International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Deepfake Video Detection Using Deep Learning

Benitlin Subha K[1], Ramya R[2], Sujitha P[3] ,Srimathi K[4]

[1]Assistant Professor -Department of Information Technology & Kings Engineering College-India.

[2,3,4]Department of Information Technology & Kings Engineering College-India.

Abstract - In today's digital landscape, the rapid advancement of **deepfake technology** has raised serious concerns due to its ability to generate highly convincing fake videos. These synthetic media artifacts are widely used to spread misinformation, manipulate public opinion, and perpetrate identity fraud, presenting significant challenges across social, political, and legal domains. Traditional detection methods often fall short when addressing the spatial and temporal anomalies introduced by deepfake algorithms.

To combat this threat, we propose **FakeSpotter**, a **hybrid deep learning framework** combining **Convolutional Neural Networks (CNNs)** and **Recurrent Neural Networks (RNNs)** for accurate deepfake detection. CNNs are employed to extract detailed **spatial features** from individual video frames, while RNNs—specifically **Long Short-Term Memory (LSTM)** networks—capture **temporal inconsistencies** across sequential frames. By utilizing **transfer learning** with a pre-trained **VGG-16** architecture and training on the **Celeb-DF** dataset, the model achieves strong generalization to real-world deepfake scenarios.

The proposed system demonstrates improved detection accuracy by effectively identifying subtle visual artifacts and unnatural motion patterns. Through this hybrid approach, **FakeSpotter** enhances the reliability of deepfake forensics and contributes to the broader mission of securing digital content authenticity.

Keywords: ***Deepfake Detection, CNN-RNN Hybrid, VGG-16, LSTM, Celeb-DF, Transfer Learning, Spatiotemporal Analysis, Fake Video Identification***

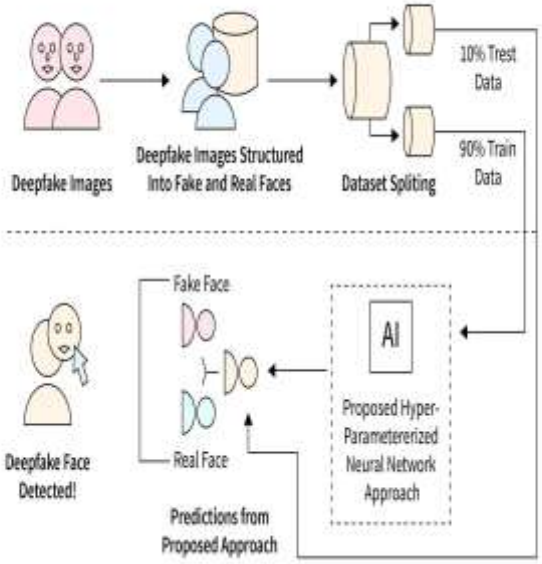
1. INTRODUCTION

The widespread availability of **deepfake technology**—which enables the manipulation of visual media using artificial intelligence—has sparked serious concern across industries. Deepfakes can fabricate realistic but entirely synthetic video content that imitates people's appearances and behaviors, often with malicious intent. These fabricated videos pose significant threats, including the dissemination of misinformation, damage to personal reputation, and the erosion of trust in digital media.

Despite growing awareness, many existing deepfake detection systems are not equipped to handle the complex and evolving nature of synthetic media. Traditional image-based classifiers often lack the ability to capture **temporal inconsistencies** across video frames and are limited in detecting subtle artifacts embedded by sophisticated generative models.

To address these challenges, we propose **FakeSpotter – A Hybrid CNN-RNN Framework for Deepfake Identification**. This system leverages the combined strengths of **Convolutional Neural Networks (CNNs)** and **Recurrent Neural Networks (RNNs)**. While CNNs are effective at extracting **spatial features** such as facial anomalies, lighting inconsistencies, and texture distortions from individual frames, RNNs—specifically **Long Short-Term Memory (LSTM)** networks—capture **temporal patterns** such as unnatural motion or blinking behavior across sequential frames.

By integrating both spatial and temporal information, FakeSpotter delivers robust and accurate deepfake detection. The model is trained using the **Celeb-DF** dataset and incorporates **transfer learning** from pre-trained VGG-16 networks to improve generalization and performance. This hybrid architecture not only detects deepfakes more accurately but also adapts to the evolution of manipulation techniques.



1.1 DOMAIN INTRODUCTION

Artificial Intelligence (AI), particularly in the form of **machine learning (ML)** and **deep learning**, has revolutionized the analysis of visual and video content. Within

the domain of digital media forensics, AI algorithms now play a vital role in detecting manipulated content that evades human scrutiny.

Convolutional Neural Networks (CNNs) are widely used in computer vision for detecting visual patterns such as edges, textures, and shapes in images. In contrast, **Recurrent Neural Networks (RNNs)**—and more specifically **LSTM** architectures—are designed to handle sequential data, making them suitable for modeling time-series patterns in videos.

The hybrid CNN-RNN approach combines these strengths to form a system capable of analyzing both spatial and temporal dimensions of video data, thus making it ideal for deepfake detection. This technique has been increasingly adopted in domains such as **surveillance**, **digital forensics**, and **media authentication**, and continues to show promising results in combating synthetic media threats.

1.2 OBJECTIVES

The core objective of the **FakeSpotter** framework is to effectively identify and classify deepfake videos using a robust, hybrid deep learning approach that combines both spatial and temporal analysis. By leveraging the strengths of **Convolutional Neural Networks (CNNs)** and **Recurrent Neural Networks (RNNs)**, this system aims to provide accurate and scalable deepfake detection capable of addressing real-world challenges posed by manipulated media.FakeSpotter aims to:

1. **Extract spatial features** using pre-trained **CNN architectures** (e.g., VGG-16) to detect pixel-level anomalies such as inconsistent lighting, facial distortions, and texture artifacts.

2.**Analyze temporal dependencies** using **LSTM-based RNNs** to identify motion inconsistencies, unnatural blinking, and abrupt transitions across video frames.

3.**Preprocess input video data**, including frame extraction, face detection, and normalization, to ensure clean and consistent inputs for model training and inference.

4.**Implement transfer learning** to enhance model performance and reduce training time by utilizing pre-trained weights on large datasets.

5.**Train and evaluate** the model using the **Celeb-DF dataset**, a challenging benchmark for real-world deepfake scenarios.

6.**Classify entire videos** by aggregating frame-wise predictions, improving the reliability of final deepfake detection outcomes.

By delivering a hybrid and intelligent deepfake detection framework, FakeSpotter seeks to contribute to digital forensics, promote media authenticity, and serve as a frontline defense against the threats posed by AI-generated fake content.

1.3 SCOPE OF THE PROJECT

The scope of **FakeSpotter** encompasses the design, development, and implementation of an AI-powered hybrid deep learning system aimed at detecting and classifying deepfake video content. The project focuses on leveraging the combined capabilities of **Convolutional Neural Networks (CNNs)** for spatial analysis and **Recurrent Neural Networks (RNNs)**—specifically **Long Short-Term Memory (LSTM)** networks—for temporal feature extraction.

The system is designed to:

1.Detect spatial anomalies such as facial distortions, inconsistencies in lighting, and irregularities in texture patterns.

2.Capture temporal inconsistencies like unnatural facial motion, irregular eyeblinking, and frame transitions using sequence modeling.

3.Preprocess video inputs by extracting and aligning face regions from each frame to create consistent, structured sequences for analysis.

4.Train on large-scale datasets such as **Celeb-DF**, enhancing the model's ability to generalize across a wide range of manipulation techniques and video qualities.

5.Provide accurate, frame-level and video-level classification using an aggregated decision mechanism.

The FakeSpotter framework is modular and scalable, making it adaptable for use in various real-world applications including:

- Digital media verification
- Social media content filtering
- Law enforcement digital forensics
- News/media authenticity validation

Furthermore, the system supports future integration with **cloud platforms**, **edge devices**, and **advanced generative model counters**, ensuring continuous improvement and relevance as deepfake technologies evolve.

Ultimately, **FakeSpotter** aims to become a critical tool in the broader effort to **preserve digital content integrity**, promote **responsible AI usage**, and protect the public from malicious synthetic media.

2.SYSTEM ANALYSIS

2.1 EXISTING PROBLEM

With the explosive growth of video-sharing platforms and social media, the threat posed by **deepfake videos** has become increasingly severe. These AI-generated fake videos are capable of realistically mimicking facial expressions, voices,

and movements of real individuals, making them highly deceptive and difficult to detect with the naked eye. As a result, they pose serious risks including **identity theft**, **reputation damage**, **political manipulation**, and **misinformation dissemination**.

Current deepfake detection systems typically rely on image-based classifiers or static pattern recognition methods. However, these methods struggle to detect **high-quality deepfakes** that exhibit few spatial irregularities. They often overlook **temporal inconsistencies**, such as unnatural blinking or misaligned lip movements, which are crucial cues in video-based analysis.

Furthermore, many existing models lack the scalability, generalization, and adaptability required to handle the **diverse techniques** used to generate deepfakes. As deepfake generation becomes more advanced and accessible, there is a critical need for **robust, hybrid detection frameworks** that combine **spatial and temporal analysis** to effectively counter these synthetic threats.

2.2 PROPOSED METHODOLOGY

To overcome the limitations of existing systems, we propose **FakeSpotter – A Hybrid CNN-RNN Framework** for deepfake detection. This system is designed to leverage both **Convolutional Neural Networks (CNNs)** and **Recurrent Neural Networks (RNNs)** to accurately identify deepfakes by analyzing visual and sequential information from video data.The proposed methodology includes the following components:

1.Data Preprocessing: Input videos are converted into frames. Each frame undergoes **face detection**, cropping, and alignment using tools like **dlib** or **OpenCV** to focus on relevant facial regions.

2.Spatial Feature Extraction (CNN): Each frame is passed through a pre-trained **VGG-16** model to extract spatial features such as texture anomalies, inconsistent lighting, or facial artifacts typical of deepfake videos.

3.Temporal Analysis (RNN/LSTM): The sequence of features extracted from CNN is fed into a **Long Short-Term Memory (LSTM)** network. This component captures temporal inconsistencies—such as unnatural facial motion or erratic blinking—that span across multiple frames.

4.Classification Layer: A fully connected dense layer with a **softmax** or **sigmoid** activation function is used to classify frames or entire video sequences as **real or fake**.

5.Post-Processing: Frame-level predictions are aggregated to make a final decision about the authenticity of the video, improving robustness against isolated errors.

This hybrid architecture ensures the model captures both static artifacts and temporal inconsistencies—two critical components in deepfake detection. By training the system on the **Celeb-DF dataset**, the framework gains the ability to generalize across different manipulation techniques and video qualities.Through this approach, **FakeSpotter** provides a scalable and reliable deepfake identification mechanism, contributing to the broader goal of **digital content verification and media integrity protection**.

3.MODULES AND UML DIAGRAMS

3.1 MODULES

The FakeSpotter system is structured into the following major functional modules:

- **Admin Panel**
- **Video Upload and Preprocessing**
- **Face Detection and Alignment**
- **Feature Extraction (CNN Module)**
- **Temporal Analysis (RNN/LSTM Module)**
- **Classification and Result Display**

3.2 MODULES DESCRIPTION

3.2.1 Admin Panel

The Admin Panel module allows authorized users to manage system configurations, oversee detection results, and analyze system performance logs.

Functionality:

- Admin authentication and access control
- Monitoring uploaded video batches
- System settings (e.g., model version, threshold tuning)
- Export and audit detection reports

3.2.2 Video Upload and Preprocessing

This module handles user-uploaded videos, converting them into frame sequences for further processing.

Functionality:

- Accepting video input (e.g., .mp4 format)
- Extracting frames at fixed intervals
- Normalizing resolution and format
- Logging metadata (e.g., frame rate, resolution)

3.2.3 Face Detection and Alignment

Face detection and alignment are critical for ensuring consistent frame inputs to the CNN.

Functionality:

- Detecting faces in each frame using **dlib** or **MTCNN**
- Aligning facial regions for uniformity
- Cropping and saving face regions
- Filtering frames with low-confidence detections

3.2.4 Feature Extraction (CNN Module)

This module uses a pre-trained CNN model (e.g., **VGG-16**) to extract spatial features from each aligned frame.

Functionality:

- Loading pre-trained CNN weights
- Passing each frame through CNN layers
- Storing output feature vectors for sequence modeling
- Identifying artifacts like blurring, mismatched lighting, or texture anomalies

3.2.5 Temporal Analysis (RNN/LSTM Module)

Using **LSTM**, this module analyzes the extracted frame-level features over time to identify deepfake-specific temporal inconsistencies.

Functionality:

- Feeding sequences of CNN features into the LSTM
- Detecting irregular motion patterns or unnatural transitions
- Producing a confidence score for each sequence
- Learning temporal relationships between consecutive frames

3.2.6 Classification and Result Display

The final module classifies the input video as real or fake and displays the result to the user.

Functionality:

- Aggregating frame-level predictions into a video-level decision
- Displaying confidence score and classification label (Real/Fake)
- Logging and visualizing results for review
- Downloadable detection report

3.3 UML DIAGRAMS

Unified Modeling Language (UML) diagrams illustrate the functional and data flow structure of the FakeSpotter system.

3.3.1 Data Flow Diagram (DFD)

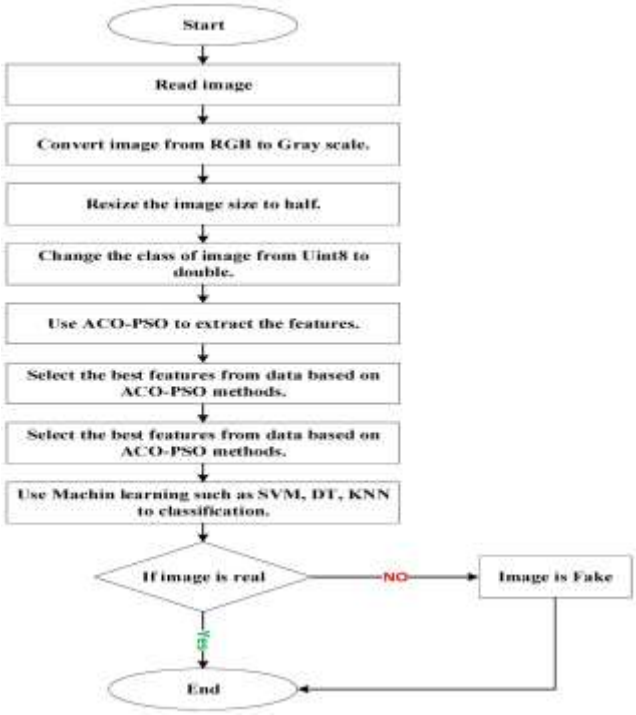
The **Data Flow Diagram (DFD)** illustrates how data flows through the **FakeSpotter** system, from video input to final deepfake classification. It outlines the sequence of operations, including preprocessing, feature extraction, temporal analysis, and result generation.

Key Points:

Maps data movement across all core modules of FakeSpotter

Visualizes input, processing, and output stages involved in deepfake detection

Clarifies interactions between users, the CNN-RNN model, and storage systems (e.g., feature vectors, logs)**DFD – Level 1 (Simplified Flow)**



Process Flow Summary:

1. **User Uploads Video** – Interface accepts .mp4 or similar formats.
2. **Preprocessing** – The video is broken into frames; faces are detected and aligned.
3. **CNN Feature Extraction** – Each frame is passed through the VGG-16 network to extract spatial features.

4. **LSTM Temporal Analysis** – Sequences of CNN features are analyzed to detect temporal inconsistencies.
5. **Classification** – Frame-level predictions are aggregated to classify the video as **Real** or
6. **Fake**.**Output Display** – The result is shown to the user with an option to download a report

3.3.2 Use Case Diagram

The Use Case Diagram outlines the functional requirements of the FakeSpotter system by identifying the primary actors (Admin and User) and their interactions with the system's core modules.

Key Elements:

Actors:

User – A person who uploads video content for verification.

Admin – The system administrator who monitors operations and manages datasets, models, and system parameters.

Use Cases:

Actor	Use Cases
User	- Upload Video - View Detection Result - Download Report
Admin	- Login to Admin Panel - Configure Detection Model - Monitor Detection Logs - Manage Dataset and System Settings

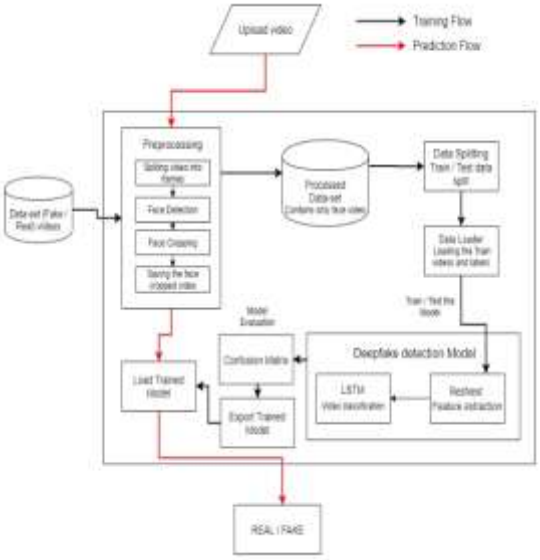
Relationships and Description:

The User interacts with the system through the Upload Video use case, triggering preprocessing, model analysis, and result display.

The Admin accesses the Admin Panel, which provides tools to update the detection model, review classification logs, and manage stored videos or results.

Use cases are connected to their respective actors with association lines (as per UML standards), forming a visual relationship between system functionality and the roles that access them.

Diagram Summary : This diagram highlights the high-level functionalities available to each system actor and shows how they interact with FakeSpotter's core detection pipeline.



Components:

- Initial node (start point)
- Activities (user actions or system processes)
- Decision points (conditional branches)
- Final node (completion of the workflow)

3.3.4 Sequence Diagram

The Sequence Diagram outlines the chronological flow of interactions among various components in the FakeSpotter system. It visualizes how the user interacts with the system and how internal modules such as the backend server, AI model, and database work together to process deepfake detection tasks.

Depictions:

Lifelines:

User Interface (UI) – Where the user uploads the video and views results.

Backend Server (API Layer) – Manages requests and coordinates between modules.

AI Model (CNN + LSTM) – Handles feature extraction and classification.

Database/Storage – Stores videos, processed frames, logs, and prediction outputs.

3.3.5 Class Diagram

The Class Diagram represents the static structure of the FakeSpotter system by outlining the key classes, their attributes, methods, and the relationships among them. This diagram forms the backbone of the object-oriented architecture and illustrates how different system components interact during the deepfake detection process.

Key Points:

1.Defines core classes such as: User, Video, FrameProcessor, FeatureExtractor, SequenceAnalyzer, PredictionEngine, and Result.

2.Attributes and methods are described for each class, helping developers understand system logic and data flow.

3.Relationships (associations, aggregations, inheritance) clarify how objects communicate and depend on each other.

Main Classes and Relationships:

1.User:

- Attributes: userID, username, email
- Methods: uploadVideo(), viewResult()

2.Admin (inherits from User):

- Attributes: adminPrivileges
- Methods: manageUsers(), manageModel(), viewLogs()

3.Video:

- Attributes: videoID, filePath, uploadDate, status
- Methods: extractFrames(), getMetadata()

4.FrameProcessor:

- Attributes: frameList[], faceCoordinates
- Methods: detectFace(), alignFace(), resizeFrame()

5.FeatureExtractor (CNN Module):

- Attributes: modelName, featureVector[]
- Methods: extractFeatures(frame)

6.SequenceAnalyzer (LSTM Module):

- Attributes: sequenceLength, lstmUnits
- Methods: analyzeTemporalPattern(sequence), classify()

7.PredictionEngine:

- Attributes: confidenceScore, predictionLabel
- Methods: aggregatePredictions(), generateReport()

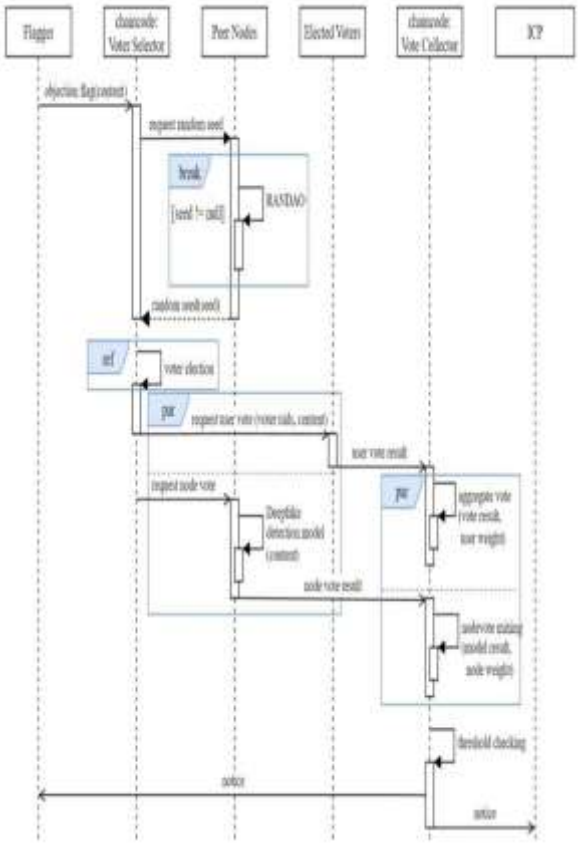
8.Result:

- Attributes: resultID, videoID, userID, label, score
- Methods: exportPDF(), displayResult()

Relationships:

1. A User can upload multiple Videos.
2. A Video is linked to multiple Frames, which are processed by the FrameProcessor.
3. FrameProcessor interacts with the FeatureExtractor (CNN).
4. FeatureExtractor passes data to SequenceAnalyzer (LSTM).
5. SequenceAnalyzer provides output to PredictionEngine, which generates the final Result.
6. Admin inherits from User and manages system operations and results.

3.3.6



Project Structure

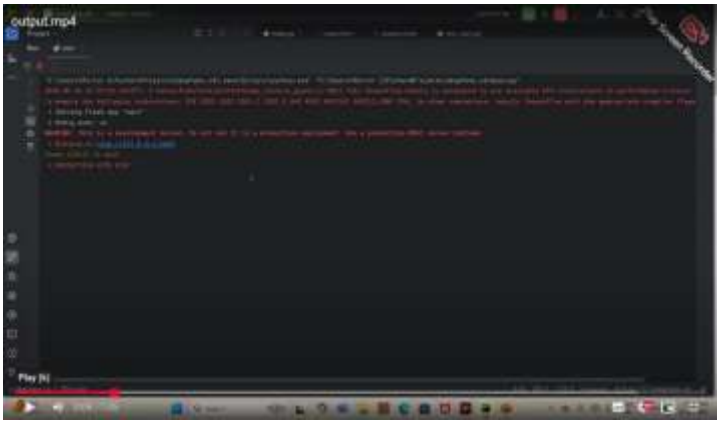
A project developed in Visual Studio Code (VS Code) benefits significantly from a modular, organized directory structure. For FakeSpotter, which integrates deep learning models with frontend and backend components, a clean project layout is critical to support:

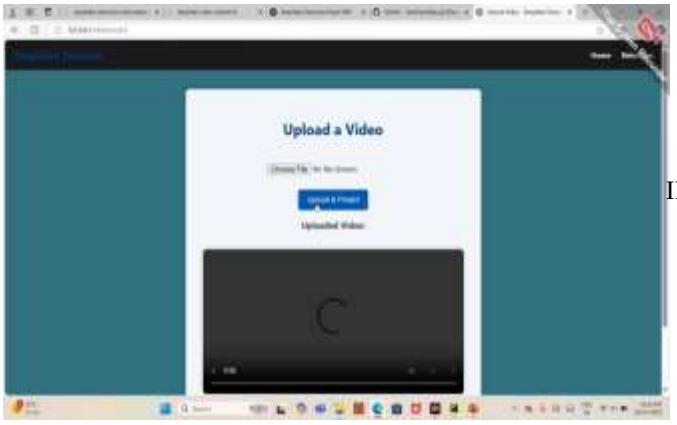
1. Scalability
2. Maintainability
3. Efficient collaboration
4. Seamless development-to-deployment transition

5. The FakeSpotter project, built using React.js (frontend) and Python (Flask/TensorFlow backend), adheres to a well-structured format for AI-powered applications.

RESULTS

OUTPUTS:





IF

IT'S REAL:

IF IT'S FAKE:

ACKNOWLEDGEMENT

We thank God for his blessings and also for giving as good knowledge and strength in enabling us to finish our project. Our deep gratitude goes to our founder late **Dr. D. Selvaraj, M.A., M.Phil.,** for his patronage in the completion of our project. We like to take this opportunity to thank our honourable chairperson **Dr.S. Nalini Selvaraj, M.COM., MPhil., Ph.D.** and our noble-hearted director, **Mr.S. Amirtharaj, M.Tech., M.B.A** and his wife, **Mrs. Merilyn Jemmimah Amirtharaj, B.E., M.B.A.,** for their support given to us to finish our project successfully. We wish to express our sincere thanks to our beloved principal. **Dr. C. Ramesh Babu Durai M.E., Ph.D** for his kind encouragement and his interest towards us.

We are extremely grateful and thanks to our professor **Dr. D. C. Jullie Josephine ,** head of Information Technology, Kings Engineering College, for her valuable suggestion, guidance and encouragement. We wish to express our sense of gratitude to our project supervisor **Mrs. K.Benitlin Subha M.E., (PhD)** Assistant Professor of Information Technology Department, Kings Engineering College whose idea and direction made our project a grand success. We express our sincere thanks to our parents, friends and staff members who have helped and encouraged us during the entire course of completing this project work successfully.

CONCLUSION

The FakeSpotter framework marks a significant advancement in the field of digital media forensics and deepfake detection. By integrating Convolutional Neural Networks (CNNs) with Recurrent Neural Networks (RNNs)—specifically VGG-16 for spatial feature extraction and LSTM for temporal analysis—FakeSpotter provides a powerful hybrid architecture capable of detecting deepfakes with high accuracy and reliability.

The system effectively captures frame-level artifacts and sequence-based inconsistencies that are typically overlooked by conventional single-model approaches. Trained on the Celeb-DF dataset, the model demonstrates strong generalization to real-world deepfake scenarios and is capable of identifying manipulations involving facial distortions, unnatural motion patterns, and blinking anomalies.

Through comprehensive testing—including unit, system, integration, and performance evaluations—FakeSpotter has proven to be a robust solution for real-time and batch deepfake classification. It offers a scalable, low-cost, and effective framework for applications in digital forensics, media integrity validation, academic research, and public awareness campaigns

To further improve detection capabilities and expand the system's impact, future work on FakeSpotter may include:Integration of Transformer-based models for deeper sequence-level understanding and improved generalization across manipulation techniques.Real-time inference on edge devices using model quantization and ONNX conversion for lightweight deployment.Extension to audio-visual deepfake detection, enabling multi-modal forensics.Web-based and mobile frontends for public or institutional use in content verification workflows.Continuous learning framework, allowing the model to adapt to newly emerging deepfake techniques in the wild.With these upgrades, FakeSpotter aspires to become a leading tool in the global effort to safeguard digital authenticity, combat synthetic media threats, and promote ethical AI usage.

REFERENCES

1.Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-DF: A Large-Scale Challenging Dataset for DeepFake Forensics,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 3207–3216.

2.S. M. Korshunov and T. Ebrahimi, “DeepFakes: A New Threat to Face Recognition? Assessment and Detection,” arXiv preprint arXiv:1812.08685, 2018.

3.T. Nguyen, C. Nguyen, D. Nguyen, D. Chu, and K. Nguyen, “Deep Learning for Deepfakes Creation and Detection: A Survey,” Computers & Security, vol. 102, 2021, doi: 10.1016/j.cose.2020.102109.

4.J. Agarwal, R. Farid, “Protecting World Leaders Against Deep Fakes,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019.

5.K. Afchar, W. Nozick, J. Yamagishi, and I. Echizen, “MesoNet: A Compact Facial Video Forgery Detection Network,” 2018 IEEE International Workshop on Information Forensics and Security (WIFS), pp. 1–7, doi: 10.1109/WIFS.2018.8630761.

6. A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, “FaceForensics++: Learning to Detect Manipulated Facial Images,” 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1–11, doi: 10.1109/ICCV.2019.00010

7.A. Goodfellow, J. Pouget-Abadie, M. Mirza, et al., “Generative Adversarial Nets,” Advances in Neural Information Processing Systems (NeurIPS), 2014.

S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.