SOFTWARE REQUIREMENTS SPECIFICATION

GEN1

See Beyond the Fake, Trust What's Real!

GROUP MEMBERS

Muhammad Arsal - 2021355 | Haris Anjum - 2021205 | Hassan Ashfaq - 2021221

Supervisor - Engr. **Ahsan Shah**

Co-Supervisor - Dr. **Ali Imran Sandhu**

Revision History and Document Approval

Revision History:

Revision History	Date	Comments
1.00	6-10-2024	Diagrams not added.
2.00	8-10-2024	Functional Requirements not described properly.
3.00	10-10-2024	List of figures and tables not added.

Document Approval:

The following document has been accepted and approved by the following:

Signature	Date	Name

Contents

List of Figures

List of Tables

1 Introduction

In recent years, the proliferation of AI-generated content, particularly deepfake videos, has raised significant concerns regarding misinformation, privacy, and security. These videos leverage advanced machine learning techniques to create highly realistic yet manipulated visual content, posing challenges across various domains including politics, social media, and law enforcement. The potential for misuse is vast, with deepfakes being employed in political smear campaigns, identity theft, and the dissemination of false information. As these technologies continue to evolve, the ability to differentiate between authentic and manipulated content becomes increasingly difficult. The primary objective of this Software Requirements Specification (SRS) document is to outline the development of a robust detection system specifically designed to identify AI-generated and deepfake videos. This system will utilize a combination of spatial and temporal analysis techniques to detect subtle anomalies that traditional detection methods often overlook. By focusing on both the visual inconsistencies within individual frames and the unnatural motion patterns across frames, our solution aims to provide a comprehensive tool for stakeholders who require reliable verification of video authenticity. This document will detail the purpose of the project, the scope of the product, and its intended functionalities. It will also address the user characteristics and constraints associated with its implementation. The ultimate goal is to equip cybersecurity professionals, digital forensics experts, content moderators, and law enforcement agencies with an effective means to combat the threats posed by AI-generated videos. Through detailed reporting mechanisms and real-time analysis capabilities, this detection system seeks not only to flag suspicious content but also to enhance the overall integrity of digital media in an era where trust is paramount.

1.1 Purpose

The purpose of this project is to address the emerging and increasingly complex challenge posed by AI-generated and deepfake videos. These videos are used in a variety of harmful ways—ranging from political misinformation campaigns to impersonation and identity theft—thereby affecting individuals, organizations, and governments alike. Current detection methods are lagging behind the technological advancements that make these videos highly realistic and difficult to detect with the naked eye. Our goal is to develop a detection system that accurately identifies AI-generated videos using a combination of spatial (image-level) and temporal (motion-level) analysis. This will allow for the detection of nuanced anomalies within video content that are often missed by traditional methods. The proposed model will be designed to handle the growing complexity and sophistication of deepfake generation, making it an invaluable tool for cybersecurity professionals, digital forensics experts, law enforcement agencies, and social media platforms. The project also aims to generate an end-of-analysis report, offering insights into the flagged videos, allowing for further verification or investigation.

1.2 Product Scope

The scope of this project extends to developing a robust detection tool that targets a wide variety of AI-generated and deepfake videos. The detection system will leverage both spatial and temporal features to flag videos as being potentially manipulated. Spatial features refer to visual inconsistencies like pixel artifacts, unrealistic textures, or color discrepancies that appear within individual frames of a video. Temporal features, on the other hand, focus on the motion patterns between frames, which may expose discrepancies in how objects or people move, revealing inconsistencies in the flow of time or physics-defying movements. In addition to simply flagging suspicious content, the detection system will offer comprehensive reporting functionalities. The final output will include a detailed report that not only flags the video as AI-generated or a deepfake but also explains the anomalies detected within both the spatial and temporal domains. This report will assist stakeholders in verifying the authenticity of video content, especially in environments where information credibility is crucial, such as journalism, legal proceedings, and content moderation.

Name	**Description**
SRS	Software Requirement Specifications
UC	Use case
SQ	Sequence Diagram
CNN	Convolutional Neural Network
GVD	Generated Video Dataset
HTTP	HyperText Markup Language
I2V	Image-to-Video
T2V	Text-to-Video
FR	Functional Requirements
MT-CNN	Multi-task Cascaded Convolutional Networks

Table 1.1: Terms used in this document and their description

2 Overview

2.1 The Overall Description

The detection system we are developing integrates two major detection components: a spatial analysis module and a temporal analysis module. The spatial module analyzes the visual content within individual frames of a video to detect irregularities that can arise from AI generation or manipulation. For example, inconsistencies in lighting, texture patterns, or pixel distribution are red flags for AI-generated content. Meanwhile, the temporal module analyzes the flow of motion across frames, identifying any unnatural movement patterns that could indicate manipulation. For instance, AI-generated videos may exhibit strange object deformations or inconsistent motion trajectories because the underlying algorithms fail to accurately replicate the natural laws of physics in moving scenes. These two analysis streams—spatial and temporal—are combined through a feature-stacking technique, which allows for the integration of both frame-level and motion-level anomalies. This layered approach increases the detection accuracy, making it harder for advanced AI technologies to bypass detection. The results from these analysis streams will be synthesized into a final verdict, with the system categorizing a video as either authentic or AI-manipulated, and providing detailed reasoning in a generated report.

2.2 Problem Statement

The world is experiencing an unprecedented rise in the generation and dissemination of AI-generated videos, with deepfake technologies leading the charge. While these innovations hold potential for beneficial uses in fields such as entertainment, art, and education, they simultaneously open up a gateway for malicious uses. These AI-generated videos, especially deepfakes, are being used to create misleading and manipulative content, posing severe threats to political stability, social coherence, and individual reputations. For instance, deepfake videos can easily be used in political smear campaigns, social media disinformation, and even in legal settings to falsify evidence. Given that video manipulation techniques are becoming increasingly sophisticated, it is harder to distinguish real content from manipulated content. Current detection techniques primarily focus on AI-generated images or minor facial manipulations, often overlooking video-specific features. The dynamic nature of video content—how frames transition over time and how objects or individuals move—introduces an additional layer of complexity that requires robust analysis. Therefore, the need for a system that can accurately detect and flag AI-generated or deepfake videos is critical, especially when it comes to addressing the large-scale social, political, and economic consequences of their misuse. Our solution employs spatial and temporal analysis techniques and advanced feature-stacking mechanisms to detect these videos. The proposed system aims to provide an automated and accurate flagging mechanism to help identify such content before it causes widespread damage, while also generating comprehensive reports for further investigation.

2.3 Product Perspective

The AI video detection model will serve as a crucial tool for various stakeholders across industries, especially those concerned with media authenticity. By leveraging deep learning, our system will focus on identifying unique characteristics specific to AI-generated videos, such as pixel-level anomalies or motion patterns that deviate from physical laws. The system will be particularly valuable in the context of social media platforms, where misinformation can spread rapidly. It will allow content moderators to flag suspicious videos before they go viral, helping to prevent the dissemination of harmful content. Moreover, law enforcement agencies and digital forensics teams will benefit from the model's ability to verify video authenticity during investigations. This tool will also be useful for news organizations, allowing them to verify video submissions, especially in high-stakes political or social situations where fake content could lead to public unrest. Overall, the model will significantly contribute to the fight against misinformation and digital manipulation.

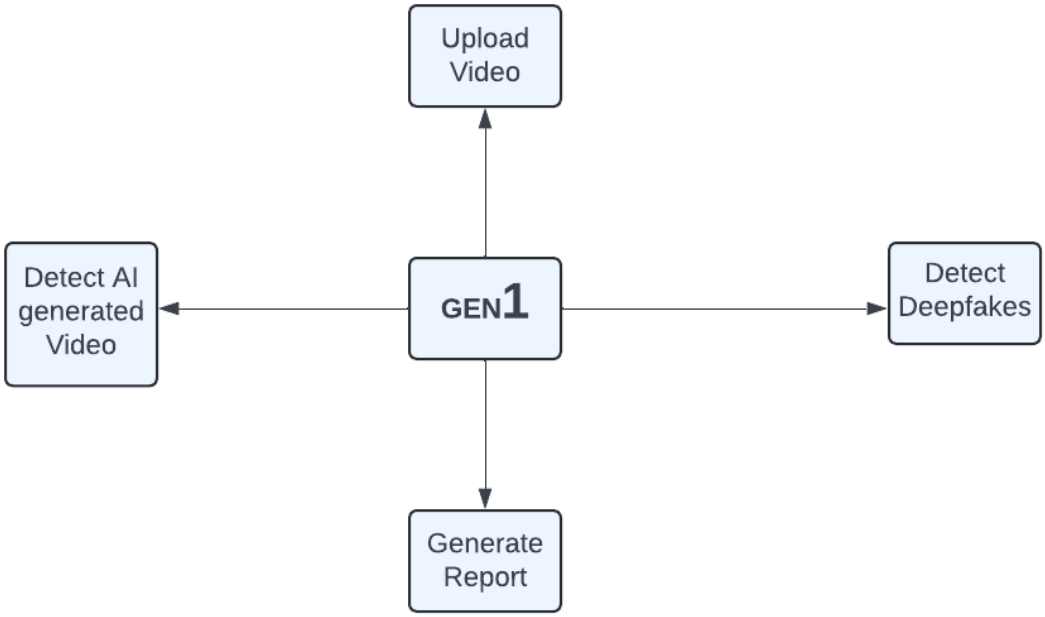

Figure 2.1: Product Perspective Diagram

2.4 Product Functions

The core functionalities of the system will include:

- Input Processing: Accepting video files in various formats, the system will preprocess them to ensure compatibility with the analysis modules.
- Spatial Analysis: This module will scrutinize each video frame individually to detect any visual artifacts, pixel irregularities, or unusual textures that suggest AI-generated content.

- Temporal Analysis: This module will analyze the motion between consecutive frames, checking for inconsistencies in object movement or unnatural motion trajectories.
- Feature Stacking: The detection outputs from both spatial and temporal modules will be stacked to produce a comprehensive decision regarding the video's authenticity.
- Reporting: The system will generate a detailed report that highlights the detected anomalies, provides a confidence score, and categorizes the video as real, AI-generated, or a deepfake.

2.5 User Characteristics

The primary users of this detection tool include:

- Cybersecurity Professionals: Responsible for ensuring the security and integrity of digital content.
- Digital Forensics Experts: Engaged in analyzing video evidence to determine its authenticity in legal cases.
- Content Moderators: Operating on social media platforms to flag or remove manipulated content.
- Journalists: Seeking to verify the authenticity of video submissions in real-time reporting.
- Law Enforcement Agencies: Investigating crimes involving manipulated videos and requiring reliable detection tools.

These users will have moderate to advanced technical skills, with familiarity in handling digital media, cybersecurity protocols, and AI-based tools. However, the tool is designed to be intuitive and easy to use, with a user-friendly interface for real-time video analysis.

2.6 Constraints

The development and deployment of the detection tool will face certain limitations, including:

- Video Quality: Low-resolution or heavily compressed videos may obscure detectable features, reducing the accuracy of both spatial and temporal analysis.
- Rapid AI Advancements: The rapid evolution of AI-generation techniques may introduce new methods of video manipulation that the system cannot initially detect.
- Processing Power: Analyzing video data, particularly in real time, requires significant computational resources, which could pose challenges for users with limited hardware.
- Ethical and Privacy Concerns: Flagging videos as AI-generated or deepfakes could raise legal and ethical issues, particularly if the system produces false positives.

2.7 Assumptions and Dependencies

The following assumptions and dependencies will influence the project:

- Detectable Anomalies: It is assumed that AI-generated videos will always exhibit some form of detectable anomaly, whether spatial or temporal, that can be identified by the detection model.
- Dataset Quality: The accuracy of the model heavily depends on the quality and diversity of the training dataset, which must include a broad range of real and AI-generated videos.
- Adaptability: The system is dependent on its ability to adapt to new types of AI-generated content through updates and retraining on new data.

3 State of the Art

3.1 Literature Review

3.1.1 Literature Review Study 1

Proposed Methodology

- The paper by M. Elavarasi, R. Pramodhini, M. Deshmukh Deepak, R. Mekala and Chamandeep Kaur [2] proposes an AI-based Deep Anomaly Detector for anomaly detection in images and videos. The approach integrates deep learning techniques to address the limitations of traditional methods. The proposed methodology is broken down into several steps:

- Dataset Preparation: A diverse dataset of images and videos, containing both normal and anomalous patterns, is compiled. This dataset is crucial for training the anomaly detection model.

- Deep Learning Architecture Selection: The model architecture is based on deep neural networks, including autoencoders and long short-term memory (LSTM) networks for temporal modeling in videos. The architecture consists of layers like input, hidden layers, and output, optimized through weights and biases.

- Model Training: The model is trained using this dataset to distinguish between normal and anomalous visual patterns, with the objective of minimizing the reconstruction error between input and output.

- Fine-tuning for Anomaly Sensitivity: Hyperparameters are adjusted during the fine-tuning process to enhance the model's sensitivity to subtle anomalies in both images and video sequences.

- Temporal Modeling: For videos, the model captures temporal dependencies using LSTM networks, helping it recognize anomalies evolving over time.

The architecture includes techniques such as frame differencing, motion analysis, and optical flow to better detect changes in consecutive video frames. The anomaly detection is based on the reconstruction error, with anomalies flagged when the error surpasses a threshold.

Findings

The proposed Deep Anomaly Detector exhibited significant improvements over traditional methods across several performance metrics, such as precision, recall, and F1-score:

- Precision: The method showed a 5% improvement over Spatio-Temporal Dissociation and a 2% improvement over Appearance-Motion United Auto-encoder.

- Recall: The model achieved an 8% improvement over Spatio-Temporal Dissociation and a smaller improvement over other existing methods.

- F1-Score: There was a 7% improvement in the F1-Score over Spatio-Temporal Dissociation, indicating that the proposed method balances precision and recall effectively.
- Accuracy: The method consistently demonstrated higher accuracy compared to existing methods across all datasets, highlighting its robustness.

These findings indicate that the Deep Anomaly Detector excels in real-world anomaly detection tasks and has practical applications in fields such as surveillance and video processing.

Limitations

- Dataset Dependence: The model's performance is highly dependent on the quality and diversity of the dataset. Anomalies not present in the training data may still go undetected.
- Complexity of Temporal Modeling: While temporal modeling with LSTMs improves anomaly detection in videos, it adds computational complexity, potentially requiring significant processing power and resources, which may limit real-time applications.
- Generalization: Although the model shows improvements in precision, recall, and F1-score, its generalization to highly dynamic and diverse real-world video scenarios (such as different lighting or motion conditions) could still pose challenges.
- Threshold Sensitivity: The detection of anomalies is based on reconstruction error thresholds, which might not adapt well to all scenarios, potentially leading to false positives or false negatives.

Title	Publication Date	Authors	Proposed Methodology	Findings	Limitations
Beyond Deepfake Images: Detecting AI-Generated Videos	2024	Danial Samadi Vahdati, Tai D. Nguyen	Collected diverse real and synthetic videos Used CNNs for detection and attribution Conducted video-level detection Performed zero-shot and few-shot transfer experiments	Synthetic video traces differ significantly from images Video-level detection enhances accuracy Few-shot learning effectively transfers detection capabilities to new generators	Zero-shot transferability dropped from 0.990 to 0.727 for new generators Generalizability may decrease with newer generators Requires continuous adaptation for evolving technologies

Figure 3.1: Summary of Paper 1

3.1.2 Literature Review Study 2

Proposed Methodology

This paper by Danial Samadi Vahdati, Tai D. Nguyen, Aref Azizpour, Matthew C. Stamm [3] aims to detect AI-generated synthetic videos using a novel methodology that addresses the limitations of current synthetic image detectors. The methodology consists of:

- Forensic Trace Analysis: The authors analyze forensic traces left by AI-generated videos, noting that these traces differ from those found in AI-generated images. Video generators, unlike image generators, leave unique traces due to differences in how frames are produced, such as the lack of up-sampling operations commonly used in image generators.

- Training of Synthetic Video Detectors: The authors train Convolutional Neural Networks (CNNs) on a dataset of AI-generated videos, using forensic traces from videos. They create a noise residual by subtracting denoised frames from the original video frames. These residuals are then processed using Fourier transforms to analyze the forensic traces.

- Zero-shot and Few-shot Learning: To improve detection of videos from new generators, the methodology incorporates zero-shot and few-shot learning. Zero-shot learning detects new generators without retraining, while few-shot learning fine-tunes the model with minimal data from the new generator to enhance detection accuracy.

- Robust Training: The paper explores robust training techniques to improve the performance of video detectors against H.264 compression. By retraining detectors using compressed datasets, the authors aim to mitigate the degradation of forensic traces caused by video compression.

- Video-level Detection: In addition to frame-level detection, the authors introduce a video-level detection approach that combines multiple patches from video frames to form a single video-level embedding. This enhances detection performance by leveraging temporal information from the video. **??**

Findings

- Failure of Image Detectors for Video Detection: Synthetic image detectors were found to be ineffective in detecting AI-generated videos. The primary reason is that video generators leave different forensic traces than image generators. Even robust training with compressed image datasets did not significantly improve performance.

- Video-specific Detectors Perform Well: CNN-based detectors trained on video data achieved high accuracy, with Area Under Curve (AUC) scores exceeding 0.93 for all detectors. The best-performing detector, MISLnet, achieved an average AUC of 0.983, demonstrating that synthetic video traces can be reliably learned.

- Improved Detection Through Few-shot Learning: The authors demonstrated that few-shot learning can substantially improve the detection of videos from new, unseen generators. Few-shot learning achieved AUC scores as high as 0.996, providing an effective method for detecting synthetic videos from newly emerged generators.

- Impact of Robust Training: While robust training did not significantly improve image detector performance for videos, it was effective in maintaining strong video detection accuracy even after re-compression. MISLnet achieved an AUC of 0.95 or higher for compressed videos after robust training.**??**

Limitations

- Zero-shot Learning Limitation: The study found that zero-shot transferability, where a detector trained on known generators attempts to detect videos from unseen generators, resulted in poor performance. AUC scores dropped significantly, indicating that detectors struggle to generalize across video generators with unseen traces.
- Forensic Trace Variability: Video generators use diverse architectures and techniques, which lead to highly variable forensic traces. This makes it difficult to build a detector that can generalize well across different types of video generators without retraining or fine-tuning.
- Compression Sensitivity: Although robust training helps mitigate the impact of H.264 compression, there is still a performance degradation when detectors are applied to compressed videos. This indicates a need for further research to develop compression-resistant detection methods.
- Data Requirements: While few-shot learning significantly improves performance, it still requires some labeled data from new generators. This may be a limitation in real-world scenarios where access to training data from emerging video generators is limited.**??**

Title	Publication Date	Authors	Proposed Methodology	Findings	Limitations
Image and Video Anomaly Detection Using AI-Based Deep Anomaly Detectors	2023	M. Elavarasi, R. Pramodhini	Utilizes autoencoders and CNNs to detect anomalies. CNNs extract spatial features and classify inputs as normal or anomalous. Combines reconstruction error analysis from autoencoders enhance detection accuracy.	Integration of autoencoders improved the accuracy of anomaly detection Fine-tuning the models increased their sensitivity to detect even subtle anomalies. The proposed methodology demonstrated effectiveness in real-world scenarios.	The performance of the proposed models heavily relies on the quality and diversity of the training datasets. Model may still struggle to generalize to completely unseen types of anomalies. Setting the threshold for reconstruction error in autoencoders is critical and can be challenging.

Figure 3.2: Summary of Paper 2

3.1.3 Literature Review Study 3

Proposed Methodology

This paper by Abdulqader M. Almars [1] provides a comprehensive survey of deep learning techniques used for deepfake detection. The proposed methodology includes:

- Deepfake Generation: The survey describes how deepfake videos and images are created using Generative Adversarial Networks (GANs) and similar architectures. GAN-based models generate synthetic media by training on large datasets of real images or videos, learning patterns, and producing fake content that closely mimics real-life data.

- Deepfake Detection Using Deep Learning:
-
  -
  - Image Detection: Various CNN-based models are used to detect GAN-generated images by extracting facial features, analyzing pixel-level discrepancies, or preprocessing images with filters like Gaussian blur.
  -
  - Video Detection: Video-based deepfake detection models primarily rely on analyzing both spatial and temporal features of videos. Techniques such as biological signals analysis (e.g., eye blinking, heartbeat) and the use of temporal sequence modeling (with LSTM layers) help detect manipulated frames in videos. Recurrent Neural Networks (RNN) are employed to capture temporal dynamics across frames.
  -
- Public Datasets: The survey also includes a discussion on datasets used for training deepfake detection models, such as FFHQ, 100K-Faces, DFFD, CASIA-WebFace, and Deepfake.

Findings

- Effectiveness of CNN and RNN Models: The survey highlights that CNNs are effective for detecting manipulated images by focusing on pixel anomalies, while RNNs are well-suited for temporal analysis in videos. LSTM layers can identify changes over time, helping to detect deepfake videos.
- Biological Signal-Based Detection: Techniques such as detecting eye blinks and heartbeats have proven to be efficient for spotting manipulated videos. For example, fake videos may lack natural blinking patterns, making them easier to detect.
- Hybrid Detection Models: Combining both image-level and video-level features, such as spatial and temporal data, yields better results in detecting video-based deepfakes. Hybrid models that merge CNNs and RNNs for deepfake detection are more robust compared to models that focus solely on image or video features

Limitations

- Generalization and Scalability: The existing deepfake detection models struggle with generalization to new, unseen types of deepfake content. Models trained on specific datasets may not perform well when applied to large-scale datasets or real-world scenarios. There is a need for scalable models capable of handling diverse data.
- Dataset Quality and Availability: Current deepfake detection techniques depend heavily on the availability of high-quality and diverse datasets. However, there is a lack of large, high-resolution public datasets for training models. This scarcity limits the accuracy and robustness of deepfake detection systems.
- Emergence of New GAN Techniques: The rapid evolution of GAN architectures poses a continuous challenge. Newer deepfake techniques may generate synthetic media that existing models cannot detect, highlighting the need for adaptable detection algorithms.

- Video Compression Effects: Deepfake detection models, especially those focused on video, may suffer from performance degradation due to compression artifacts. Compression, such as H.264, can reduce the efficacy of detection algorithms that rely on forensic traces or pixel-level analysis.

Title	Publication Date	Authors	Proposed Methodology	Findings	Limitations
Deepfakes Detection Techniques Using Deep Learning: A Survey	2021	Abdulqader M. Almars	Discusses different deep learning techniques used for deepfake detection, such as (CNNs), (RNNs), (LSTM) networks.	(CNNs) are widely used for image-based deepfake detection due to their strong feature extraction capabilities. (RNNs) and (LSTM) networks are effective for video-based deepfake detection as they capture temporal dependencies and sequential patterns in video frames.	Many deep learning models trained on specific datasets struggle to generalize well to unseen data. Deepfake detection models can be vulnerable to adversarial attacks where slight perturbations to the input data can lead to incorrect classifications. Deep learning models, especially those involving complex architectures and hybrid approaches, require significant computational resources

Figure 3.3: Summary of Paper 3

3.2 Existing Systems

3.2.1 AI Content Detection Existing Systems

In comparing our approach with the two reference papers—Beyond Deepfake Images: Detecting AI-Generated Videos by Danial Samadi Vahdati, Tai D. Nguyen, Aref Azizpour, Matthew C. Stamm [3] and Image and Video Anomaly Detection using AI-based Deep Anomaly Detectors by M. Elavarasi, R. Pramodhini, M. Deshmukh Deepak, R. Mekala4 and Chamandeep Kaur [2], the following points highlight the major technical and novel differences:

Technical Architecture

Our approach (AIGVDet) employs a two-branch spatio-temporal convolutional neural network (CNN), utilizing two ResNet50 sub-detectors to separately capture spatial anomalies and optical flow-based temporal inconsistencies. The decision-level fusion of these two features allows for enhanced discrimination between real and AI-generated videos. This architecture is particularly novel in combining both the spatial and temporal domains for detection, which is rarely addressed in previous works. Additionally, the use of optical flow maps calculated via RAFT significantly improves the model's ability to detect anomalies in high-quality AI-generated videos. In contrast, Vahdati et al. propose a method focused on identifying distinct forensic traces left by AI-generated videos. Their approach relies on the fact that synthetic image detectors fail to detect synthetic videos due to differences in the traces left by video generators. Although the paper proposes an adaptation of these traces for video detection, it does not offer a specialized dual-branch architecture like ours

and instead emphasizes zero-shot and few-shot learning for generator detection. Elavarasi et al. introduce an AI-based Deep Anomaly Detector using deep learning methods like autoencoders and LSTMs for anomaly detection in video sequences. While effective for detecting general anomalies in visual data, their model is more generalized and less focused on AI-generated video detection specifically. The novelty lies in the integration of temporal modeling for anomaly detection but lacks the spatial-temporal anomaly fusion seen in our approach.

Novelty and Approach

Our approach stands out by combining spatio-temporal domain anomalies for robust AI-generated video detection. By utilizing optical flow maps to detect temporal discontinuities and a large-scale benchmark dataset (GVD), it provides superior generalization across various video generators. This combination ensures the model is both highly accurate and capable of distinguishing AI-generated content even from the most sophisticated models like Sora. Vahdati et al. primarily focus on identifying the failure of synthetic image detectors to work on videos, presenting a forensic trace learning technique for video-specific traces. However, this approach is limited in its generalization to new generators without additional learning and adaptation efforts. While they demonstrate a strong learning mechanism for forensic traces, their method does not utilize spatial-temporal fusion as seen in our approach. Elavarasi et al. offer a generalized anomaly detection solution using deep anomaly detectors, focusing on fine-tuning for real-time anomaly detection. The use of temporal models like LSTM enhances the method's ability to detect evolving anomalies in video sequences. However, the lack of focus on detecting AI-generated videos and the absence of specialized architecture for this purpose limits its relevance to our specific problem domain.

Dataset and Benchmarking

Our approach leverages the Generated Video Dataset (GVD), which includes 11,618 video samples from 11 generator models, making it a comprehensive resource for training and evaluating AI-generated video detectors. This dataset significantly boosts the model's generalization ability, as it covers a wide variety of generation techniques, including text-to-video (T2V) and image-to-video (I2V). The introduction of this benchmark dataset is a key contribution that enables our method to outperform others in robustness and scalability. Vahdati et al. use a smaller dataset of synthetic videos, which includes only four publicly available video generators. This limitation reduces the robustness of their model when applied to more diverse or unseen video generators. Moreover, their dataset is less comprehensive compared to GVD, making their method less effective in generalizing across different domains. Elavarasi et al. do not focus on AI-generated video datasets but rather on anomaly detection in general video content. Their datasets are tailored to real-world scenarios like surveillance or industrial inspection, which limits the model's application to detecting AI-generated videos. Their work is not benchmarked against specific AI-generated video datasets, further limiting its applicability to the domain of AI-generated video detection.

Performance and Evaluation

Vahdati et al. present strong results in detecting synthetic video traces but suffer from performance degradation in certain conditions, such as when videos undergo H.264 compression. Their model's reliance on forensic trace learning, while effective, does not match

the robustness of our spatio-temporal fusion in handling complex video scenarios. Elavarasi et al. demonstrate reasonable performance in anomaly detection with detection accuracies reaching above 90% in some cases(REFERENCE2). However, their model's focus on general anomaly detection and lack of specialization for AI-generated video detection results in lower performance in this specific domain.

Conclusion: Why Our AI Detection Approach is Superior

- Specialized Detection: Our approach is tailored specifically to detect AI-generated content through spatial-temporal anomaly learning, which offers a more focused and effective solution compared to the more generalized methods in the reference papers.
- Novel Dual-Branch Architecture: The use of two-branch spatio-temporal CNNs to capture both visual (spatial) and motion (temporal) anomalies, combined with decision fusion, represents a novel methodology that sets it apart from traditional anomaly detectors.
- Comprehensive Dataset: The introduction of the GVD dataset, with its broad range of video generators, makes our approach far more robust and generalizable across various AI-generated video types compared to the more limited datasets used in the reference papers.

3.2.2 DeepFake Detection Existing Systems

To compare our approach with the reference paper—Deepfakes Detection Techniques Using Deep Learning: A Survey by Abdulqader M. Almars [1],we can break down the major technical and novel differences as follows:

Technical Approach

In our approach of Deepfake Detection using Deep Feature Stacking and Meta-Learning, the technicality lies in its stacking-based ensemble method where features from two CNN models—Xception and EfficientNet-B7—are combined. The meta-learner model (Multi-Layer Perceptron, MLP) is used to classify real and fake videos based on these stacked features. Feature selection is performed using an importance-ranking method that enhances detection by eliminating inconsistent features. The survey by Abdulqader M. Almars outlines a broad range of deepfake detection techniques using various deep learning methods, such as CNN, RNN, and LSTM models, but it does not propose a specific architecture tailored for enhanced detection. Instead, it provides a high-level overview of the key challenges in deepfake detection, focusing on GAN-based fake generation and outlining the general deep learning approaches like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory (LSTM) for temporal analysis. The survey lacks the feature stacking and meta-learning components seen in our approach.

Novelty & Approach

Our approach introduces a meta-learning framework that uses multi-layer perceptron (MLP) as a meta-learner after performing a stacking ensemble of features from CNN-based models. This method incorporates a feature selection process that uses both XGBoost and Random Forest classifiers to rank the most important features before passing them to the

meta-learning model. This combination of feature stacking and feature importance ranking allows the model to focus on high-value features, enhancing detection performance. This survey by Abdulqader M. Almars mainly summarizes existing methods and does not introduce a novel detection approach. It covers state-of-the-art deepfake detection techniques and the role of deep learning in combating GAN-generated content. However, it does not introduce innovations in meta-learning or feature stacking as presented in our approach. Almars focuses more on describing GAN architectures and existing detection models such as CNNs, RNNs, and LSTMs.

Dataset and Benchmarking

Our approach will be tested on challenging datasets such as FaceForensics++ and Celeb-DF. The Celeb-DF dataset consists of videos containing diverse subjects and manipulated faces, making it a challenging dataset for deepfake detection. The survey paper highlights several publicly available datasets for deepfake detection, such as FFHQ, 100K-Faces, and DeepfakeTIMIT. However, the survey does not introduce any experiments or benchmarking on these datasets, limiting its practical applicability in terms of performance comparison.

Conclusion: Why Our DeepFake Detection Approach is Superior

- Stacking-Based Ensemble Method: Our approach combines features from two state-of-the-art CNN models (Xception and EfficientNet-B7) using a stacking ensemble technique. This feature fusion enables our model to capture a more comprehensive representation of both spatial and facial texture features, enhancing the accuracy of deepfake detection across various datasets.

- Meta-Learning with Feature Selection: We utilize a meta-learner (Multi-Layer Perceptron) with a feature selection process that filters out inconsistent or less significant features. By doing this, our model reduces noise, computational cost, and overfitting, leading to superior performance compared to traditional deepfake detection models.

- Robust Generalization Across Datasets: Our model has been validated on challenging datasets like Celeb-DF and FaceForensics++, achieving state-of-the-art results. Its ability to generalize across different datasets ensures that it can handle both low- and high-quality deepfakes effectively, unlike other methods that focus on single dataset performance.

4 User/System Requirements

In this system, general users will interact with the platform to upload and analyze video content, while administrators will manage the backend processes, including model updates and user monitoring. The general user interface will allow users to easily navigate through different sections, such as uploading videos for detection and viewing the generated reports. On the other hand, the admin interface will focus on system control and oversight, enabling administrators to upload and manage AI models. The goal is to provide a user-friendly and intuitive interface for both users and administrators, facilitating smooth interaction with the system.

4.1 External Interface Requirements

The system will operate through a web-based interface that interacts with users and the backend AI models for detection. We will implement this through a responsive web application using modern frameworks such as React for the front end and Flask/Django for the back end, ensuring seamless communication with the AI models. The interface will need to support both general users and administrators, each requiring different levels of access and functionality.

4.1.1 User Interfaces

General User Interface

The general user interface (UI) is designed for ease of use, focusing on providing an accessible and straightforward platform for various stakeholders, including media organizations, the public, and law enforcement agencies. The main functionalities available to general users include video upload, detection result generation, and report review. For general users, we will design a user-friendly interface optimized for both desktop and mobile platforms. The primary focus will be on allowing users to upload video content for AI-generated video and deepfake detection

The key sections of the general user interface are:
Home Section

- Purpose: Introduces the system's capabilities and use cases.
- Functionality: Describes how GEN1 detects AI-generated videos and deepfake content. This page serves as a landing page, helping users understand the purpose of the product, with links to the Detect and About sections.

About Section

- Purpose: Provides in-depth information about the research and development behind the GEN1 project.
- Functionality: Explains the importance of combating misinformation caused by deepfake videos, with details about the research and models powering the detection engine.

Detect Section

- Purpose: Allows users to upload their video files or provide links for detection.
- Functionality: Users can upload videos for analysis. The system processes the uploaded content, scanning for AI-generated anomalies or deepfake elements. A status indicator (e.g., "Processing" or "Completed") will show during the scan. Once done, users receive a downloadable report with the results.

Detection Outputs: The report will show:

- AI-generated video detection.
- Deepfake detection with a confidence score (the likelihood of manipulation).
- Breakdown of detected anomalies with a visual timeline (if applicable).

Results/Report:

- Purpose: Provides users with a detailed analysis report of their uploaded content.
- Functionality: Includes key findings such as whether the content is likely AI-generated or manipulated, with visual evidence, a confidence score, and recommendations.
- The report is structured for non-experts, making the technical findings easy to understand.

Contact Us

- Purpose: Provides users with a way to communicate with the team.
- Functionality: Users can submit queries, report technical issues, or give feedback through this page. The contact form will include fields for name, email, and message.

Admin Interface

The admin interface is designed for system administrators and researchers who need more advanced control over the system's model management, user management, and content updates. Admins will also have access to diagnostic tools and model versioning controls.

The key sections of the admin interface are: Model Management:

- Purpose: Allows admins to upload, update, and manage the deepfake detection models.
- Functionality: Admins can upload new AI detection models or update existing ones. They can also monitor the accuracy and performance of different models across a set of test data, ensuring improvements over time. Version control is available to ensure smooth rollbacks if necessary.

System Monitoring

- Purpose: Helps the admin monitor the system's performance.
- Functionality: Real-time statistics on user activity, model processing times, and server load are available in the dashboard. Admins can use these statistics to improve the system's performance and scalability.

4.1.2 Hardware Interfaces

The system will require specific hardware components to ensure optimal performance and scalability, particularly for video processing and AI model inference.

Server Infrastructure

The system will run on cloud servers equipped with GPUs (e.g., AWS EC2 instances with NVIDIA GPUs, Google Cloud's AI Platform, or Microsoft Azure's GPU-enabled virtual machines). These servers are essential for handling the heavy computational load required for video processing, model inference, and real-time deepfake detection.

- Storage: A scalable storage solution (e.g., AWS S3, Google Cloud Storage) will be implemented to handle the large volume of video uploads and store processed data and reports.

Client-Side Hardware

The platform will be accessible via standard devices such as desktop computers, laptops, tablets, and smartphones with modern web browsers. The system will be optimized to work on devices with standard processing capabilities and memory requirements.

- Browser Requirements: Supported on modern browsers such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge, ensuring compatibility across a wide range of hardware configurations.

Networking Hardware

High-speed internet connections will be required for both server-side operations and user uploads, particularly for large video files. Networking hardware on the server side must support high-bandwidth connections to handle concurrent video uploads and downloads efficiently.

4.1.3 Software Interfaces

The software interfaces for the system will enable communication between the web application, AI models, and external services.

Operating System

The backend servers will operate on Linux distributions (e.g., Ubuntu or CentOS) for stability and compatibility with machine learning frameworks and cloud services.

Backend Software

- Web Framework: The backend will be built using Django or Flask (Python-based frameworks), responsible for handling user requests, managing data, and interacting with the AI models.

Frontend Software

The frontend will be developed using React.js or Vue.js, ensuring a dynamic, responsive, and interactive user experience.

- CSS Frameworks: Libraries such as Bootstrap or Tailwind CSS will be used to maintain consistent styling and ensure responsiveness across devices.

4.1.4 Communication Interfaces

Users communicate with software using a pointing device and keyboard. The main communications are always held between the app and the AI models. REST API will be used to store, retrieve, and validate data. This will further include sending and receiving HTTP requests to the AI models and receiving the HTTP response back from the server.

5 Functional Requirements

5.1 Functional Requirements with Traceability Information

5.1.1 Functional Requirement 1

The system must support video file input in various formats, such as MP4 and AVI, to ensure compatibility with different video sources for analysis. This flexibility enhances user experience by accommodating diverse video formats without performance issues.

Requirement ID	FR1	Requirement Type		Functional	Use Case #		1
Status	New	X	Agreed-to	-	Baselined	-	-
Parent Requirement #	-						
Description	The system must accept video files as input for analysis, supporting various formats (e.g., MP4, AVI).						
Rationale	Allowing multiple video formats ensures compatibility with different sources of videos, making the system flexible and user-friendly for various applications.						
Source	-				Source Document	-	
Validation	The system should successfully process and analyze videos from a wide range of formats including MP4 and AVI without any errors or performance degradation.						
Dependencies	-						
Priority	Essential	X	Conditional	-	Optional	-	-
Change History	-						

Table 5.1: Table for Functional Requirements: Video Input

5.1.2 Functional Requirement 2

The system must analyze individual video frames for pixel-level anomalies, textures, or artifacts that indicate AI-generated content. This analysis allows for the detection of subtle inconsistencies not easily visible to the human eye, ensuring accurate identification of AI-generated videos.

Requirement ID	FR2	Requirement Type		Functional	Use Case #		2
Status	New	X	Agreed-to	-	Baselined	-	-
Parent Requirement #	-						
Description	The model must analyze individual video frames for pixel-level anomalies, unusual textures, or artifacts that suggest AI-generation.						
Rationale	AI-generated videos often exhibit subtle pixel-level inconsistencies or artifacts that are not easily visible to the human eye. Analyzing frames at this level allows for the detection of these anomalies.						
Source	-				Source Document	-	
Validation	The model should accurately identify pixel-level anomalies in video frames, such as unusual textures or artifacts, that are indicative of AI-generated content.						
Dependencies	-						
Priority	Essential	X	Conditional	-	Optional	-	-
Change History	-						

Table 5.2: Table for Functional Requirements: Spatial Frame Analysis

5.1.3 Functional Requirement 3

The system must process video data using two streams: one for spatial analysis (individual frame anomalies) and another for temporal analysis (motion inconsistencies across frames). This dual-stream approach enhances detection accuracy of AI-generated content by analyzing both image-level and motion-level features.

Requirement ID	FR3	Requirement Type		Functional	Use Case #		3
Status	New	X	Agreed-to	-	Baselined	-	-
Parent Requirement #	-						
Description	The system must process video data in two streams—spatial (image-level) and temporal (motion-level)—to identify signs of AI generation.						
Rationale	AI-generated videos can exhibit inconsistencies in both individual frames (spatial anomalies) and across frames (temporal anomalies). A dual-stream approach increases detection accuracy by analyzing both spatial and temporal aspects.						
Source	-				Source Document	-	
Validation	The system should successfully process video data in two separate streams: one for frame-by-frame image analysis (spatial) and another for motion dynamics between frames (temporal). Both streams should contribute to a combined classification.						
Dependencies	F2						
Priority	Essential	X	Conditional	-	Optional	-	-
Change History	-						

Table 5.3: Table for Functional Requirements: Dual-Stream Detection

5.1.4 Functional Requirement 4

The system must detect face manipulations (deepfakes) using a stacking-based ensemble approach that combines features from the Xception and EfficientNet-B7 models. This method enhances detection accuracy by leveraging the strengths of both models for robust classification of deepfakes.

Requirement ID	FR4	Requirement Type		Functional	Use Case #		4
Status	New	X	Agreed-to	-	Baselined	-	-
Parent Requirement #	-						
Description	The system must detect face manipulations (deepfakes) using a stacking-based ensemble approach, leveraging features from Xception and EfficientNet-B7 models.						
Rationale	Deepfake videos often exhibit subtle face manipulations that are difficult to detect. A stacking-based ensemble approach, using features from both Xception and EfficientNet-B7, improves the detection accuracy by combining the strengths of both models.						
Source	-				Source Document	-	
Validation	The system should successfully detect deepfakes by applying the stacking-based ensemble method and combining features from the Xception and EfficientNet-B7 models for robust classification.						
Dependencies	F3						
Priority	Essential	X	Conditional	-	Optional	-	-
Change History	-						

Table 5.4: Table for Functional Requirements: Deepfake Detection

5.1.5 Functional Requirement 5

The system must process video data in real-time or near-real-time to provide immediate feedback, ensuring practical use for content moderators, cybersecurity professionals, and law enforcement. This allows for timely decision-making in critical situations.

Requirement ID	FR5	Requirement Type		Functional	Use Case #		5
Status	New	X	Agreed-to	-	Baselined	-	-
Parent Requirement #	-						
Description	The system must be capable of providing results in real-time or near-real-time for user-friendly, practical application.						
Rationale	Real-time or near-real-time processing ensures that the system is practical for users who need immediate feedback, such as content moderators, cybersecurity professionals, and law enforcement. This allows for faster decision-making in critical scenarios.						
Source	-				Source Document	-	
Validation	The system should be able to process video data quickly enough to provide results in real-time or near-real-time, enabling effective and timely decision-making.						
Dependencies	F1, F3						
Priority	Essential	X	Conditional	-	Optional	-	-
Change History	-						

Table 5.5: Table for Functional Requirements: Real-Time Processing

5.1.6 Functional Requirement 6

The system must have a user-friendly interface that allows users to easily input video data and view analysis results. This ensures accessibility for non-expert users, such as content moderators and law enforcement, enhancing the system's usability and practical application.

Requirement ID	FR6	Requirement Type		Functional	Use Case #		6
Status	New	X	Agreed-to	-	Baselined	-	-
Parent Requirement #	-						
Description	The system must have a user-friendly interface that allows users to easily input video data and receive analysis results.						
Rationale	A user-friendly interface is crucial to ensure that non-expert users, such as content moderators and law enforcement, can efficiently operate the system without specialized knowledge. This enhances usability and broadens the system's application.						
Source	-				Source Document	-	
Validation	The user interface should be intuitive, easy to navigate, and allow for seamless input of video data and display of analysis results. User experience testing should validate ease of use.						
Dependencies	F1						
Priority	Essential	X	Conditional	-	Optional	-	-
Change History	-						

Table 5.6: Table for Functional Requirements: User Interface

5.1.7 Functional Requirement 7

The system must generate a comprehensive report at the end of the analysis that details the percentage of fake versus real content in the video, along with explanations of the model's classification decision. This report enhances transparency and helps users understand the model's reasoning, supporting more informed decision-making.

Requirement ID	FR7	Requirement Type		Functional	Use Case #		7
Status	New	X	Agreed-to	-	Baselined	-	-
Parent Requirement #	-						
Description	The system must generate a detailed report at the end of the analysis, which provides the percentage of fake and real content in the video, as well as the basis for the model's classification decision.						
Rationale	Providing a comprehensive report with percentages and explanations for the model's classification ensures transparency and gives users insight into the reasons behind the results. This can help in validating the analysis and taking informed actions.						
Source	-				Source Document	-	
Validation	The system should produce a report that includes the percentage of fake and real video content, along with an explanation of the features (such as spatial anomalies, motion irregularities, etc.) that influenced the model's decision.						
Dependencies	-						
Priority	Essential	X	Conditional	-	Optional	-	-
Change History	F4, F6						

Table 5.7: Table for Functional Requirements: Report Generation

6 Non-Functional Requirement and Software System Attributes

6.1 Non-Functional Requirements

The nonfunctional requirements outline the key attributes that the system must meet to ensure its reliability, efficiency, security, and overall user experience. These requirements focus on aspects such as system performance, usability, and maintainability, which are essential for ensuring the platform operates effectively and meets user expectations.

6.1.1 Security

The system must ensure that all user data and video uploads are securely handled. This includes implementing encryption protocols for data transmission (such as HTTPS) and secure storage for sensitive information like user credentials and video files. User authentication will be enforced to protect against unauthorized access, especially for administrator functions. Regular security audits should be conducted to identify and mitigate vulnerabilities, ensuring the system adheres to security best practices.

6.1.2 Scalability

The platform must be designed to handle increasing volumes of video uploads and user interactions without degradation in performance. As the user base grows or more videos are submitted for detection, the system should be capable of scaling both vertically (enhancing server capacity) and horizontally (adding more servers). Cloud-based infrastructure will support this scalability, ensuring the system can dynamically allocate resources based on demand.

6.1.3 Availability

The system must be highly available, providing uninterrupted service to users. Downtime should be minimized, and the platform should aim for at least 99.9% uptime. This includes implementing redundant systems and failover mechanisms to ensure that, in the event of a server failure, the system remains operational. Regular monitoring should be in place to detect issues early and automatically restart or switch to backup systems when needed.

6.1.4 Accuracy

The AI detection models used in the system must provide high levels of accuracy in detecting AI-generated videos and deepfakes. The system should consistently produce reliable results with a high confidence score for real and manipulated content. Continuous improvement and retraining of models should be part of the system's lifecycle to ensure accuracy keeps pace with new types of deepfake technologies.

6.1.5 Usability

The platform should offer an intuitive and user-friendly interface that requires minimal training or technical knowledge. Users should be able to easily upload videos, understand their detection reports, and navigate through the site. The layout should be clean, with clear labels, instructions, and feedback mechanisms, ensuring a smooth user experience. Special attention should be given to accessibility features to accommodate users with disabilities.

6.1.6 Data Integrity

All data processed and stored by the system must remain accurate, complete, and consistent. The system must ensure that once a video is uploaded and processed, the results are not tampered with. Data backups should be regularly maintained, and mechanisms should be in place to prevent data corruption during transmission or storage. Any changes to models, settings, or system configurations must be tracked and logged for auditing purposes.

6.1.7 Maintainable

The system must be easy to maintain, ensuring that updates, patches, and improvements can be deployed without causing disruptions. The codebase should be modular and well-documented, allowing future developers to modify or upgrade components as needed. Regular system maintenance, including cleaning logs, optimizing databases, and updating models, should be scheduled to ensure long-term reliability and performance.

6.2 Software System Attributes

The software system attributes ensure the system operates effectively and meets user needs. Key attributes include security to prevent unauthorized access, scalability to handle increased demand, and availability to ensure constant access. Accuracy ensures reliable detection of AI-generated videos, while usability focuses on providing an intuitive interface. Data integrity maintains consistency and accuracy of information, and maintainability allows for updates and modifications with minimal disruption. Together, these attributes support a robust and efficient platform.

6.2.1 Performance Requirements

The system must ensure reliable and efficient performance under varying conditions. It should be capable of processing user requests and video uploads in a timely manner, providing results without significant delays. The platform needs to support multiple users simultaneously, maintaining consistent performance and responsiveness. Scalability is important to accommodate increased usage without compromising the system's speed or reliability. Additionally, the system must be able to handle unexpected errors gracefully, ensuring that performance remains stable and users receive continuous service. Overall, the platform should provide a smooth and efficient user experience, even under heavy load or high demand.

7 Project Design/Architecture

4+1 Architecture View Model

7.1 Use Case View

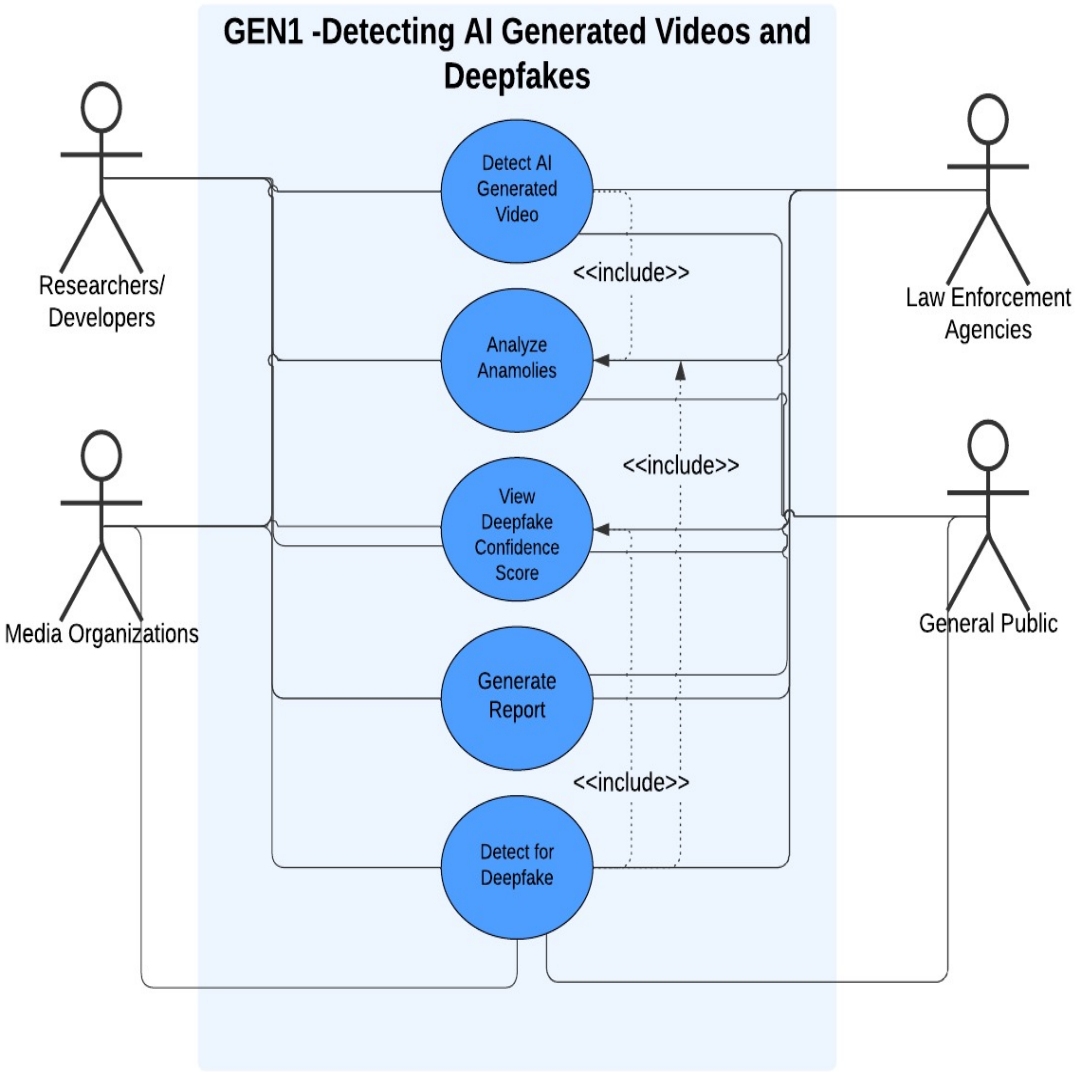

Figure 7.1: Use Case Diagram

7.2 Logical View

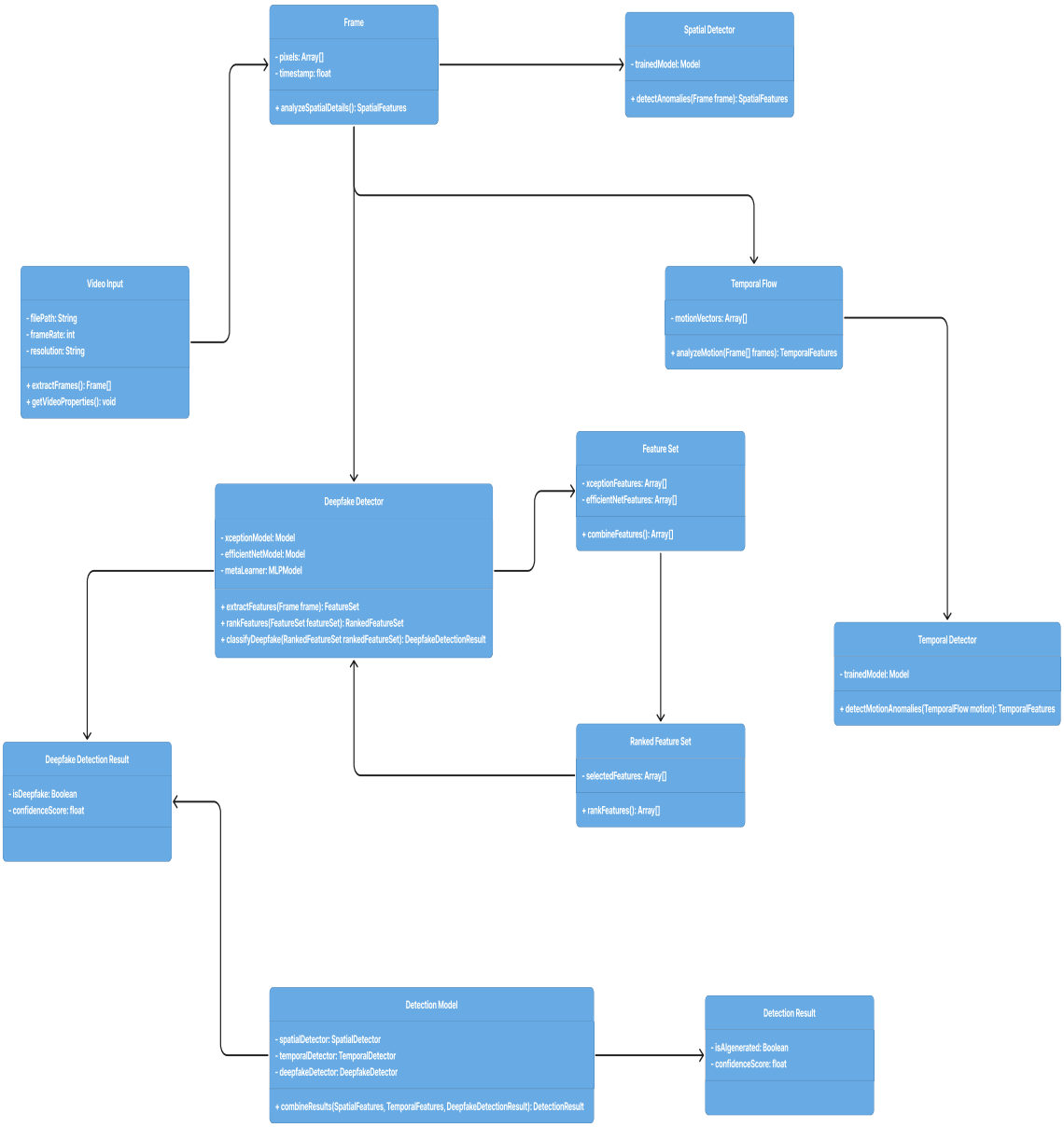

Figure 7.2: Class Diagram

The class diagram for the AI-generated video and deepfake detection project outlines several core components:

- Video Input: This class captures video properties such as file path, frame rate, and resolution. It includes methods to extract frames from the video and retrieve its properties.

- Frame: Each video frame consists of pixel data and a timestamp. This class has a method to analyze spatial details, providing spatial features for further analysis.
- Spatial Detector: A trained model is used to detect spatial anomalies within individual frames. It identifies inconsistencies in spatial features.
- Temporal Flow: This class tracks the motion vectors between frames and analyzes temporal characteristics.
- Temporal Detector: It uses a trained model to detect anomalies in motion vectors, identifying irregularities in the temporal flow of the video.
- Deepfake Detector: This component uses multiple models (Xception, EfficientNet, and a Meta-Learner MLP model) to extract and rank features from video frames. It then classifies whether the video is a deepfake.
- Feature Set and Ranked Feature Set: These classes handle the combination and ranking of features extracted from frames using different models to provide a unified set of data.
- Detection Model: This model integrates results from the spatial, temporal, and deepfake detectors, combining their outputs to generate an overall detection result.
- Detection Results: This class captures the final detection results, indicating whether a video is AI-generated or a deepfake, along with a confidence score.

7.3 Deployment View

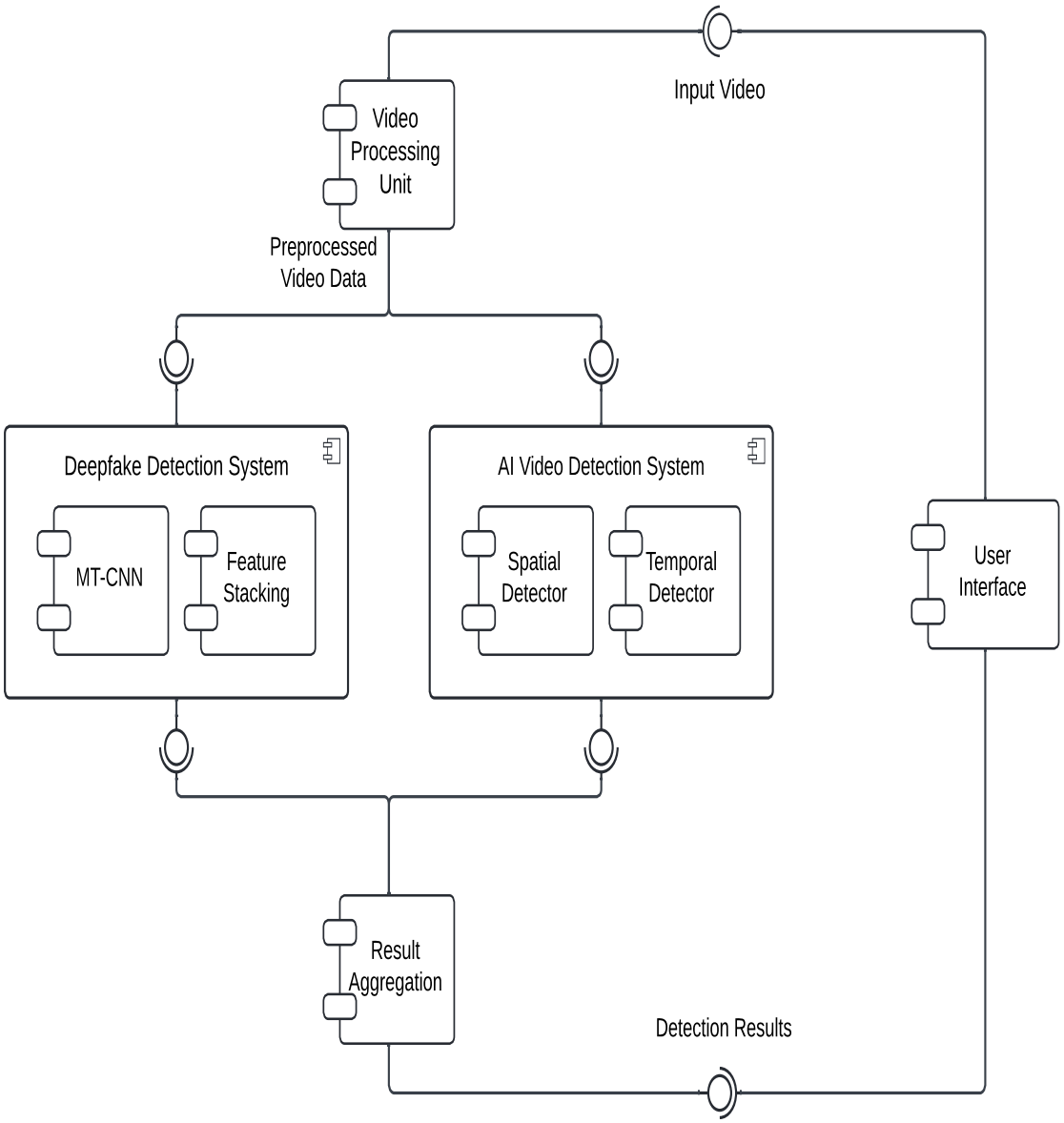

Figure 7.3: Component Diagram

The component diagram for the AI-generated video and deepfake detection project outlines a system that processes input videos through a series of detection stages. The User Interface enables video submission and displays results, while the Video Processing Unit handles video preprocessing. The system leverages MT-CNN Feature Stacking to combine spatial, temporal, and deepfake-related features, feeding into both the Deepfake Detection System, which uses spatial and temporal detectors, and the AI Video Detection System

for AI-generated content identification. Finally, the Result Aggregation component consolidates detection outcomes to provide a comprehensive report to the user.

7.4 Physical View

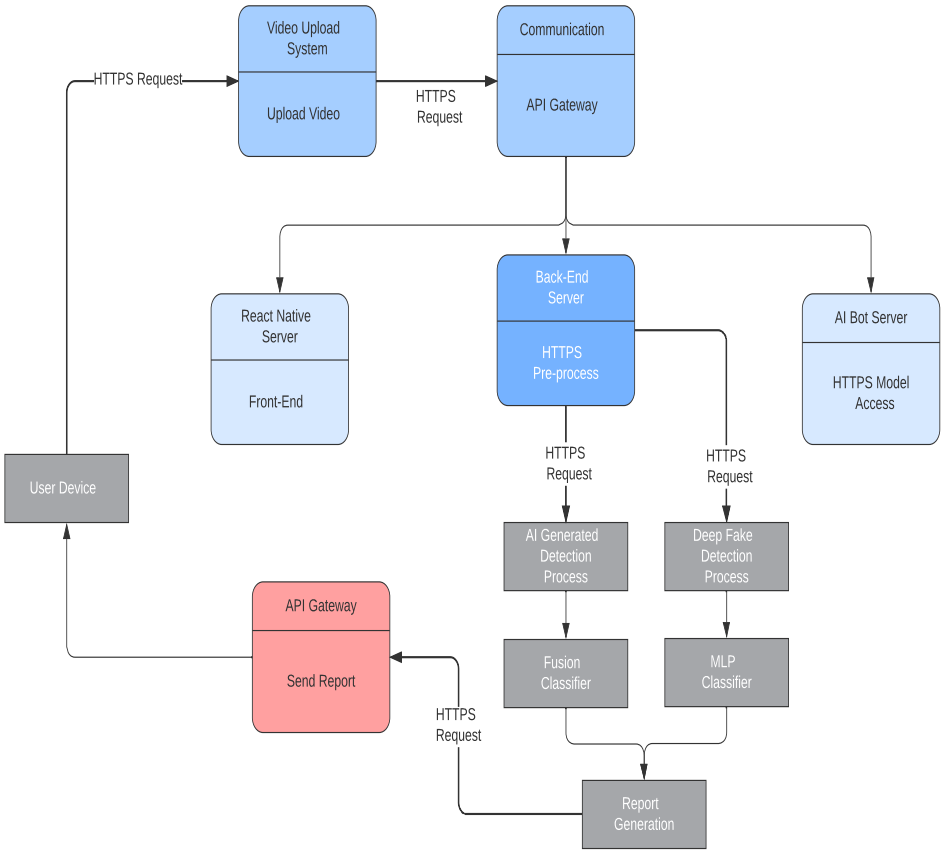

Figure 7.4: Physical View Diagram

The project involves detecting AI-generated and deepfake videos through a structured system. A user uploads a video via a front-end interface connected to a back-end server. The

system sends the video for processing, where it passes through two detection processes: AI-generated detection and deepfake detection. Both processes use machine learning models (including MLP and Fusion classifiers) to classify the video content. Once analyzed, the results are communicated via HTTPS through an API Gateway, and a report is generated and sent back to the user, summarizing the detection outcome.

7.5 Sequence Diagram

AI Content Sequence Diagram

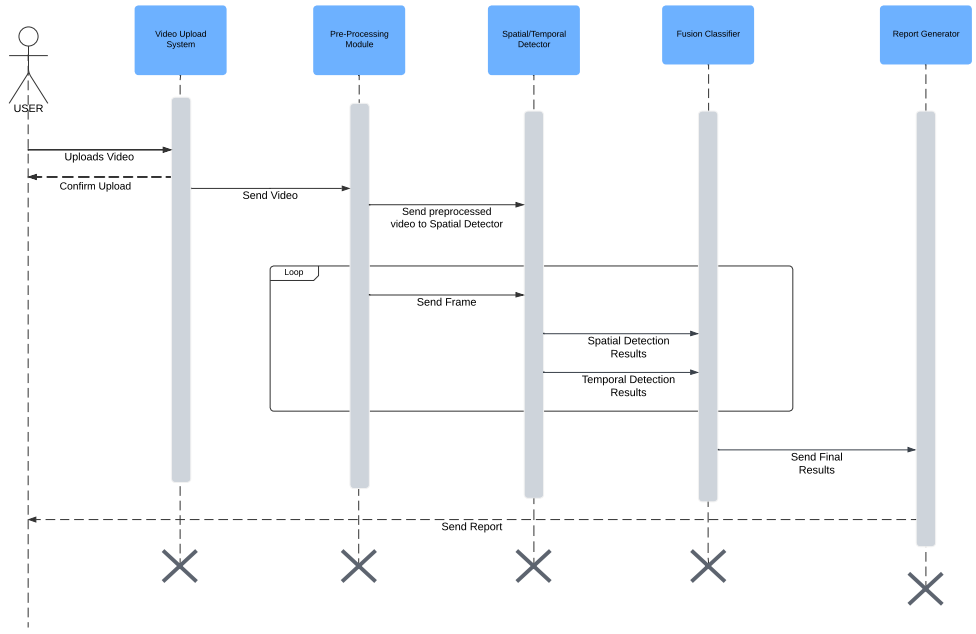

Figure 7.5: Sequence Diagram

The sequence diagram illustrates the process of detecting AI-generated and deepfake videos. It begins with a user uploading a video, which is confirmed by the system. The video is first sent to a pre-processing module, which prepares the video for analysis. The pre-processed video is then sent to both spatial and temporal detectors. The results from these detectors are processed and passed on to a fusion classifier. After the final detection results are compiled, they are sent to a report generator, which creates and sends a detailed report back to the user.

Deepfake Sequence Diagram

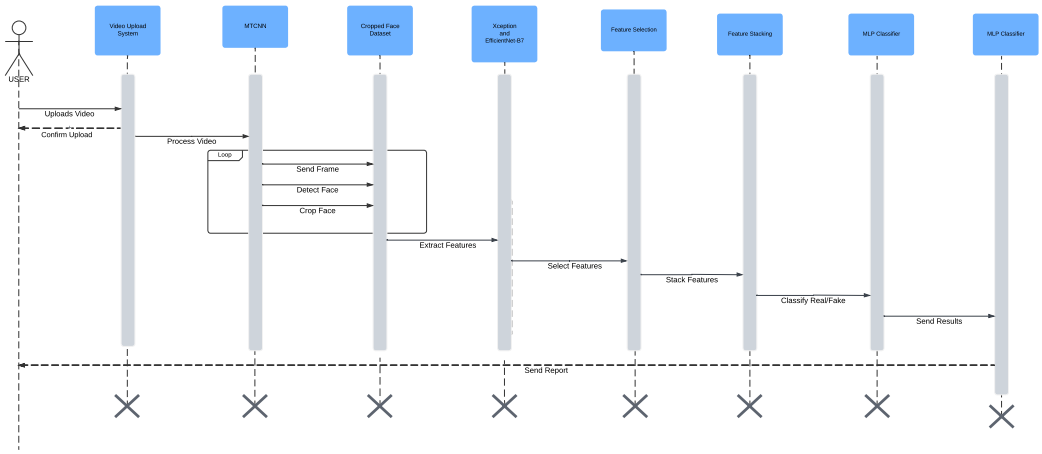

Figure 7.6: Deepfake Sequence Diagram

The sequence diagram for the deepfake detection process illustrates the steps involved from video upload to final classification. The User uploads a video via the Video Upload System, which triggers the processing of the video. Faces are detected and cropped using MTCNN, followed by feature extraction using Xception and EfficientNet-B7 models. These extracted features are then passed through a Feature Selection and Feature Stacking process, before being classified as real or fake using an MLP Classifier. The system then sends the detection results and generates a report for the user.

Bibliography

[1] Abdulqader M Almars. Deepfakes detection techniques using deep learning: a survey. *Journal of Computer and Communications*, 9(05):20–35, 2021.

[2] M Elavarasi, R Pramodhini, M Deshmukh Deepak, R Mekala, and Chamandeep Kaur. Image and video anomaly detection using ai based deepanomaly detectors. *ICTACT Journal on Image & Video Processing*, 14(2), 2023.

[3] Danial Samadi Vahdati, Tai D Nguyen, Aref Azizpour, and Matthew C Stamm. Beyond deepfake images: Detecting ai-generated videos. *arXiv e-prints*, pages arXiv–2404, 2024.