AVANTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)

AI-Powered Deep Fake Detection

INTRODUCTION

Deepfake technology leverages artificial intelligence (AI) to synthesize hyper-realistic videos that manipulate facial expressions, voice, and context. While innovative, this technology poses significant societal risks, including the spread of political misinformation, financial fraud through CEO impersonation, and identity theft via forged biometric data. Current detection tools often lack scalability, transparency, or user-centric design, creating a gap in accessible solutions for non-technical users. This project addresses these challenges by developing a **Django-based web application** that combines computer vision and deep learning to detect deepfakes in real time, prioritizing usability and accuracy.

PROBLEM STATEMENT

Existing deepfake detection systems face three critical shortcomings:

1. **Slow processing speeds**, limiting real-time analysis.
2. **Complex interfaces**, excluding non-technical users.
3. **Opaque decision-making**, lacking interpretable results (e.g., confidence scores)

This project aims to deliver a web application that provides:

- **Real-time analysis** of uploaded videos.
- **Intuitive design** for seamless user interaction.
- **Transparent results** with visual explanations.

METHODOLOGY

Hybrid Detection Approach

1. **Computer Vision Pipeline**
    - **Frame Extraction**: OpenCV processes uploaded videos into frames.
    - **Facial Analysis**: The face_recognition library detects inconsistencies in facial movements (e.g., unnatural blinking, lighting artifacts).
2. **Deep Learning Models**
    - **Model Architecture**: Pretrained PyTorch models
    - **Classification**: CNNs and RNNs analyze frames for AI-generated artifacts, outputting authenticity probabilities.

User Interface

- **Frontend**: HTML/CSS/JavaScript for video upload, progress tracking (via TQDM progress bars), and result visualization.
- **Backend**: Django handles file management (UUID, OS module), data processing (Pandas, NumPy), and model inference.

Workflow

1. User uploads a video.
2. System extracts frames, detects faces, and runs hybrid analysis.
3. Results display confidence scores

SOLUTION OFFERED

Automated DeepFake Detection: The system processes uploaded videos, extracts frames, detects faces, and classifies authenticity using deep learning.

Real-time Feedback: Users receive real-time results with confidence scores to assess video authenticity.

User-Friendly Web Application: A simple and intuitive interface for seamless video upload and accuracy result.

END USERS

- **Journalists/Media:** Verify authenticity of viral content.
- **Legal Teams:** Analyze evidence for forensic investigations.
- **Social Media Platforms:** Automatically flag Check personal media credibility,deepfakes.
- **General Public:** Check personal media credibility.

TECHNOLOGIES USED

Programming Language -> Python

Web Framework -> Django (backend), HTML/CSS/JS (frontend)

Deep Learning -> PyTorch, Torchvision (pretrained models)

Computer Vision -> OpenCV, face_recognition (facial analysis)

Data Handling -> Pandas, NumPy, scikit-learn, PIL

System Interaction -> OS module, UUID (file management)

Progress Tracking -> TQDM (user-facing progress bars)

RESULT

- Expected >90% accuracy on benchmark datasets.
- Real-time processing (<30 seconds for a 15-second video).
- Interactive dashboard showing detection and visual explanations.

CONCLUSION

This project bridges the gap between advanced AI research and practical tools by combining Django's scalability, PyTorch's deep learning capabilities, and OpenCV's real-time processing. Key achievements include:

- A hybrid detection system for robust deepfake identification.
- A user-friendly interface democratizing access to AI-powered verification.