Junior Design Project

DeepFake Detection Using Multimodal Analysis (CNN and RNN)

Semester: Spring 2025

Course Code: CSE 299.5

Course Title: Junior Design

Faculty: Tanjila Farah

Submitted By:-

Atique Shahrier Chaklader- 2132882642.
Samin Yeasar Badhon- 2212208042.
Md Waliul Sakib - 2212570042.

1.Introduction

With the rise of deepfake media, detecting fake content has become crucial. This project focuses on building a Deepfake Detection system that leverages both visual and audio cues to classify videos as "Real" or "Fake".

The system integrates two machine learning models:

A CNN (Convolutional Neural Network) for analyzing video frames

An RNN (Recurrent Neural Network) for analyzing audio spectrograms

The project’s goal was to build a robust, multimodal Deepfake detection tool

2. Objectives

- Detect fake videos using CNN (TensorFlow) model.

- Detect fake audio using RNN (LSTM) model.

- Develop a system that combines both detections.

- Provide confidence scores and results visualization.

3.(a) Literature Review

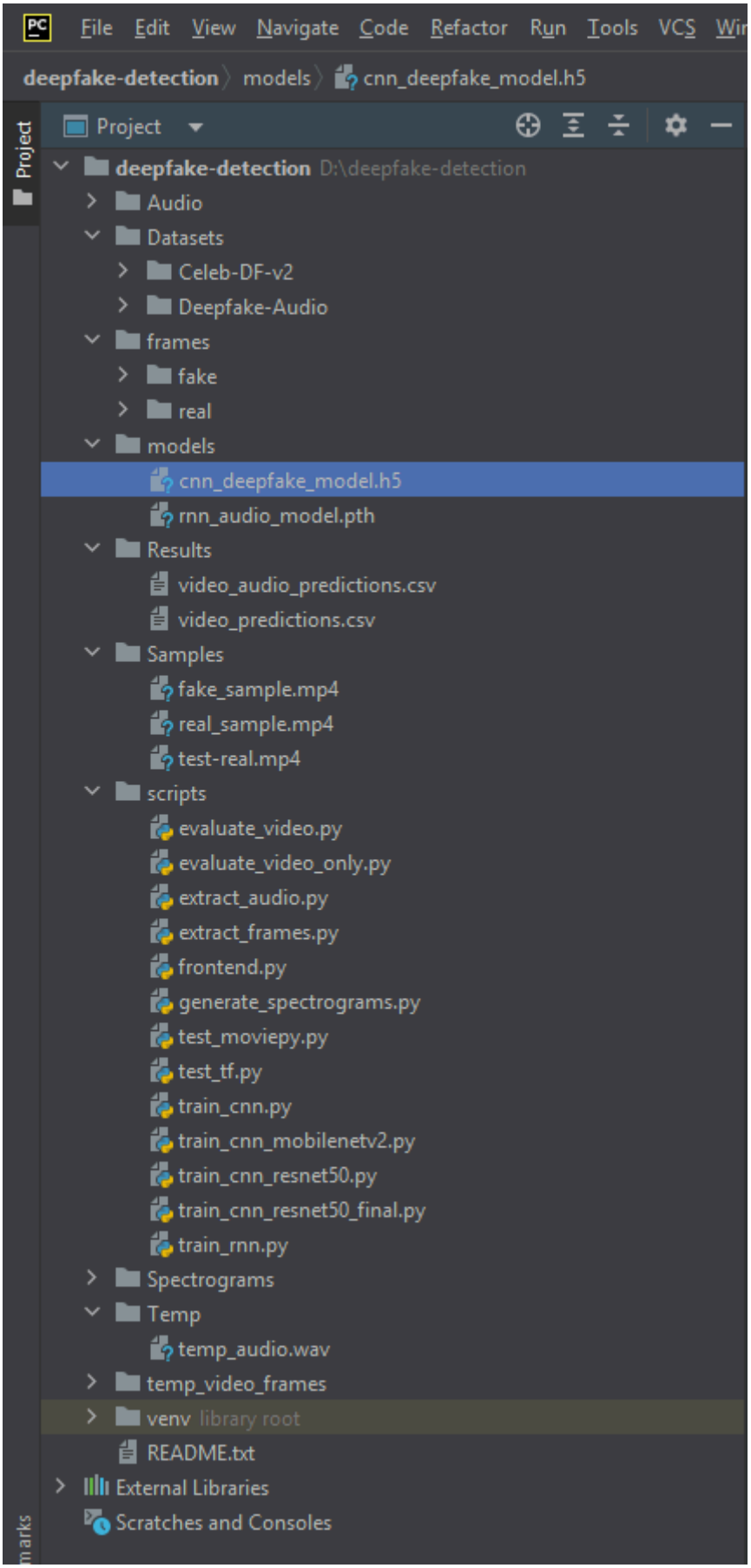
Previous research on deepfake detection has extensively utilized Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and hybrid models that combine spatial and temporal feature extraction. Studies have shown that models leveraging these architectures can effectively capture visual and temporal inconsistencies present in manipulated media. Among available datasets, Celeb-DF-v2 has been widely adopted as a benchmark due to its improved visual quality and closer resemblance to real-world forgeries, making it particularly suitable for evaluating deepfake detection systems. Building upon these approaches, we customized traditional methods by enhancing frame-level feature aggregation techniques and incorporating spectrogram analysis, aiming to better exploit multimodal cues for more accurate detection.

3.(b) Existing Research and Limitations

In recent years, extensive research has been conducted in the field of deepfake detection, utilizing convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to classify real and fake multimedia content. Public datasets such as Celeb-DF v2 and FaceForensics++ have been used widely for model training and benchmarking. Although significant progress has been made, achieving high accuracy across all types of deepfakes remains a challenge. In our project, while the CNN-based model demonstrated relatively good performance on real video frames, it faced difficulties when classifying fake frames, often predicting fake videos as real with high confidence. This imbalance is largely attributed to two factors: the dataset imbalance (more fake frames than real frames) and the subtle artifacts present in newer high-quality deepfake videos, which the model struggles to distinguish. Furthermore, the limited diversity of training examples and the overfitting to dataset-specific features reduced the model's generalization capability to unseen deepfakes. These challenges highlight the current limitations in deepfake detection systems and indicate the need for larger, more diverse datasets and more robust model architectures for future improvement.

4.Methodology

Our multimodal deepfake detection methodology involved extracting video frames using OpenCV and extracting audio tracks using FFmpeg. A CNN classifier, built with TensorFlow, was trained on frames from the Celeb-DF-v2 dataset to detect visual anomalies. Simultaneously, a Tensorflow-based RNN was trained on spectrograms from the Deepfake-Audio dataset to capture temporal audio inconsistencies. Finally, we developed an integration script that combines predictions from both modalities to deliver a unified deepfake detection result.



Project scripts and folder contents.

Complex Engineering Problems and Activities

Attribute	Addressing the complex engineering problems (P) in the project
P1: Depth of knowledge required (K3–K8)	The project requires knowledge of Artificial Intelligence, deep learning (K8), computer vision and signal processing (K6), CNNs and RNNs (K6), and handling large datasets. It also involves preprocessing with ffmpeg and spectrogram generation (K5).
P2: Range of conflicting requirements	Conflicting requirements exist in balancing performance and computational efficiency, especially due to large datasets and high model complexity. Model tuning to reduce false positives vs. false negatives presents trade-offs.
P3: Depth of analysis required	Extensive analysis was needed for model selection, dataset balancing, preprocessing decisions (frame count, spectrogram dimensions), and confidence aggregation strategies.
P4: Familiarity of issues	Issues included working with unfamiliar tools like PyTorch, TensorFlow, ffmpeg, spectrogram libraries (Librosa), and video/audio data. Also tackled evaluation difficulties due to prediction ambiguity.
P5: Extent of applicable codes	No existing standard code exists for the full deepfake detection pipeline. Custom scripts for audio/video preprocessing, model training, and evaluation were written.
P6: Extent of stakeholder involvement	Stakeholders include content moderation systems, users of online media platforms, and regulatory bodies aiming to identify misinformation and digital fraud.
P7: Interdependence	The project includes multiple interdependent systems: frame extraction, CNN for video, spectrogram + LSTM for audio, unified evaluation logic, and a visual frontend for prediction results.

Attribute	Addressing the complex engineering activities (A) in the project
A1: Range of resources	The project uses machine learning frameworks (TensorFlow, PyTorch), video/audio datasets (CelebDF-v2, Deepfake-Audio), GPUs/CPUs for training, and frontend libraries for interface deployment.
A2: Level of interactions	Interactions occur between model predictions (video + audio), between multiple scripts/modules, and among team members collaborating on training, evaluation, and frontend design.
A3: Innovation	Innovative use of a dual-modality (video + audio) approach to classify deepfakes. Integrates post-processing aggregation without needing retraining to boost performance.
A4: Consequences to society/Environment	Strong societal impact by offering a tool to combat digital misinformation and identity fraud. Ethical implications in media security and authentication.
A5: Familiarity	Requires familiarity with spectrograms, frame extraction, ffmpeg, neural networks (ResNet, LSTM), as well as frontend tools for deployment and interpretation of AI results.

5.(a) Dataset

Celeb-DF-v2 Dataset (for videos and frames)

https://github.com/yuezunli/Celeb-DF

Deepfake Audio Dataset (for audio detection)

https://github.com/AsaduzzamanCSE/deepfake_audio_dataset

5.(b) Dataset Preparation:

Extracted frames from videos using OpenCV.(Frame generation was varied among 1 frames/sec, 2 frames/sec and maximum number of frames generated without any time specified. We determined the usage of the maximum frame generation method for clearer and more accurate data training and analysis.

Extracted audio using ffmpeg and converted into spectrograms using Librosa.Spectrograms were created for each .wav audio file and stored in a separate folder for further usage.

6.Tools and Libraries Used

- TensorFlow – for training and evaluating CNN models (tensorflow.org)
- Keras – API used for CNN model building
- PyTorch – for building and training the RNN (LSTM) model (pytorch.org)
- OpenCV – for video frame extraction
- Librosa – for audio processing and spectrogram generation
- ffmpeg – for extracting audio from video files (ffmpeg.org)
- Matplotlib – for plotting predictions
- NumPy – for numerical operations
- Celeb-DF-v2 Dataset Paper: https://arxiv.org/abs/2003.07590
- MobileNetV2 Paper: https://arxiv.org/abs/1801.04381
- ResNet Paper: https://arxiv.org/abs/1512.03385
- TensorFlow: https://www.tensorflow.org/
- PyTorch: https://pytorch.org/ Deepfake Detection Using CNN and RNN
- OpenCV: https://opencv.org/
- Librosa: https://librosa.org/
- FFmpeg: https://ffmpeg.org/

7. Implementation Details

Frame Extraction: Up to 200 frames per video using OpenCV.

CNN Model:

Architecture: MobileNetV2 and ResNet50.

Classifies each frame as Real or Fake.

Fine-tuned with imbalanced dataset management.

Audio Extraction and Processing:

Extracted audio from videos using ffmpeg.

Converted audio into spectrograms using Librosa.

Used LSTM (RNN) model for spectrogram classification.

Voting Mechanism:

Frame predictions and audio predictions are separately classified.

Majority voting on frames and spectrogram output determines the final video label.

Frontend:

A user-friendly Python-based Tkinter frontend.

User can input videos for Real/Fake evaluation.

Evaluation:

Predictions are saved to a structured CSV file after analysis.

8.Training Details

Training on local computer was extensively resource demanding for this project. Still, however, the program could use more training and data collection for further accuracy.Training the models, especially ResNet50 was very resource demanding and implementing this on our personal computer resulted in some hardware issues due to excessive RAM and CPU usage, causing overheating.

- CNN Model: Based on MobileNetV2 / ResNet50 architectures.

- RNN Model: Bi-directional LSTM layers trained on 128 Mel-spectrogram inputs

- Video Dataset:

- Trained CNN using class weights. →

```
DATA_DIR = r"D:\deepfake-detection\frames"
MODEL_SAVE_PATH = r"D:\deepfake-detection\Models\cnn_deepfake_model.h5"
IMAGE_SIZE = (224, 224)
BATCH_SIZE = 4
EPOCHS = 5
```

- Frames extracted and balanced.-->

```
def extract_frames(video_path, output_folder, frame_interval=15):
    cap = cv2.VideoCapture(video_path)
    frame_count = 0
    video_name = os.path.basename(video_path).split('.')[0]

    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break
        if frame_count % frame_interval == 0:
            frame_filename = os.path.join(output_folder, f"{video_name}_frame_{frame_count}.jpg")
            cv2.imwrite(frame_filename, frame)
        frame_count += 1

    cap.release()
    print(f"✅ Extracted frames from {video_name}")
```

- Audio Dataset:

- Generated spectrograms from audio clips. →

```
1 usage
def convert_to_spectrogram(audio_path, output_folder):
    try:
        y, sr = librosa.load(audio_path, sr=None)
        spectrogram = librosa.feature.melspectrogram(y=y, sr=sr)
        spectrogram_db = librosa.power_to_db(spectrogram, ref=np.max)

        plt.figure(figsize=(10, 4))
        librosa.display.specshow(spectrogram_db, sr=sr, x_axis='time', y_axis='mel')
        plt.colorbar()

        audio_name = os.path.basename(audio_path).replace( __old: ".wav", __new: ".png")
        plt.savefig(os.path.join(output_folder, audio_name))
        plt.close()
        return True
    except Exception as e:
        print(f"❌ Error processing {audio_path}: {e}")
        return False

audio_files = [f for f in os.listdir(AUDIO_DIR) if f.endswith(".wav")]

if len(audio_files) == 0:
    print("❌ No audio files found! Run `extract_audio.py` first.")
else:
    for audio_file in tqdm(audio_files, desc="Generating spectrograms"):
        convert_to_spectrogram(os.path.join(AUDIO_DIR, audio_file), SPECTROGRAM_DIR)

print("🎉 Spectrogram generation complete!")
```

- Trained RNN with appropriate hyperparameters.

9. Prediction System

In the final stage of our system, predictions are made independently for the video and audio components. For video, individual frames extracted from the input are processed through the trained CNN model, and frame-level predictions are aggregated to form an overall video verdict. Similarly, the extracted audio is transformed into spectrograms and evaluated using the trained LSTM-based RNN model, producing an audio-based prediction. To arrive at a final decision, the outputs from both modalities are combined, leveraging aggregation strategies such as majority voting or weighted averaging. Based on the integrated analysis, each input sample is classified as either **Real** or **Fake**. The system then records all predictions along with corresponding metadata in a structured CSV file, facilitating easy report generation, performance evaluation, and further analysis.

10.Conclusion

By utilizing a combination of video frame and audio spectrogram analysis, this system provides a more robust, multi-modal deepfake detection method compared to using only video.
The combination of CNN and RNN improved the final prediction accuracy, and made the system less vulnerable to deepfakes that only tamper either video or audio separately.