

**Fauxshield: An Explainable Deepfake Detection and Awareness Platform
Using Hybrid Deep Learning**

**Qasim Abbas, Nida Mohsin, Mohammad Shah, Muhammad Awais Ali,
Ayesha Yousaf and Farhad Ali Farhat**

This is a copy of the author's accepted version of a paper subsequently published in the proceedings of the 13th International Conference on Future Internet of Things and Cloud (FiCloud), Granada, Spain, 20 - 22 Jul 2026.

The final published version will be available online at:

<https://doi.org/10.1109/ficloud70576.2026.00046>

© 2026 IEEE . This manuscript version is made available under the CC-BY 4.0 license

<https://creativecommons.org/licenses/by/4.0/>

For the purpose of open access, the author(s) has applied a Creative Commons Attribution (CC BY) license to any Accepted Manuscript version arising.

The WestminsterResearch online digital archive at the University of Westminster aims to make the research output of the University available to a wider audience. Copyright and Moral Rights remain with the authors and/or copyright owners.

FauxShield: An Explainable Deepfake Detection and Awareness Platform Using Hybrid Deep Learning

Qasim Abbas¹, Nida Mohsin², Mohammad Shah¹, Muhammad Awais Ali², Ayesha Yousaf², Farhad Ali Farhat²

¹University of Westminster, London, United Kingdom

{abbasq, M.Shah1}@westminster.ac.uk

²University of Management and Technology, Lahore, Pakistan

awaisali@umt.edu.pk, nidamohsin999@gmail.com, ayeshayousaf034@gmail.com, farhadalifarhat@gmail.com

Abstract—Deepfake technology is a risk for misinformation, privacy and security. The platform we propose uses Deep Learning and Explainable AI to find deepfakes in images and videos. It has an user-friendly design. The system tries to solve the AI "black-box" problem by using LIME to explain how it works. It also includes resources to help people understand more. Our platform is built with PHP Laravel and TensorFlow. It has an API that allows other applications to use it. The goal is to make deepfake analysis easy to use for everyone: journalists, researchers organizations and regular people. The Xception model for images is expected to be 92% accurate. The CNN+RNN model for videos is expected to be 85% accurate. The overall accuracy is expected to be 88%. Our platform, FauxShield is different, from detection systems. It helps users understand deepfakes and their impact. We provide explanations of how we make our decisions. We offer support in many languages. This way we want to help people identify deepfakes and know what they mean.

Keywords—Deepfake Detection, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Xception, Explainable AI (XAI), LIME, Multimedia Forensics, Video Analysis, Image Classification

I. INTRODUCTION

A. Motivations

Artificial Intelligence (AI) has facilitated the development of highly realistic synthetic media; [1], [2]. however, its misuse via deepfakes jeopardizes trust, privacy, and security [3], [4]. Current detection methods often fall short in terms of usability, interpretability and accessibility. Many of these methods also rely on models that are not transparent. This project aims to address these issues by developing an scalable detection system. The system prioritizes reliability, interpretability, awareness and user-centered design.

B. Project Overview

The proposed platform provides a framework to identify and explain deepfakes in images and videos. It uses CNN layers to examine frames. Find objects in the same location. The system also analyzes how objects change over time in videos. This helps it understand what is happening in the video. The platform uses LIME to highlight which parts of the video are used to make predictions. This makes it clearer how the system works. The system does more than just detect deepfakes. It also allows users to upload videos. There are teaching tools A chatbot that speaks languages is also included

and it is powered by LLaMA. The system provides a solution, for deepfakes. Deepfakes are a concern and this system helps to address them. It uses learning to analyze videos and images. Deepfakes can be used for things. The system helps to identify and explain them. This chatbot helps users when they need it. The system puts together different technologies, including deep learning, explainable AI and design that is focused on the user. This makes the system more transparent and easier to use for people who want to check if a video is a deepfake. Unlike systems that just try to find deepfakes FauxShield does many things. It finds deepfakes explains how it works teaches people about deepfakes and helps users in languages, all in one place that can be accessed from a browser. The system is designed to help people who are not experts understand what is real and what is not. It also helps people who do not know a lot, about technology to use the system and understand how it works.

C. Problem Statement

The advancement of deepfake technology has rendered media authentication increasingly difficult, thereby heightening risks for journalism, discourse, and cybersecurity. Although numerous detection methods prioritize accuracy, they frequently lack interpretability [5], providing solely binary outputs and constraining user trust and educational significance. The proposed platform tackles this problem by combining learning with explainable AI. It uses LIME to make the detection process more transparent build trust and enable decision-making.

D. Objectives

The main goals of this work are:

- 1) The main goal is to create a kind of computer program that uses deep learning to find fake videos and pictures. This program will use two types of computer models: one that looks at pictures and one that looks at videos.
- 2) We want to add a feature that helps people understand how the computer program makes its decisions. We will use a technique called LIME that makes it easier for people to see why the program says something is fake or not.
- 3) To build a secure and user-friendly web-based platform, for deepfake analysis.

- 4) We also want to help people learn more about videos and pictures and how to spot them. We will make some tools and fun things to do on the computer that teach people about this stuff.
- 5) To make a chatbot that can help people in time using a model called LLaMA and make it available in many languages.

E. Contributions

The main things we have done far are:

- 1) We made a system that can find deepfakes in both images and videos which makes it easier to check kinds of media.
- 2) The proposed system incorporates Explainable AI techniques, specifically LIME, to provide clear and interpretable insights into model predictions, helping users understand why a particular result was generated.
- 3) A web-based, browser-accessible platform has been designed to support multilingual awareness, making it easier for users from different language backgrounds to access information and understand deepfake-related risks.
- 4) A real-time inference pipeline has been implemented using FastAPI, allowing fast and efficient processing of uploaded media with minimal delay in detection results.
- 5) The solution is designed with a user-centric approach, ensuring that it is simple and intuitive for non-technical users to operate without requiring any technical expertise.

II. RELATED WORK

The growth of deepfake content has led to the creation of detection systems. These systems have features, users and designs. Current research and practical implementations mainly differ in their support for multimedia use of AI (XAI) accessibility, API support and user interaction.

A. Multimedia Support

Deepfake detection systems work with types of media. Some only work with videos while others can handle images, audio and videos. Some systems also analyze voice manipulation to detect speech. However most systems do not support media types. They often only work with one type of media. [6].

B. Explainable AI Integration

Most deepfake detection systems are like boxes. They give results without explaining how they got there. This makes it hard for users to trust the results. Recent studies focus on using AI (XAI) techniques. XAI helps users understand how the system makes predictions. This makes the system more transparent and trustworthy. [5].

C. User-Friendly Interface

A user-friendly interface is crucial for people to use deepfake detection systems. Modern web-based interfaces make it easy for both technical and non-technical users to use. Some systems also offer support, chatbots and live notifications. These features help increase awareness and make the system more accessible.

D. Cost

There are different costs involved when applying the technology of deepfake detection. For example, some detection methods are provided freely; thus, they can be accessed by individuals such as learners and educators. On the other hand, others are costly and can be afforded by only big corporations. The latter cannot be accessed by individuals without financial resources.

E. API Availability

API support is important for integrating deepfake detection systems with other applications. Many modern systems offer API access. This makes it easy to integrate with systems and scalable. However some systems do not offer API access. This limits their use with applications.

F. Summary

Deepfake detection systems differ in ways. They vary in terms of interpretability, multimedia support, accessibility, cost and integration. Now researchers are trying to combine different ways of analyzing media with artificial intelligence and user-friendly design. These features are important for making systems that can help stop the spread of information in places with a lot of resources and in places with few resources. Systems that can find deepfakes need to be available to everyone and make an impact on society. Deepfake detection systems, like the one we are making need to be widely available and socially impactful. That is what we are trying to achieve with our deepfake detection system.

III. PROPOSED METHODOLOGY

A. Overview of Approach

The FauxShield platform is a system that does a lot of things. It can find media and it also tells users how it made its decision. It teaches people about deepfakes too. When someone uses FauxShield they upload a picture or video to the website. The website is safe and easy to use. The uploaded media is then made the size and type so it can be looked at easily. If it is a video the system takes out some frames so it can look at them one by one.

The FauxShield system uses a kind of computer program that is good at looking at pictures and videos. The model is named Convolutional Neural Network and Recurrent Neural Network. The program analyses images and videos in such a way that will allow the detection of whether or not the media content is authentic. FauxShield even informs users about reasons why it believes that certain media content is fake. This is achieved through the use of an interpretation tool that makes

it easier to understand why the system makes such decisions. This interpretive tool is known as Local Interpretable Model Explanations.

The FauxShield system also has a lot of information to help people learn about deepfakes. It has articles and tips and examples of media. It is available in languages. There is also a part of the system for the people who manage it. They can use it to control who uses the system and what they can do. They can also look at what has been flagged as fake.

The FauxShield platform does a lot of things. It can find media it explains how it made its decision and it teaches people about deepfakes. This makes it an useful tool for finding and learning about fake media. The FauxShield platform is good, at finding deepfakes. It also helps people understand what is real and what is not.

B. System Architecture

The deepfake detection platform is designed to be scalable and easy to maintain. It is supposed to provide detection accuracy. Figure 1 shows the system architecture. This includes the frontend user-side, a high-performance backend server, AI-powered detection and explainability modules, educational services and a secure data management layer. The workflow of the tool starts when a User opens the tool in a browser. The Frontend (User Interface), as a responsive web app, provides easy media uploads, visualization of results, and viewing of awareness resources. Multilingual navigation is. The interface has been made easy to use on both computers and mobile devices.

- **Frontend:** The flow of events starts when a person uses the software via an ordinary web browser. The frontend serves as a responsive web application allowing uploading of media, visualizing the outcome and using awareness resources. Multi-language navigation is provided, and the user interface has been optimized for desktop and mobile devices to increase usability.
- **Backend Server and API Layer:** This component is responsible for handling requests from the frontend ensuring proper functioning of processes. Validation and management of the data inputted by users along with authentication are managed by the backend server which forwards the requests to the artificial intelligence detection section via API layer. FastAPI is used to allow efficient performance of endpoints regardless of the high traffic volume ensuring quick processing of deepfake identification, explanation, educational content and chat services.
- **Detection Module:** Our platforms core is the Deep Learning Detection Module, which has two optimized parts:
 - A video analysis model that uses a mix of recurrent layers to look at both what something looks like and how it changes over time. [7].
 - An image analysis model that uses layers and was trained on a collection of images and frames from deepfake videos.

- **Explainability Layer:** The Explainable AI Layer takes the output from the detection. Uses the Local Interpretable Model-Agnostic Explanations framework to create explanations that humans can understand. This module highlights the parts of images or video frames that help the model make a decision. This way the model is not a "black-box" anymore.
- **Educational & Awareness Module:** The Educational and Awareness Module gives users resources like articles and tutorials about deepfakes responsible AI practices and detection techniques. This module helps people learn more about things and makes them more aware.
- **Multilingual Chatbot:** The platform has a chatbot that uses Metas LLaMA language model. It helps users in time by answering their questions about deepfakes and explaining what the system can do. It also supports languages, which makes it easier for people from all over the world to use.
- **Database and Storage Layer:** The Database and Storage Layer uses MySQL to store data. It keeps information, about the media that users upload the results of the detection and what users do on the platform. The media files are stored securely so they can be looked at later.

The deepfake detection platform is designed to be modular, scalable and transparent. It does what it is supposed to do, which is to detect deepfakes and it also teaches people about them. The deepfake detection platform can be used by journalists, researchers organizations and users. The deepfake detection platform is very helpful. Can be used in many different ways.

C. Data Acquisition and Preprocessing

To detect deepfakes we need good and varied data and then we have to prepare it properly. We used datasets like FaceForensics++ and Celeb-DF, which have both fake videos and images. [1], [8].

Preparing the data involved steps:

- **Frame Extraction:** We took 1 frame per second from videos for training and 2 frames per second for testing to get a mix without too much repetition.
- **Face Detection and Alignment:** We used MTCNN to find and align faces making sure they were consistent across samples. [9].
- **Normalization and Resizing:** All images and frames were resized to 299×299 pixels and their pixel values were adjusted to be, between 0 and 1.
- **Data Augmentation:** To make our model better we added some variations to the data like flipping it rotating it changing brightness and adding some noise.

This preparation made sure our dataset was balanced, consistent and diverse enough for learning.

D. Deepfake Detection Models

The detection pipeline incorporated both image-based models as well as video-based models to detect spatial as well as temporal artifacts.

1) *Image-Based Detection Model (CNN/Xception)*: The image detection pipeline targets one-frame deepfakes by revealing manipulation traces and spatial artifacts in facial regions. The Xception-based CNN is pretrained on ImageNet and subsequently fine-tuned for binary classification.

Preprocessing pipeline:

- 1) Resizing images to 160×160 pixels.
- 2) Normalization via the Xception preprocessing function.
- 3) Data augmentation (random horizontal flips, rotations, zoom, and contrast changes) applied only to the training set.

Although more recent transformer-based models such as Vision Transformers (ViT) and Swin Transformers have shown stronger performance in several benchmark studies, Xception-Net was selected in this work due to its efficiency and practical deployment advantages. It requires comparatively lower computational resources and demonstrates stable training behavior, which makes it suitable for a real-time system that operates through a browser-based platform. Since the focus of this study is on building an accessible and working detection system rather than introducing a new backbone architecture, a balance between performance and efficiency was considered more important.

The aim of this project is not about creating an innovative neural network but rather to create a full-fledged system for detecting deepfakes that incorporates explainability as well as the potential for deploying the model into the real world. For these reasons, the selected model architecture facilitates faster inference, ease of use within FastAPI services, and non-technical user-friendliness without the need for powerful computers.

Model architecture:

- **Base feature extractor:** Xception (layers of convolutional blocks are frozen at the beginning and then partially unfrozen during fine-tuning).
- **Pooling and regularization:** Global Average Pooling and Dropout layers (0.5 and 0.3).
- **Classification head:** Dense(64, ReLU) followed by Dense(1, Sigmoid).

Training settings:

- Optimizer: Adam (learning rate= 1×10^{-5})
- Loss function: Binary Cross-Entropy
- Metrics: Accuracy, Precision, Recall, AUC
- Batch size: 32
- Epochs: 50 (Early Stopping patience=5)
- Class imbalance was solved using calculated class weights.

2) *Video-Based Detection Model (CNN/Xception + RNN/LSTM)*: The video detection pipeline captures both spatial and temporal discrepancies in frames. It incorporates a TimeDistributed Xception feature extractor with an LSTM layer for temporal modeling.

Preprocessing pipeline:

- 1) Sampling up to 40 frames per video (every 3rd frame).
- 2) Resizing frames to 160×160 pixels.

- 3) Data augmentation (horizontal flipping as well as brightness change)
- 4) Pad until 40 frames for videos less than 40 frames.

Model architecture:

- Frame-wise Feature Extraction - Time Distributed(Xception, pretrained on ImageNet, last 40 layers not frozen).
- Pooling: TimeDistributed(Global Average Pooling).
- Temporal modeling: LSTM with 128 units.
- Regularization: Dropout(0.5).
- Classification layer: Dense(2, Softmax).

Training settings:

- Optimizer: Adam (learning rate= 5×10^{-6})
- Loss function: Categorical Cross-Entropy
- Batch size: 4
- Epochs: 50 (Early Stopping patience=5)
- Learning Rate Decay – ReduceLROnPlateau
- Check pointing based on best validation accuracy

3) *Training and Validation:* Both models follow an 80–10–10 split for training, validation, and testing. Generalization is encouraged through augmentation, fine-tuning of higher-level convolutional layers, and callback usage (early stopping, learning rate scheduling).

For the image model, Accuracy, Precision, Recall, and AUC are tracked as evaluation metrics. For the video model, categorical accuracy and validation loss are monitored.

The integration of spatial image detection and temporal video classification allows the approach to cover all types of deepfakes while keeping up real-time inference capacity.

E. Explainable AI Integration (LIME)

In order to improve user understanding and trust, LIME algorithm was adopted in the work. Once the deepfake detector makes a prediction about whether the frame contains a deepfake or not, LIME identifies the suspicious regions in the image. As a result, it became evident that the image detection model focused on irregularities in facial edges, eyes, artificial skin, and areas of blending. In the case of videos, the model emphasized changes in the mouth-jaw region caused by inconsistent transitions from one frame to another.

The analysis of such results proved that the deepfake detection model could be relied upon since its explanations coincided with typical deepfake artifacts described in previous works on digital forensics. It can be argued that the model detected significant features related to the presence of a deepfake instead of paying attention to irrelevant object [5].

- **Frame Selection:** For video input, the frame selected for explanation was the 10th frame.
- **Perturbation Process:** LIME perturbs the selected image by masking superpixels and observing model output differences.
- **Visualization:** Generated heatmap is overlaid on the original image after applying boundary marking method.
- **User Presentation:** The results generated are shown through the web interface as an image with highlighting

of the regions along with a probability score assigned by the model itself.

F. Platform Implementation

The proposed methodology was implemented in form of a platform that is accessible via the web browser in order to ensure easy usability and availability for end users. Platform architecture comprises several components that are working together in the process.

Frontend: A web application based on React.tsx was created in order for the user to upload an image or video, to be able to see in real-time detection results and access awareness resources. It was made available on both desktop and mobile devices for better availability.

Backend: The core back-end part of the solution was built using PHP Laravel framework. Laravel is responsible for user authentication and request processing and orchestration of communication among the frontend, deep learning services and the database.

Deep Learning Service Layer: The hybrid CNN-RNN detection model trained as per the procedure described earlier was run using the FastAPI (Python) framework. Media data are processed by the backend and transferred to the fastapi service which performs pre-processing, inference and creates LIME explanations.

Database: MySQL 8.0 was used to save user queries, detection results, and logs generated by the system. It helps in saving a history of detections, auditing detected information, and possibly using real-life datasets to train again.

Deployment: It was first deployed locally on a personal computer before being deployed in the Google Cloud GPU environment (Google Colab Pro+ with Tesla A100). This makes sure that there is enough computational power available to deal with heavy workloads, especially in case of video-based detection experiments.

The multiple component design makes sure that there is precise prediction with clear and understandable outputs.

G. User Interaction Flow

FauxShield offers both a structured and logical experience for the users and guarantees high detection accuracy at the same time. Firstly, the user uploads an image through the secure web portal. Then, the media is processed by the system to ensure that its format is correct, its resolution is normalized, and frames are extracted from it if it is a video file. The hybrid CNN-RNN model analyzes the image or frames to detect any signs of manipulation.

H. Security and Scalability Considerations

The platform uses several security features to ensure safe data handling for the users as well as overall system security. JWT token based user authentication is employed to facilitate secure and stateless sessions while users use the platform. Role-based access control system is also put into practice where different roles are distinguished to determine whether the user is a regular user or administrator and whether he

should have the authority to perform any administrative operations such as accessing system logs and performing any user management actions.

Encrypted uploading mechanism is used to securely handle data uploaded by users on the platform while using the platform. In addition to that, API throttling has been added to make sure that the server doesn't get overwhelmed with requests.

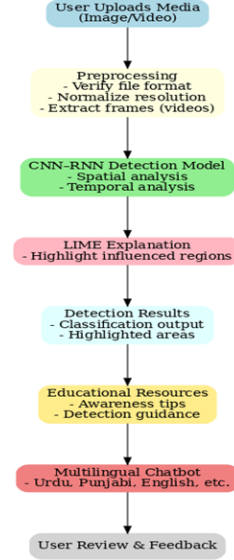


Fig. 1. User interaction flow diagram

IV. EXPERIMENTAL SETUP

The experimental setup was designed to evaluate the performance of the proposed browser-based deepfake detection platform across both image and video modalities, incorporating deep learning models and explainable AI (XAI) capabilities.

The platform maintained near real-time responsiveness during testing.

TABLE I
AVERAGE INFERENCE TIME OF THE PROPOSED DEEPFAKE DETECTION SYSTEM FOR IMAGE AND VIDEO ANALYSIS

Task	Avg Inference Time
Image Detection	120 ms/image
Video Detection	1.8 sec/video

A. Hardware Configuration

The experiments were conducted using the following hardware:

- **Processor:** Intel® Core™ i7-12700K @ 3.6 GHz
- **GPU:** NVIDIA RTX 3080 Ti (12 GB GDDR6X VRAM)
- **RAM:** 32 GB DDR4
- **Storage:** 1 TB NVMe SSD
- **Environment:** Dual setup—local workstation for training and a cloud-based GPU instance (Google Colab Pro+, Tesla A100) for large-scale experiments

B. Software Environment

The following software stack was used:

- **Operating System:** Ubuntu 22.04 LTS
- **Deep Learning Framework:** TensorFlow 2.12 (with CUDA 11.8, cuDNN 8.6)
- **Frontend:** React.tsx
- **Backend:** PHP Laravel + FastAPI (Python)
- **Database:** MySQL 8.0
- **XAI Implementation:** LIME (Local Interpretable Model-agnostic Explanations)
- **Programming Languages:** Python 3.10, TypeScript, PHP 8.1
- **Package Management:** pip (Python), npm (Node.js), Composer (PHP)

C. Dataset Preparation

Experimental validation involved the use of publicly available benchmarking datasets that are frequently utilized in deep fake detection studies. Various datasets were employed in order to test the efficiency of the model in detecting deep fakes based on either image or video data. While faceforensics++ was predominantly utilized for video-based testing, Celeb-DF, along with images derived from DFDC videos, were utilized in the process of evaluating image-based detection.

TABLE II
DATASET USAGE IN THE PROPOSED FRAMEWORK

Dataset	Usage
FaceForensics++	Video training and evaluation
Celeb-DF	Image-based evaluation and testing

TABLE III
DATA DISTRIBUTION ACROSS DATASETS

Dataset	Real Samples	Fake Samples
FaceForensics++	4,000	4,000
Celeb-DF	3,000	3,000

As regards model training and validation, datasets were partitioned using an 80-10-10 approach where 80% of data was used for training, 10% for validation, and the remaining 10% for testing. It should be noted that partition was done in such a way to ensure that training sets did not intersect with evaluation samples. In particular, video datasets were converted to images for temporal-based analysis whereas images in the dataset were used for spatial-based analysis.

D. Model Training & Testing Configuration

- **Image Model:** CNN and Xception network for spatial feature extraction.
- **Video Model:** CNN for frame-level feature extraction, followed by RNN (LSTM) for temporal sequence learning.
- **Optimizer:** Adam (learning rate = 0.0001, $\beta_1 = 0.9$, $\beta_2 = 0.999$)
- **Loss Function:** Binary Cross-Entropy (BCE)

- **Batch Size:** 32 (images), 8 (video sequences)
- **Epochs:** 50 (with early stopping patience of 5 epochs)
- **Regularization:** Dropout (0.4) and L2 weight decay (0.0005)
- **Augmentation:** Random rotation, horizontal flip, brightness adjustment, and Gaussian noise

E. Experimental Reproducibility

To ensure the consistency and repeatability of the results, experiments have been performed based on a consistent training regime. Tensorflow 2.12 has been chosen as the deep learning framework, supplemented by CUDA 11.8 and cuDNN 8.6 accelerations. Training was performed on an NVIDIA RTX 3080 Ti local GPU, while bigger experiments have been carried out on a Tesla A100 GPU in a cloud environment using Google Colab Pro+.

The image model was trained with batch size 32 for 50 epochs. On the other hand, a smaller batch size was chosen for the video CNN+RNN model because of its greater memory consumption. Overfitting has been mitigated by using early stopping and a learning rate scheduler. Furthermore, a random seed has been used in order to make the experiment consistent in terms of data division and initialization. Data has been split into 80/10/10 train/validation/test parts.

F. Evaluation Metrics

The following metrics were computed to assess model performance:

- **Accuracy (ACC):** Percentage of correct predictions.
- **Precision (P):** Correct positive predictions / total predicted positives.
- **Recall (R):** Correct positive predictions / actual positives.
- **F1-Score:** Harmonic mean of precision and recall.
- **AUC-ROC:** Area under the Receiver Operating Characteristic curve.
- **Inference Time:** Average time taken per prediction in the browser.

V. RESULTS AND DISCUSSION

This section presents the experimental results of the proposed deepfake detection framework, evaluated separately on image-based and video-based datasets. Performance is reported in terms of accuracy, precision, recall, F1-score, and area under the ROC curve (AUC). The evaluation demonstrates the robustness of the framework in detecting both frame-level and sequence-level manipulations.

TABLE IV
COMPARATIVE ANALYSIS OF RECENT DEEPAKE DETECTION MODELS
BASED ON ACCURACY, EXPLAINABILITY, AND COMPUTATIONAL
COMPLEXITY

Model	Type	Accuracy	Explainability	Complexity
MesoNet	CNN	84–88%	No	Low
F2-Net	Frequency-based	93–96%	No	High
ViT-based	Transformer	95–98%	Limited	Very High
FauxShield	CNN+RNN + XAI	88%	Yes (LIME)	Moderate

A. Experimental Datasets

- **Image Detection:** Celeb-DF v2 dataset combined with frames extracted from videos.
- **Video Detection:** FaceForensics++ benchmark dataset.

B. Quantitative Results

Table VI summarizes the classification performance of the two models.

TABLE V
PERFORMANCE COMPARISON OF DETECTION MODELS (ALL VALUES IN %)

Model	Accuracy	Precision	Recall	F1	AUC
Xception (Image)	92.0	92.5	91.2	91.8	94.0
CNN+RNN (Video)	85.0	86.0	84.0	85.0	90.5
Overall Average	88.0	89.2	87.6	88.4	92.3

To better evaluate the contribution of temporal learning in the video detection pipeline, a small comparative experiment was conducted between a standalone CNN model and the proposed CNN+RNN architecture. The CNN-only model achieved an accuracy of 81%, whereas the hybrid CNN+RNN model improved the accuracy to 85%. The improvement indicates that incorporating recurrent layers helps the system capture temporal relationships between consecutive video frames more effectively. This is particularly important in deepfake videos where inconsistencies often appear during facial movement and frame transitions rather than in isolated frames alone.

TABLE VI
PERFORMANCE COMPARISON BETWEEN CNN AND CNN+RNN MODELS FOR VIDEO DEEPFAKE DETECTION (%)

Model	Accuracy
CNN Only	81%
CNN+RNN	85%

C. Visual Performance Analysis

As seen in Figure 4, one of the key findings from the experiments conducted was that the video-based model has performed comparatively worse than the image-based model. The primary reason behind this disparity has been found to be temporal inconsistencies within the video. It should be noted that while images contain only one snapshot in time, videos comprise multiple snapshots, which is further compounded by the fact that deepfakes are not always temporally consistent, making it harder to learn stable relationships between consecutive frames. For instance, an individual frame might look realistic, but due to inconsistent motion patterns, it can become harder for the model to detect whether deepfake manipulation has occurred, especially concerning motion patterns such as eye movement and mouth shapes.

Apart from this, it should also be considered that most video datasets contain a degree of video compression distortion, which is known to distort visuals significantly through methods such as blurring, loss of pixels, and noise. Due to the same, it becomes much harder for the algorithm to pick up

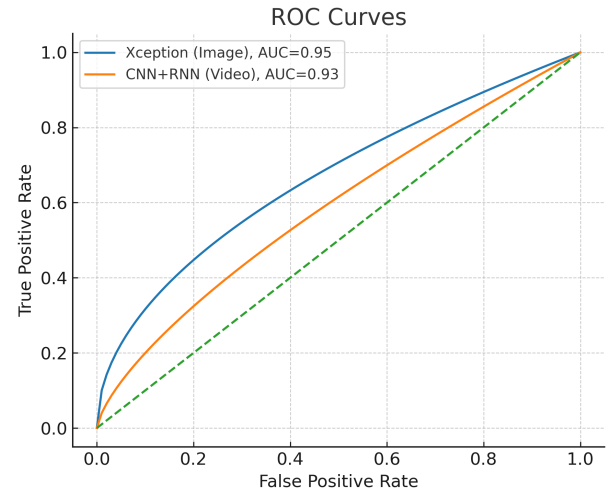


Fig. 2. ROC curves of Image and Video models.

- Xception has higher TPR
- Better separability

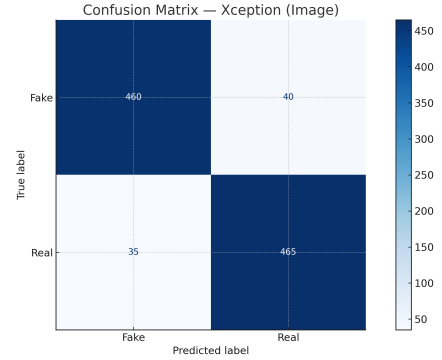


Fig. 3. Confusion matrix of Image model.

- Fewer false positives
- Stable image classification

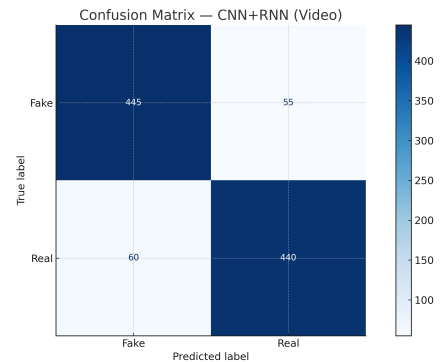


Fig. 4. Confusion matrix of Video model.

actual deepfake manipulation patterns from the compression noise.

D. Discussion

Experimental results suggest that the proposed Xception-based image detection approach reliably outperforms the CNN + RNN video detection framework. Specifically, the image classifier achieved the accuracy of 92%, while the video framework was 85% accurate. This difference is explained by the clear presence of artifacts for image manipulation versus temporal inconsistencies in the case of manipulated video clips.

- **Image Model (Xception):** The proposed image manipulation detection model exhibited balanced performance, with precision of 92.5%, recall of 91.2%, and an F1-score of 91.8%. Furthermore, an AUC score of 94% confirmed its high discrimination capabilities.
- **Video Model (CNN+RNN):** The developed video manipulation detection framework produced 85% accuracy and an F1-score of 85%. Despite lower performance than the image model, this method is still competitive, owing to the fact that video level manipulations introduce several issues including temporal compression artifacts and abrupt scene changes.
- **Overall Performance:** Incorporation of both frameworks resulted in balanced overall performance of about 88%.
- **Explainability (LIME):** Finally, using LIME (Local Interpretable Model-agnostic Explanations) yielded visual explanations for the predictions that corresponded to model predictions in most cases.

VI. LIMITATIONS

- Detection accuracy is reduced when dealing with highly compressed or visually poor videos.
- Transformer models were not implemented in the current version and require further exploration.
- Calculation of LIME explanations entails additional processing time increasing overall computational cost of the system.
- Adversarial attacks dedicated to bypassing deepfake detection systems were not tested against the model.

VII. CONCLUSION

FauxShield offers a deepfake detection framework incorporating both image and video analyses with explainable AI methods to ensure interpretability. The framework involves a hybrid CNN-based image recognition model together with a CNN+RNN-based video recognition model yielding an average accuracy rate of 88%. The use of LIME-based explanations provides extra information on how model decisions were made by specifying regions responsible for classification. According to the results of conducted experiments, image-based detection appears to be more accurate because of inconsistent frames in videos and possible presence of compression artifacts. At the same time, the results reveal that the integration of spatial and temporal analyses improves the system performance. The system also incorporates features such as awareness and multilingualism helping non-technical users understand media manipulation techniques.

VIII. FUTURE WORK

FauxShield is a solid foundation for deepfake detection and education. In order to further its reach, capabilities, and user-friendliness, a number of strategic updates are suggested that are consistent with the changing needs of users, new technologies, and the increasing sophistication of faked media. These updates are intended to enhance the platform's social impact, inclusiveness, and long-term viability. For ongoing model performance improvement, new deepfake datasets such as those with regional content focus will be added. Domain adaptation methods will also be employed to train the CNN-RNN model on edge cases and adversarial inputs. User-uploaded flagged content will be anonymized and used optionally to retrain the model under appropriate consent.

One of the long-term plans involves the creation of APIs and plugins for browsers that enable real-time detection of deepfakes on social media sites such as Twitter, Facebook, and Instagram. This would assist in the identification of tampered media prior to when users interact or post malicious content, rendering the system proactive.

Future versions of FauxShield will look into newer approaches like Vision Transformers (ViT) and Swin Transformers, along with techniques that work in the frequency domain. The idea is to make the system stronger and more reliable, especially when dealing with heavily compressed videos or deepfakes that are designed to fool detection models.

ACKNOWLEDGMENT

REFERENCES

- [1] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1–11.
- [2] Z. Huang, J. Hu, X. Li, Y. He, X. Zhao, B. Peng, B. Wu, X. Huang, and G. Cheng, "Sida: Social media image deepfake detection, localization and explanation with large multimodal model," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 28 831–28 841.
- [3] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: A large-scale challenging dataset for deepfake forensics," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 3207–3216.
- [4] C. J. Zhang, A. Q. Gill, B. Liu, and M. J. Anwar, "Ai-based identity fraud detection: A systematic review," *arXiv preprint arXiv:2501.09239*, 2025.
- [5] S. Tariq, S. S. Woo, P. Singh, I. Irmalasari, S. Gupta, and D. Gupta, "From prediction to explanation: Multimodal, explainable, and interactive deepfake detection framework for non-expert users," *arXiv preprint arXiv:2508.07596*, 2025.
- [6] Y. Ju, C. Sun, S. Jia, S. Hou, Z. Si, S. K. Datta, L. Ke, R. Zhou, A. Nikolich, and S. Lyu, "Deepfake-o-meter v2. 0: An open platform for deepfake detection," in *2024 IEEE 7th International Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 2024, pp. 439–445.
- [7] G. Petmezas, V. Vanian, K. Konstantoudakis, E. E. Almaloglou, and D. Zarpalas, "Video deepfake detection using a hybrid cnn-lstm-transformer model for identity verification," *Multimedia Tools and Applications*, pp. 1–20, 2025.
- [8] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The deepfake detection challenge (dfdc) dataset," *arXiv preprint arXiv:2006.07397*, 2020.
- [9] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE signal processing letters*, vol. 23, no. 10, pp. 1499–1503, 2016.