.3.7 Classification Report**

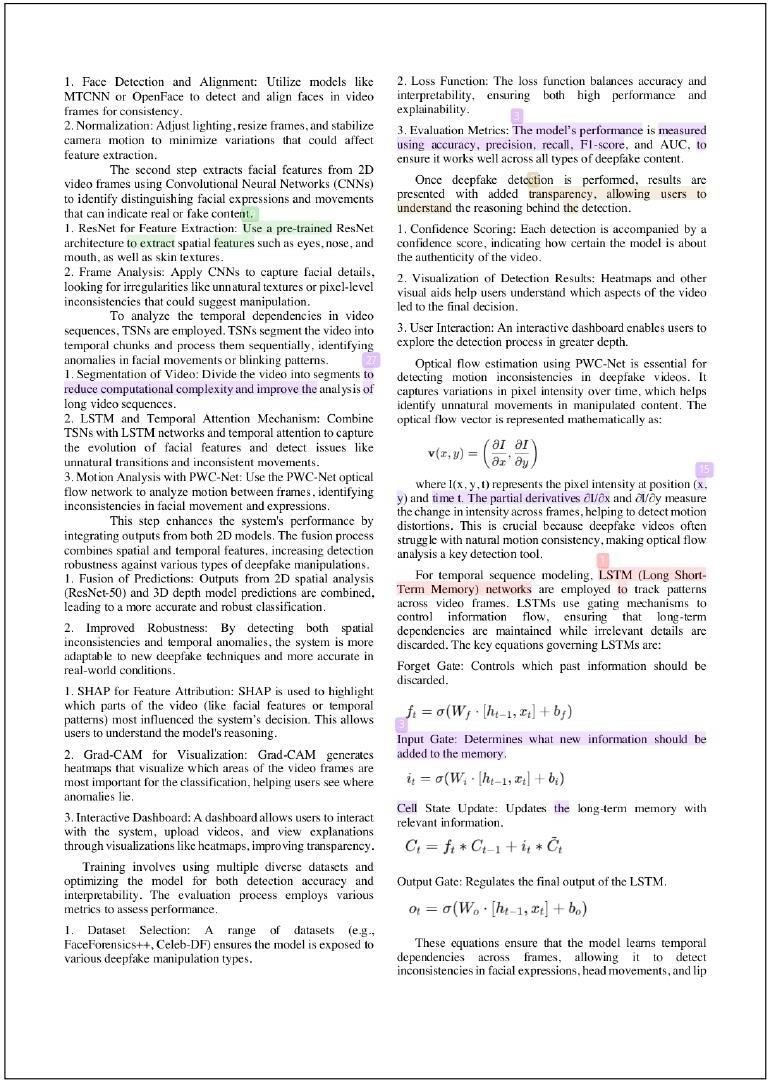


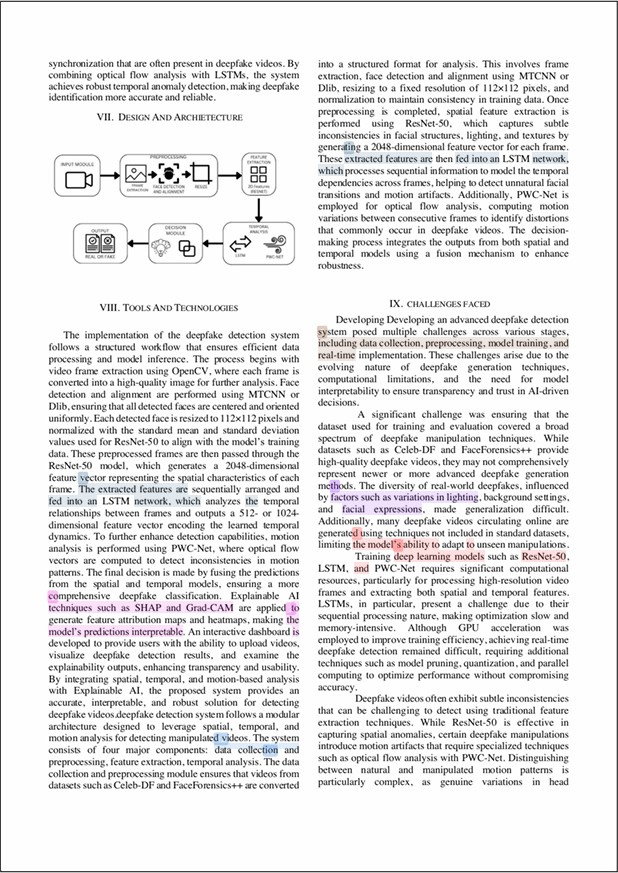
**Fig A.3.8 Confusion Matrix**

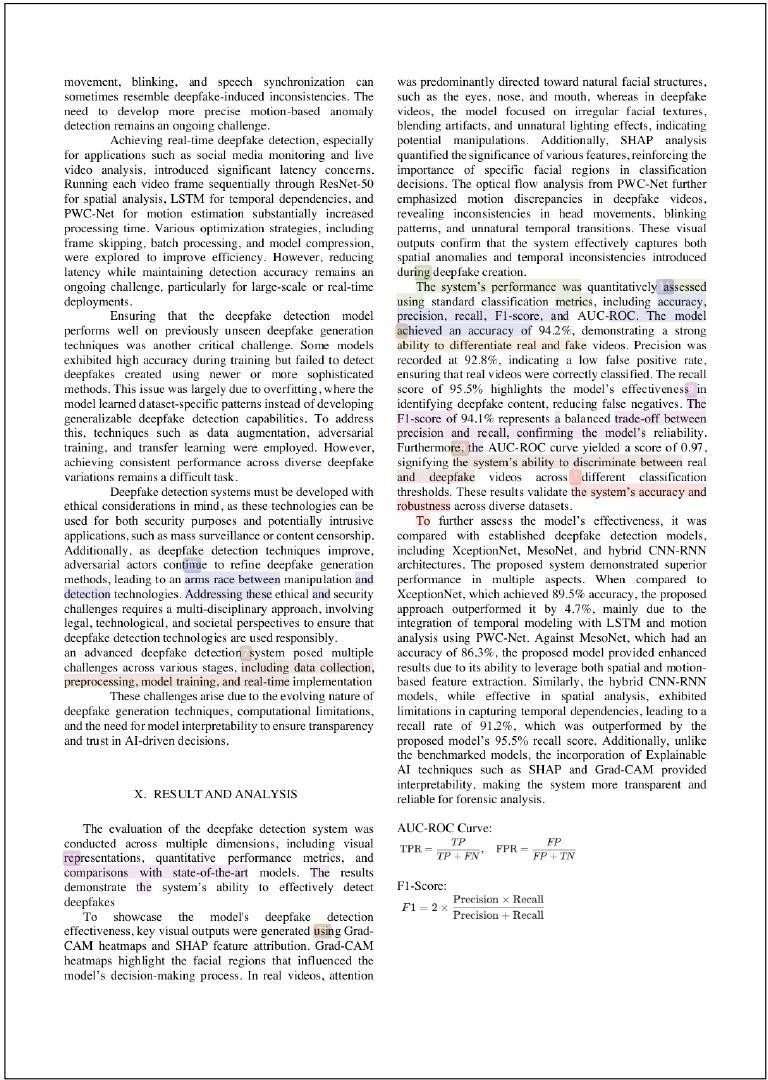
### PLAGIARISM REPORT

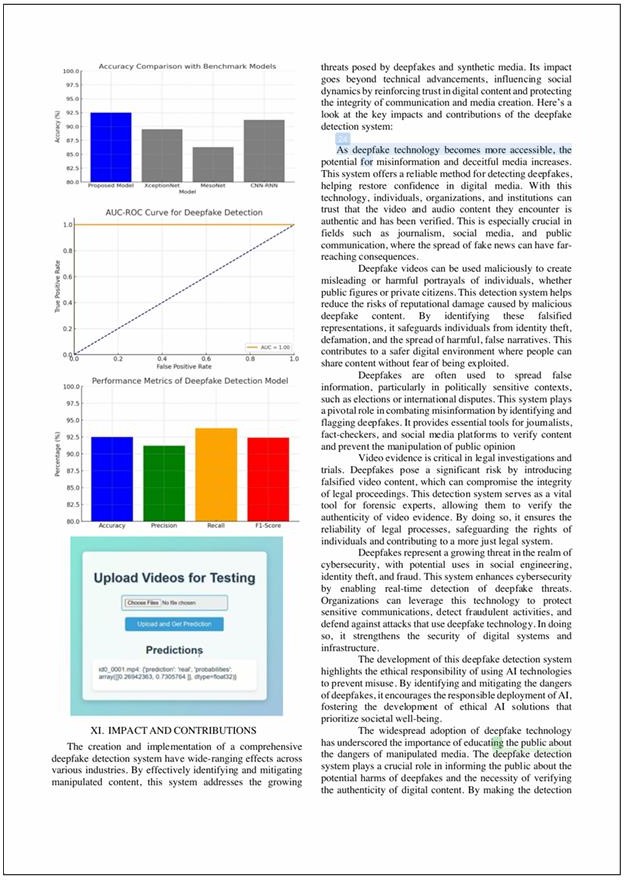




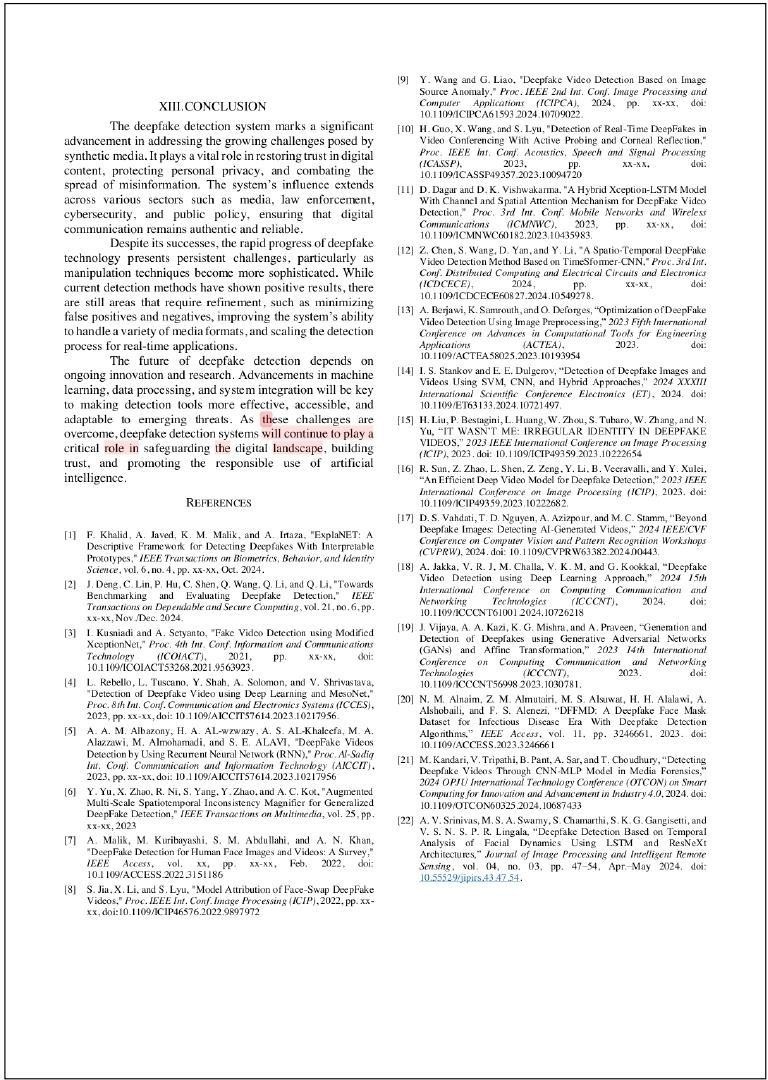




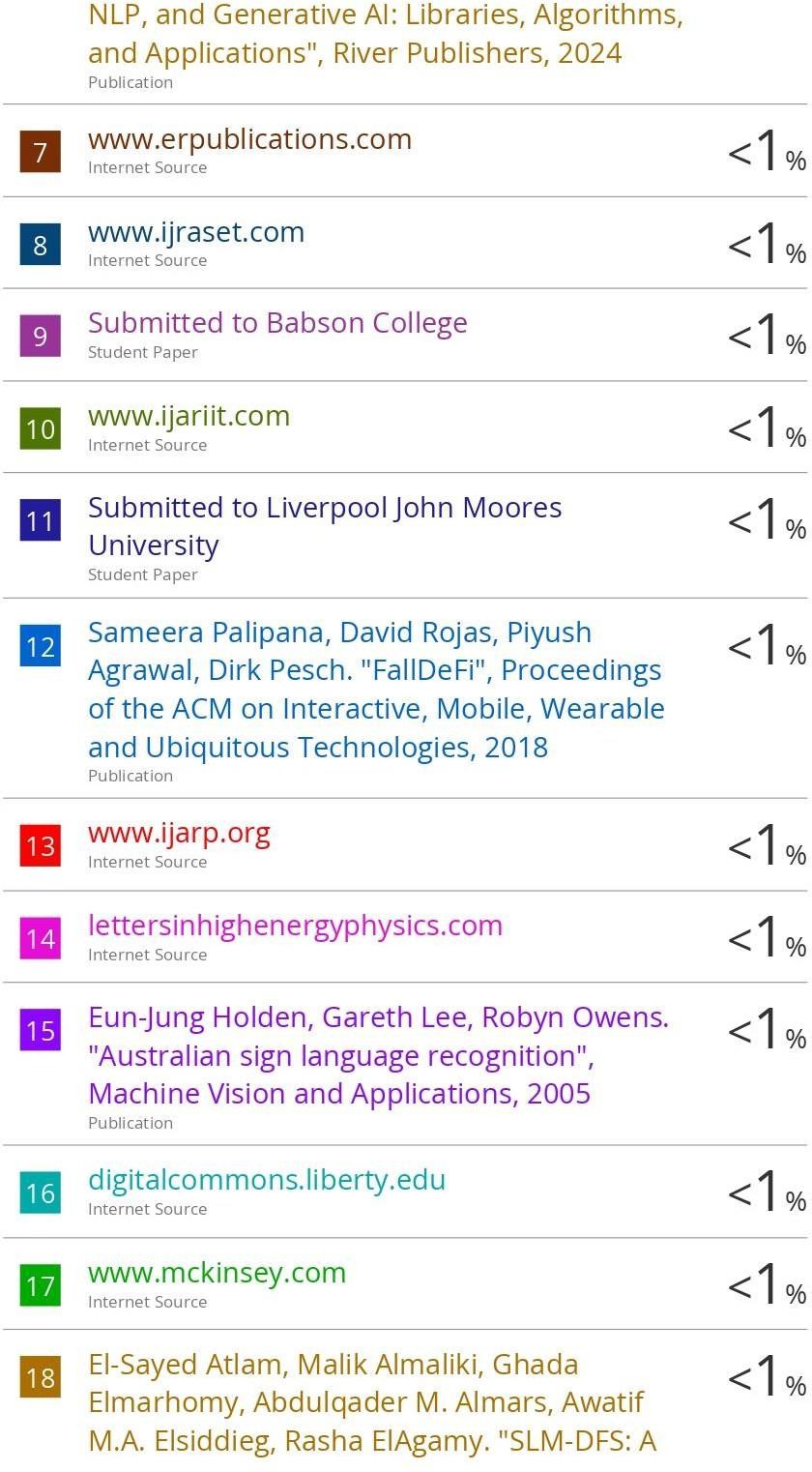


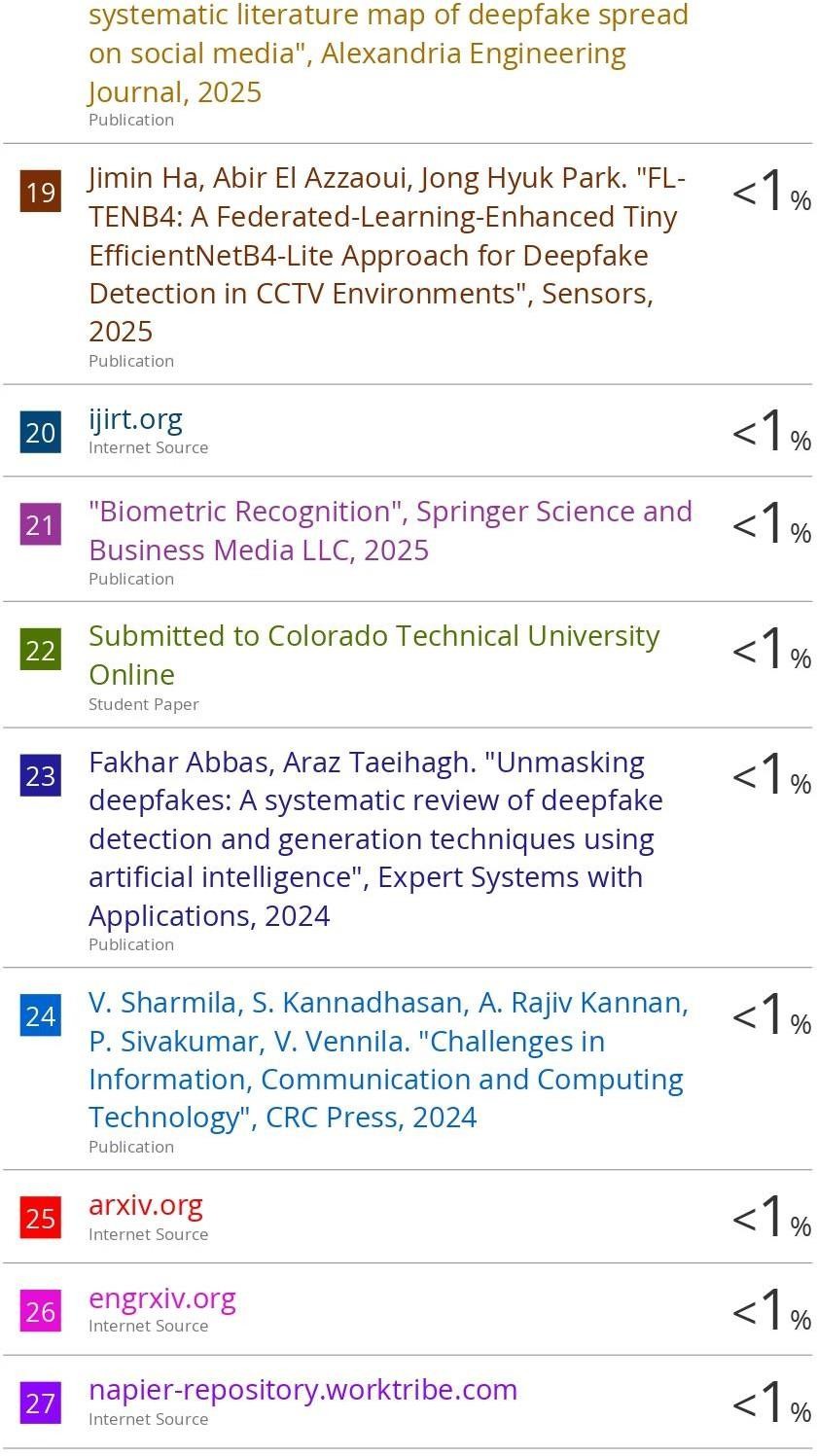


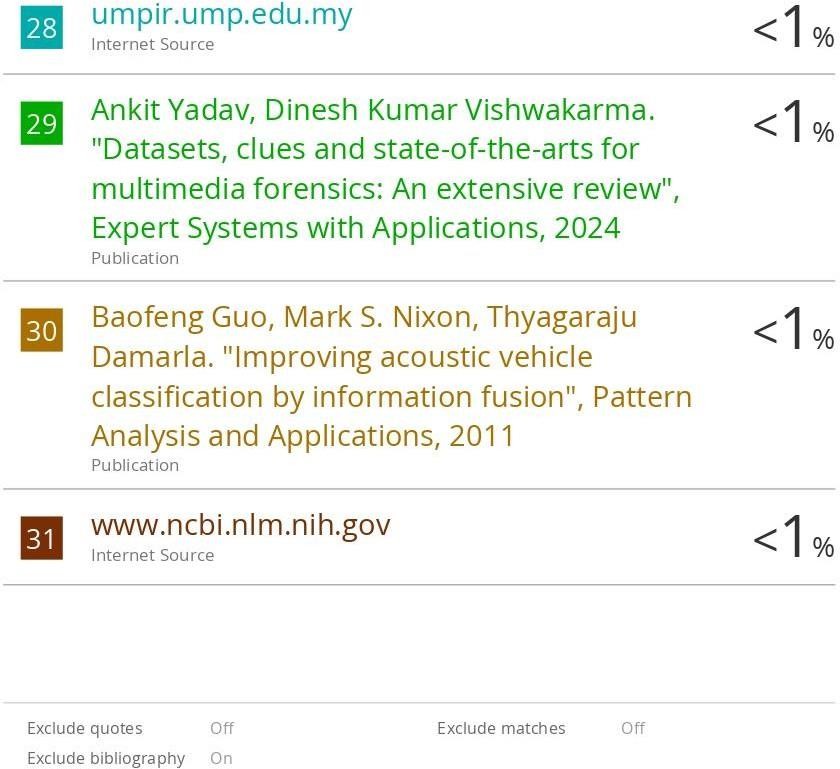












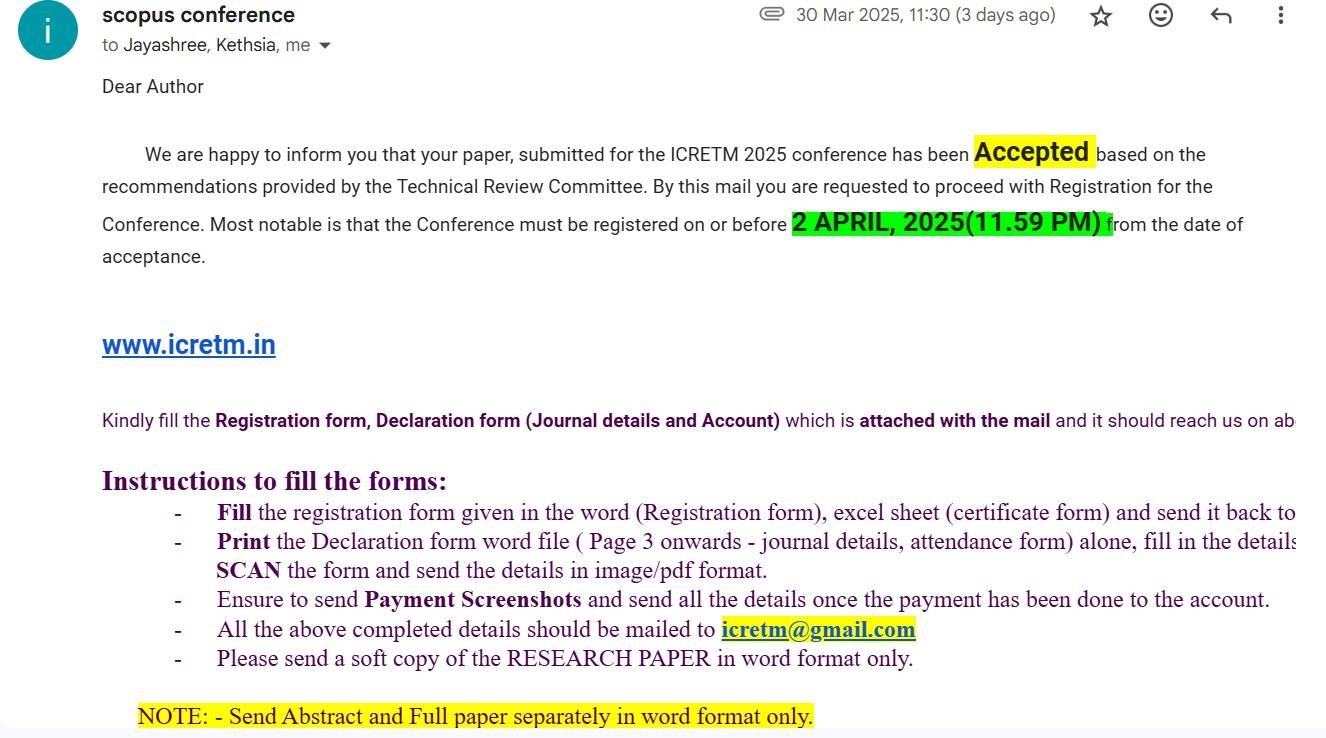
**A.5 PAPER PUBLICATION**

**Publication Name: ICRETM 2025 Scopus** Conference

#### Conference Details:

* 5th International Conference On Recent Trends In Engineering Technology And Management .
* Organized By [Suguna College of Engineering](https://www.sugunace.com/), Coimbatore, Collaboration with [Samarkand State University](https://www.samdu.uz/) Uzbekistan and Research Organisation (Osiet).

**Conference Date:** 4th – 5th April 2025



# REFERENCES

1. Sunisa Soponmanee Jirapond Muangprathub Laor Boongasame, Jindaphon Boonpluk and Karanrat Thammarak. Design and implement deepfake video detection using vgg-16 and long short-term memory. Wiley Journal on Applied Computational Intelligence and Soft Computing, 2023.
2. Rongrong Ni Siyuan Yang Yao Zhao Yang Yu, Xiaohui Zhao and Alex C. Kot. Augmented multi-scale spatiotemporal inconsistency magnifier for generalized deepfake detection. IEEE Transactions on Multimedia, 25:8487–8498, 2023.
3. Haibo Hu Qiao Xue Yaxin Xiao Li Tang, Qingqing Ye and Jin Li. Deepmark: A scalable and robust framework for deepfake video detection. ACM Transactions on Privacy and Security, 27:1–26, 2024.
4. Jiyou Chen Zhiqing Guo, Gaobo Yang and Xingming Sun. Exposing deepfake face forgeries with guided residuals. IEEE Transactions on Multimedia, 25:8458– 8470, 2023.
5. Baoying Chen and Shunquan Tan. Featuretransfer: Unsupervised domain adaptation for cross-domain deepfake detection. Wiley Security and Communication Networks, 2021.
6. Wei Wang Juan Hu, Xin Liao and Zheng Qin. Detecting compressed deepfake videos in social networks using frame-temporality two-stream convolutional network. IEEE Transactions on Circuits and Systems for Video Technology, 32:1089–1102, 2022.
7. Yingcan Li Xinpeng Zhang Pradeep K. Atrey Feng Ding, Guopu Zhu and Siwei Lyu. Anti-forensics for face swapping videos via adversarial training. IEEE Transactions on Multimedia, 24:3429–3441, 2022.
8. Nishat Tasnim Roza S. M. Ahsanul Hoq Mohammad Monirujjaman Khan Arjun Singh Atef Zaguia Hasin Shahed Shad, Md. Mashfiq Rizvee and Sami Bourouis. Comparative analysis of deepfake image detection method using convolutional neural network. Wiley Journal on Computational Intelligence and Neuroscience, 2021.
9. Zahid Akhtar Wassim Hamidouche Abdenour Hadid Bachir Kaddar, Sid Ahmed Fezza and Joan Serra-Sagrista´. Deepfake detection using spatiotemporal transformer. ACM Transactions on Multimedia Computing, Communications and Applications, 20:1–21, 2024.
10. Vincenzo Loia Chiara Pero Federico Becattini, Carmen Bisogni and Fei Hao. Head pose estimation paterns as deepfake detectors. ACM Transactions on Multimedia Computing, Communications and Applications, 20:1–24, 2023.
11. Shudong Li Zhaoquan Gu Huimin Zhao Kaihan Lin, Weihong Han and Yangyang Mei. Detecting deepfake videos using spatiotemporal trident network (stn). ACM Transactions on Multimedia Computing, Communications and Applications, 20:1–20, 2024.
12. Zhenrong Deng Xiaonan Luo Rui Yang, Rushi Lan and Xiyan Sun. Deepfake video detection using facial feature points and ch-transformer. ACM Transactions on Multimedia Computing, Communications and Applications, 2024.
13. Mayank Vatsa Aman Mehra, Akshay Agarwal and Richa Singh. Motion magnified 3-d residual-in-dense network for deepfake detection. IEEE Transactions on Biometrics, Behavior, and Identity Science, 5:39–52, 2022.
14. Kyungha Min Sangho Son, Jaekyu Lee and Wooju Kim. Enhancing deepfake detection: Spatial-temporal preprocessing and self-attention resi3d model. 2023 6th Artificial Intelligence and Cloud Computing Conference (AICCC), pages 27–35, 2023.
15. Kelton A. P. Costa Luis A. Souza Ju´nior Douglas Rodrigues Javier Del Ser David Camacho Leandro A. Passos, Danilo Jodas and Joa˜o Paulo Papa. A review of deep learning-based approaches for deepfake content detection. Wiley’s Survey Article on Expert System, 2024.
16. Wei Lu Xiangyang Luo Jiarui Liu, Kaiman Zhu and Xianfeng Zhao. A lightweight 3d convolutional neural network for deepfake detection. Wiley’s International Journal Of Intelligent Systems, 2021.
17. Wenyu Liu Zenan Shi and Haipeng Chen. Face reconstruction-based generalized deepfake detection model with residual outlook attention. ACM Transactions on Multimedia Computing, Communications and Applications, 2024.
18. Khalid Mahmood Malik Fatima Khalid, Ali Javed and Aun Irtaza. Explanet: A descriptive framework for detecting deepfakes with interpretable prototypes. IEEE Transactions on Biometrics, Behavior, and Identity Science, 2024.
19. Thai-Trang Nguyen Lisa Fiedler Peizhu Qian Vaibhav Unhelkar Tina Seidel Gjergji Kasneci Yao Rong, Tobias Leemann and Enkelejda Kasneci. Towards human-centered explainable ai: A survey of user studies for model explanations. IEEE Transactions on Pattern Analysis and Machine Intelligence, 46, 2023.
20. Massimiliano Todisco Wanying Ge, Jose Patino and Nicholas Evans. Explaining deep learning models for spoofing and deepfake detection with shapley additive explanations. ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing, 2022.
21. Zahid Akhtar Wassim Hamidouche Abdenour Hadid Bachir Kaddar, Sid Ahmed Fezza and Joan Serra-Sagrista´. Explainable deep-fake detection using visual interpretability methods. 2020 3rd International Conference on Information and Computer Technologies (ICICT), 2020.

.