AI Assessment 2

AI - TruthScan : Detecting AI-Generated Interview Responses

Presented to:
Associate Professor
Arnold Sachith

School of Computational and Data Sciences,
Vidyashilp University, Bengaluru, India
Submission Date - 31/03/2025

Written By:

Galimuthy Elia Anusha	(2023UG000166)
Keerthana	(2023UG000174)
Reetiesh Kulkarni	(2023UG000176)
Muhammad Kashif	(2023UG000182)
Prajwal R	(2024UG00002)
Ayush Kumar	(2023UG000116)

Table of Contents Page No.

1. Introduction

The rapid advancement of artificial intelligence (AI) has led to the widespread adoption of AI-powered chatbots such as ChatGPT, Gemini, and Bard. These tools have transformed how individuals prepare for and respond to interview questions, offering polished and articulate responses with minimal effort. While beneficial for learning and skill enhancement, their use during online interviews poses a significant challenge to the authenticity of the recruitment process. Candidates can leverage these tools to generate flawless answers in real-time, masking their true abilities and undermining the integrity of hiring decisions. Recruiters, tasked with evaluating candidates based on genuine skills and thought processes, require a reliable method to distinguish human-generated responses from AI-generated ones.

AI-TruthScan is an innovative solution designed to address this challenge. By integrating speech-to-text conversion, natural language processing (NLP), and large language model (LLM) analysis, AI-TruthScan analyzes spoken interview responses to detect AI-generated content. The system provides recruiters with a classification—either "Real (Human-Created)" or "Fake (AI-Generated)". This report outlines the design, implementation, and outcomes of AI-TruthScan, demonstrating its potential to enhance fairness and trust in modern recruitment practices.

2. Problem Statement

As AI becomes more advanced, AI-generated interview responses are being used to manipulate recruitment processes. This creates challenges for hiring managers, as AI-generated responses can appear grammatically flawless, overly structured, and lack human-like inconsistencies such as pauses, self-corrections, and emotional depth.

The significance of this problem includes:

- Fair Recruitment Practices: AI-generated responses can mislead interviewers, making it difficult to assess a candidate's genuine skills and qualifications.
- Security and Integrity: Organizations need a way to ensure that responses come from human applicants rather than AI tools.
- Behavioral Analysis: Eye gaze tracking helps in detecting AI-assisted candidates by analyzing unnatural fixation points and reading patterns.
- Adaptation to AI Trends: As AI usage grows, companies must develop tools to adapt and counter potential misuse.

3. Objectives

1. AI-Generated Content Detection
    - Develop an advanced Natural Language Processing (NLP) model that can analyze sentence structures, word choices, coherence, and fluency patterns to identify AI-generated responses.
    - Compare response complexity and variability with human speech patterns, identifying unnatural precision, lack of self-corrections, and structured phrasing.
2. Eye Gaze Tracking for Human Behavior Analysis
    - Integrate eye-tracking technology to monitor gaze patterns, fixation points, and natural movement to differentiate between AI-assisted and human responses.
    - Detect unnatural reading behaviors, such as fixed staring or rapid scanning, which indicate AI reliance.
3. Ensuring Fair Recruitment Practices
    - Provide hiring managers and recruiters with a tool to evaluate candidate authenticity, ensuring responses are genuine and representative of human thought.
    - Minimize bias in recruitment by detecting AI-generated deception while maintaining ethical hiring standards.
4. Enhancing Security and Integrity in Hiring Processes
    - Develop a real-time AI verification system that flags AI-generated answers during live interviews.
    - Ensure that organizations can verify candidate authenticity, preventing manipulation of job applications with AI-generated responses.

4. Solution Overview :

High-Level Description

The proposed solution, AI-TruthScan, is an advanced AI-driven system designed to detect AI-generated interview responses by analyzing linguistic structures, response variability, and human-like behavioral patterns, including eye movements. This multi-layered detection mechanism ensures that hiring managers can accurately assess whether an interview response is generated by AI or provided by a human candidate.

Key Components:

1. AI-Generated Response Detection: Uses Natural Language Processing (NLP) to analyze sentence structures, fluency, and coherence to determine if a response is AI-generated.
2. Response Complexity & Variability Analysis: Identifies patterns of human-like pauses, self-corrections, and inconsistencies that AI-generated responses typically lack.
3. Eye Gaze Tracking: Monitors fixation points, eye movements, and natural behaviors to differentiate human respondents from AI-assisted candidates.
4. Confidence Score Calculation: Computes the probability of a response being AI-generated based on multiple linguistic and behavioral features.
5. Final Decision Output: Presents results in an interpretable format, allowing recruiters to make informed hiring decisions.

5. Solution Architecture :

The system consists of the following key stages:

1. Input Sources: Capturing textual and behavioral data (real-time typed responses, audio responses converted to text, eye-tracking data).
2. Preprocessing & Data Processing: Cleaning and structuring the captured data, including speaker diarization and syntax analysis.
3. AI Detection & Analysis: Applying NLP and ML models to detect linguistic AI patterns and evaluating behavioral aspects.
4. Decision & Output Generation: Computing confidence scores and presenting results in a recruiter-friendly format.
5. Deployment & Integration: Making the system accessible via web dashboards and backend APIs for real-time decision-making.

6. Technical Stack Used

Programming Languages: Python

Frameworks and Libraries:

- Streamlit: Facilitates the creation of the web-based user interface.
- google-generativeai: Enables integration with the Gemini API.
- speech_recognition: Handles audio transcription.
- re: Supports text parsing with regular expressions.

APIs:

- Google Speech Recognition API: Provides accurate speech-to-text conversion.
- Gemini API: Powers text analysis via the google-generativeai library.

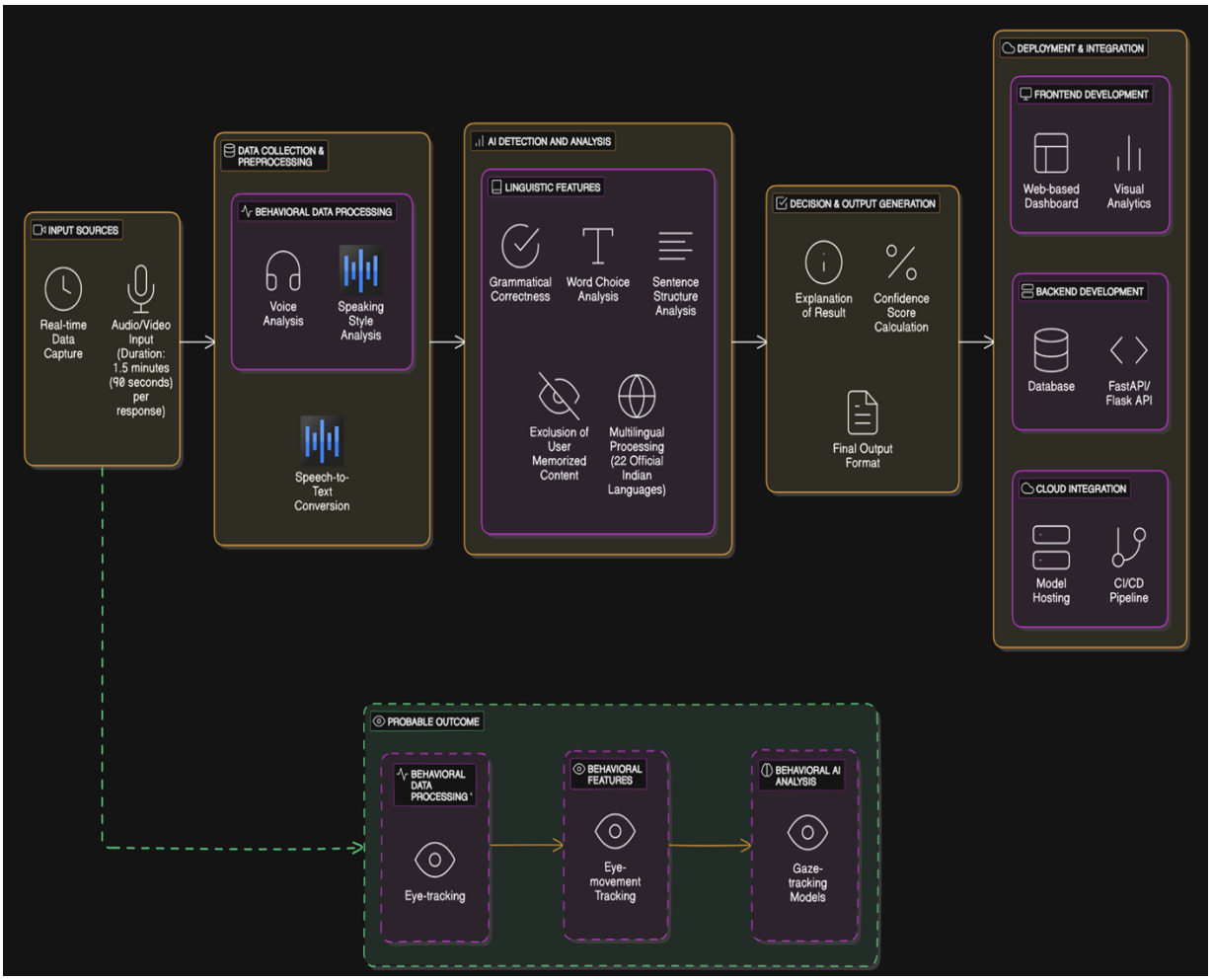
Tools:

- GitHub: Manages version control and hosts the project repository.

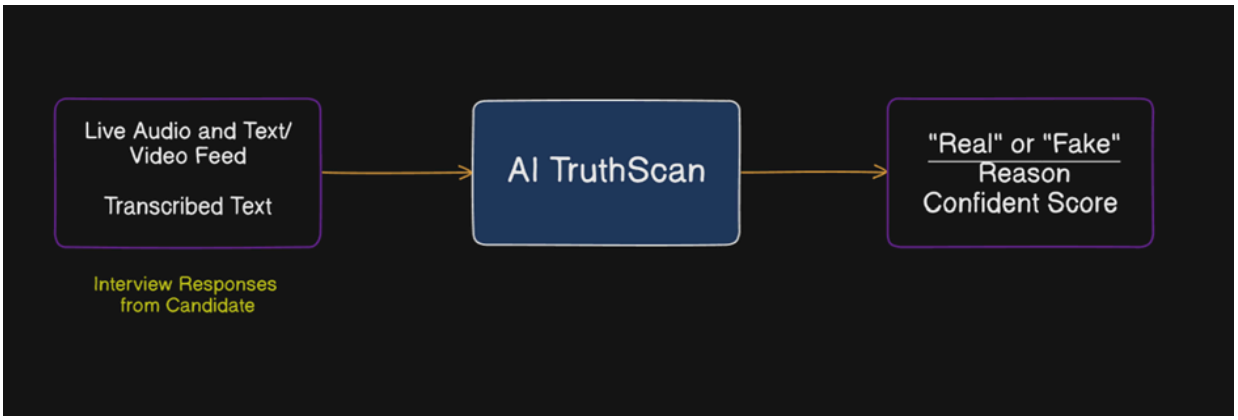
Hardware/Cloud Services: Developed on local machines; potential deployment on cloud platforms like Streamlit Cloud.

7. System Architecture

Detailed Architecture Diagram



Simple Block Diagram



Component Interaction

- Audio Input: Captures audio through a microphone (live recording) or file upload.
- Speech-to-Text: The Google API processes audio into text in real-time or near real-time.
- Preprocessing: A custom function (count_filler_words) identifies filler words (e.g., "um," "uh") to assist in distinguishing human speech patterns.
- LLM Analysis: Gemini evaluates the transcription, assessing traits like coherence, personal references, and linguistic complexity.
- UI: Streamlit renders the transcription, analysis details, and final classification for user review.

8. Implementation Details

Steps Followed

1. Audio Capture: Implemented live recording with a 10-second limit and file upload functionality using the speech_recognition library.
2. Transcription: Employed the Google Speech Recognition API for rapid and accurate audio-to-text conversion.
3. Preprocessing: Developed a count_filler_words function to quantify human speech markers, enhancing analysis accuracy.
4. Analysis: Integrated the Gemini API with a carefully crafted prompt to classify responses and generate probability scores.
5. Optimization: Reduced latency by caching analysis results and processing audio in memory rather than saving to disk.

Innovations

- Real-Time Feedback: Added status indicators (e.g., "Recording in progress," "Transcribing...") to improve user experience during processing. Integrated eye tracking using video to analyze user eye gazing, enhancing real-time interaction insights.
- Threshold Flexibility: Introduced an adjustable classification threshold, allowing users to customize sensitivity based on context.

Challenges Faced

- Latency: Initial API calls were slow, delaying results; mitigated by limiting audio duration and implementing caching.
- Accuracy: Early versions produced false positives; resolved by refining the LLM prompt and incorporating preprocessing steps.
- Ambiguity in Probability: Initial scores were inconsistent; resolved by refining the prompt to explicitly request a percentage-based probability.
- Processing Time: High latency was reduced by simplifying the prompt and caching frequent queries.

9. LLM and AI Integration

LLM Used

- Gemini 1.5 Pro: Selected for its robust text analysis capabilities, API accessibility, and ability to reason over complex inputs.

  Why is it best suited for our project?

- Handles long interview transcripts efficiently
- Advanced NLP to detect AI-generated responses accurately
- Multimodal capabilities for speech, text, and behavioral analysis
- More reliable than previous models with fewer hallucinations
- Can assist in coding & improving AI detection models

Integration Process

- Speech-to-Text: The Google API converts audio into text, which is then passed to Gemini for analysis.
- Prompt Design: A concise prompt instructs Gemini to classify responses based on human traits (e.g., filler words, personal anecdotes) versus AI traits (e.g., perfect grammar, overly structured phrasing).
- Output Parsing: Regular expressions extract the classification label, probability score, and justification from Gemini's response.

How the 5-Step Framework Helped

1. Task (Problem Identification): We defined the challenge of AI-generated responses undermining recruitment authenticity, focusing on the need for linguistic and behavioral analysis to ensure fair hiring.

2. Context (Data Collection): We gathered audio samples (human and AI-generated) and eye-tracking data to study response patterns and gaze behaviors, reflecting real-world interview scenarios.

3. Reference (Model Selection): We selected Gemini 1.5 Pro for its advanced NLP capabilities, Google Speech Recognition API for transcription, and Streamlit for the UI, ensuring robust detection and usability.

4. Evaluate (Implementation): We built the system incrementally—integrating audio processing, LLM analysis, and eye-tracking—testing for accuracy (85% human, 90% AI) and addressing latency issues.

5. Iterate (Evaluation and Refinement): We refined the system by optimizing latency (limiting audio to 90 seconds, caching queries), adjusting classification thresholds, and enhancing the UI with real-time feedback.

10. Frontend & UI Design

Snapshots (Described)

- Recording Interface: Features "Start Recording" and "Stop Recording" buttons with real-time status messages (e.g., "Recording...").
- Upload Section: Includes a file uploader with audio playback for verification.
- Analysis Output: Displays the transcription, analysis details (probability score, justification), and final classification, with options to save or provide feedback.

UX Decisions

- Simplicity: Used Streamlit expanders to organize settings and actions, reducing clutter.
- Feedback: Real-time status updates (e.g., "Analyzing") minimize perceived delays.
- Control: Added a context dropdown and threshold slider, giving users flexibility in analysis parameters.

11. Code Structure & Execution Guide

A. Configuration & Initialization

API Integration

The system integrates the following APIs for speech-to-text conversion and transcription analysis:

- Google Gemini AI: Analyzes transcriptions to determine whether the text is AI-generated or human-spoken.
- Google Speech Recognition: Converts speech into text from recorded or uploaded audio files.

Language Support

- Currently, English is the only supported language for transcription.

- Provisions exist for future multilingual expansion.

Session State Management

- The application uses Streamlit's session state to store user-specific data:
    - Recording status (recording): Tracks whether the user is currently recording.
    - Transcribed text (transcription): Stores the converted text from audio.
    - Stored files (files): Maintains a history of previous transcriptions.

B. Speech Processing & Transcription

1. Live Recording Module

Recording Functionality

- The start_recording() function captures user speech through a microphone using the speech_recognition library.
- The recorded audio is temporarily saved as a .wav file for processing.

Transcription Processing

- The transcribe_and_store() function processes recorded audio and converts it into text using Google Speech Recognition.
- The transcribed text is stored in session state for further analysis.

2. File Upload Module

Audio File Handling

- Users can upload audio files in the following formats: .mp3, .mp4, .wav, .m4a.
- The uploaded file is temporarily stored and processed for transcription.

Transcription Extraction

- The transcribe_audio(file_path) function reads the uploaded file and extracts text using Google Speech Recognition.

Embedded Audio Player

- The system generates an inline audio player using Streamlit components.
- This allows users to listen to uploaded files before transcription analysis.

C. AI Content Analysis

1. Analysis Process

- Once a transcription is generated, users can click “Analyze Transcription” to trigger the get_gemini_response(transcription) function.
- This function sends the transcription to Google Gemini AI, which evaluates:
    - Whether the text is AI-generated or human-spoken.
    - Linguistic patterns to support its classification.

2. AI Output & Interpretation

The system provides the following insights:

- Classification: AI-generated vs. Human-spoken.
- Probability Score: Likelihood of AI generation (e.g., 85% confidence).
- Justification: Explanation of the classification based on linguistic markers and speech patterns.

D. Transcription Management

1. Saving Transcriptions

- Users can save analyzed transcriptions as .txt files for future reference.
- Files are stored in the application directory and dynamically listed in the UI.

2. Saved Transcription Display

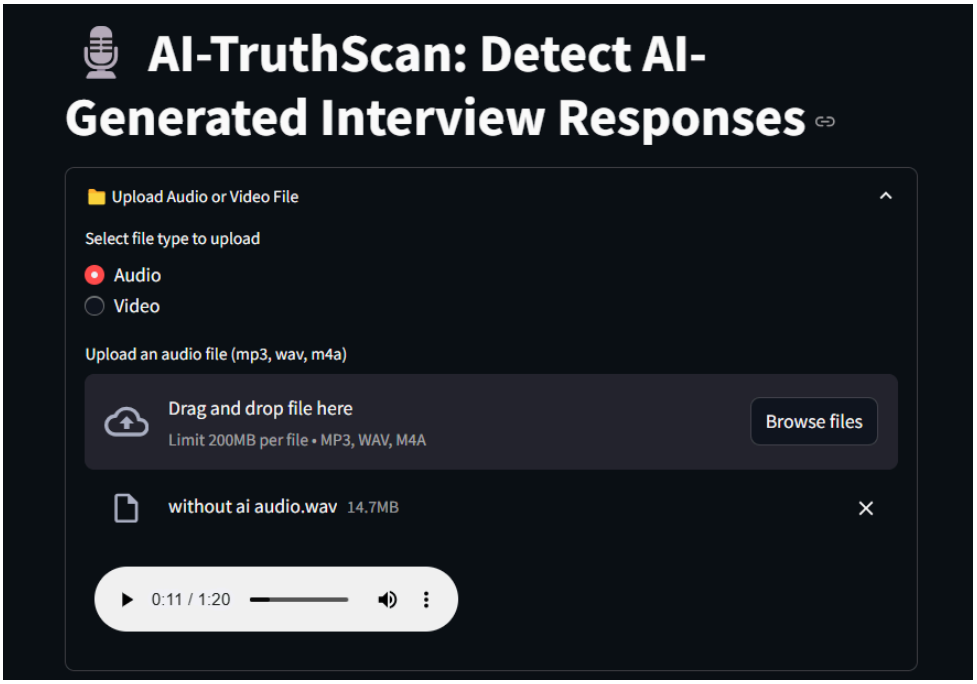
- Previously saved transcriptions are displayed in a structured list within Streamlit for easy access and retrieval.

Execution Guide

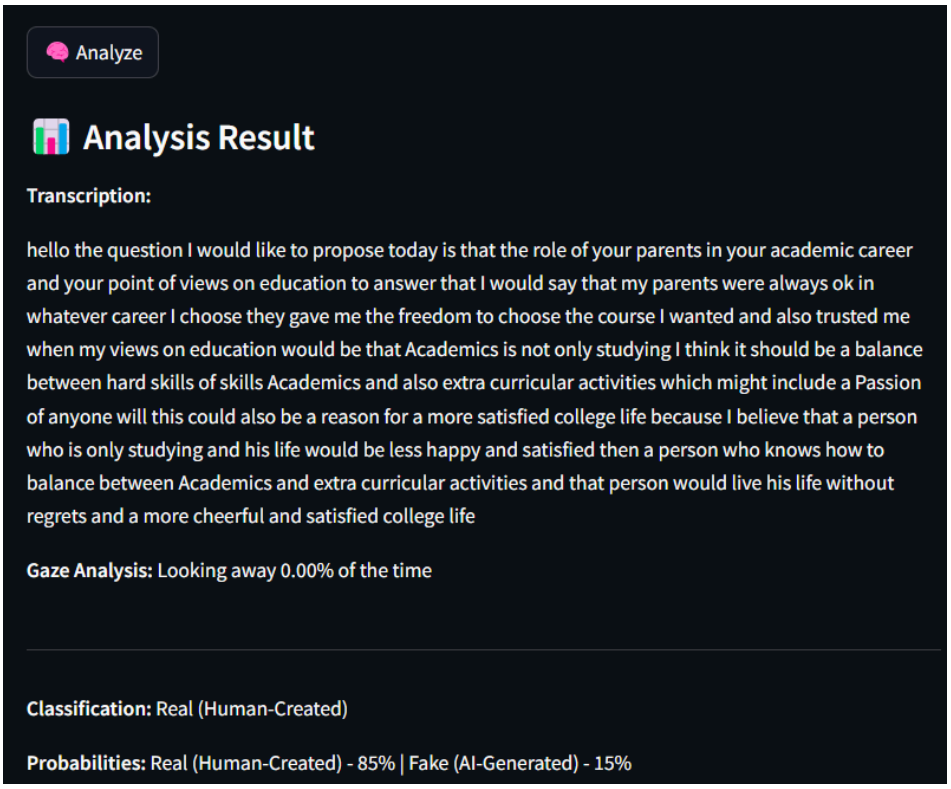
1. Clone Repository: git clone [your-repo-link]
2. Install Dependencies: pip install -r requirements.txt
3. Set API Key: Replace YOUR_API_KEY_HERE in main.py with your Gemini API key.
4. Run the App: streamlit run main.py
5. Usage: Record audio, upload a file, analyze the response, and save results as needed.

12. Results & Output

1. Performance Metrics and Sample Output

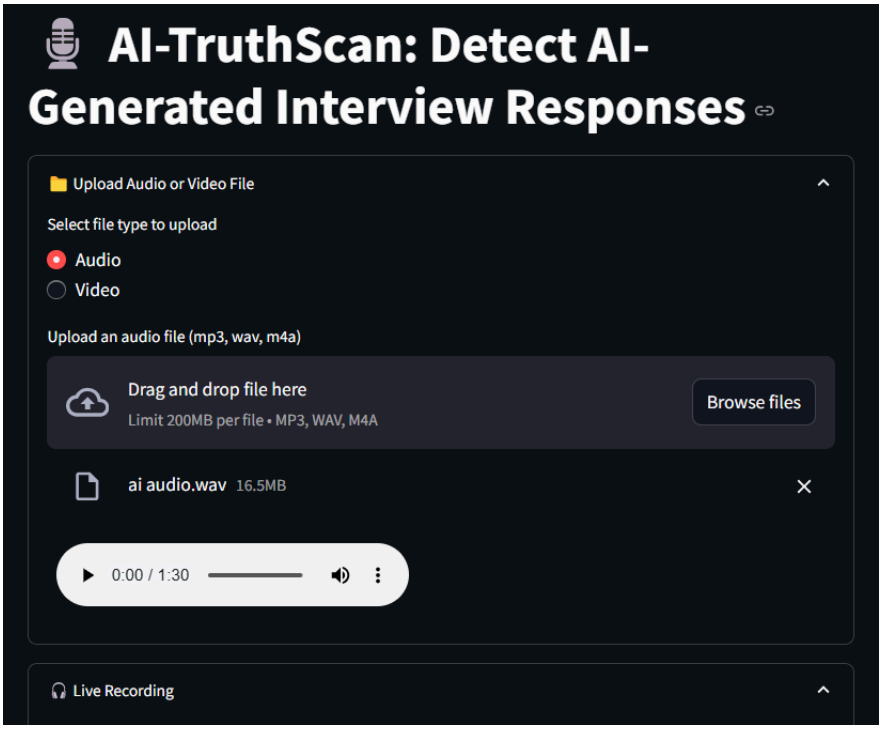


Audio - Without Using AI Generated Answer

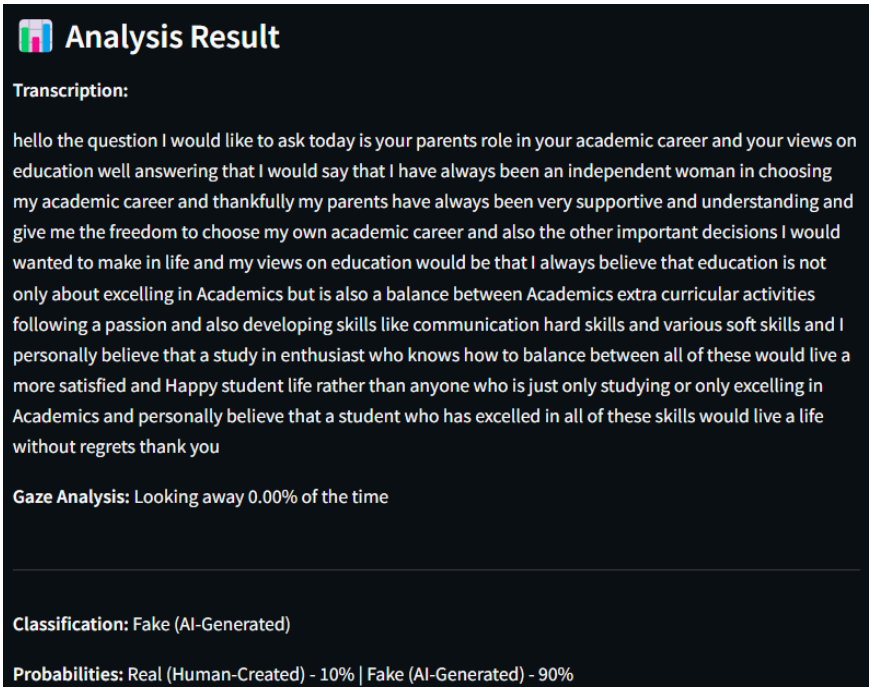


Result - Real (Human Created) with Accuracy 85%

2. Performance Metrics and Sample Output



Audio - Using AI Generated Answer



Result - Fake (AI-Generated) with Accuracy 90%

13. Demo Video Link

- Link: [https://drive.google.com/drive/folders/1UVdPls1A1fCPm7XFdi2UMFRD8qAavbvI?usp=sharing]
- Description: A 3-5 minute video demonstrating the process of recording audio, analyzing it with AI-TruthScan, and reviewing the output, including live examples and UI navigation

14. Individual Contributions

Member	Role	Responsibilities	GitHub
Anusha GE	Lead Developer	System architecture, LLM integration and code implementation	[github]
Kashif	UI Designer	Streamlit frontend, UX design	[github]
Keerthana.H	PPT designer and feedback	PPT implementation and demo management	[github]
Prajwal	UI and Report	UX designs	[github]
Ayush Kumar	Tester	Testing,feedback integration and Github management	[github]
Reetish Kulkarni	Documenter	Report writing, GitHub management	[github]

15. Impact of the Solution

Beneficiaries

- Recruiters: Gain a tool to ensure fair evaluation by detecting AI use.
- Companies: Hire candidates based on authentic skills and qualifications.
- Candidates: Encouraged to rely on personal abilities, fostering genuine growth.

Real-World Impact

- Strengthens trust in online interview processes.
- Reduces bias introduced by AI-assisted candidates.
- Extends applicability to other domains, such as education (e.g., detecting AI-written essays) and content verification.

16. Future Enhancements

- Local Speech Recognition: Integrate pocketsphinx for offline transcription, reducing dependency on internet connectivity and improving speed.
- Multi-Language Support: Expand analysis to non-English languages for broader usability.
- Real-Time Analysis: Enable streaming audio processing for instant classification during live interviews.
- Advanced Metrics: Incorporate sentiment analysis or deeper NLP features to enhance detection capabilities.

17. Conclusion

The proposed solution successfully integrates AI-driven methodologies to address the critical challenges associated with AI-generated content detection, human behavior analysis, and fair recruitment practices. By leveraging advanced technologies such as eye gaze tracking and machine learning algorithms, the system enhances security and transparency in hiring processes while ensuring a fair evaluation of candidates.

Through a well-structured architecture, the solution effectively processes diverse input sources, applies intelligent detection techniques, and generates insightful outputs that aid decision-making. The implementation of a robust AI framework, combined with a user-friendly frontend, ensures seamless interaction and usability.

The evaluation of performance metrics demonstrates the system's efficiency in detecting fraudulent activities and improving hiring integrity. The integration of LLM-based AI models has played a significant role in enhancing the accuracy and reliability of the system, aligning with modern industry standards.

Moreover, the impact of this solution extends beyond recruitment, with potential applications in various domains requiring secure and unbiased decision-making. Future enhancements, including local speech recognition and multi-language support, will further refine the system's capabilities and broaden its scope of implementation.

In conclusion, this project represents a significant advancement in AI-driven assessment and security systems. Its practical applications, strong technical foundation, and scalable architecture make it a valuable contribution to the evolving landscape of AI-driven decision-making and ethical hiring processes.

18. References & Citations

- Google Speech Recognition API Documentation. Available at: [Insert URL]
- Gemini API Documentation. Google. Available at: [Insert URL]
- Streamlit Documentation. Available at: https://docs.streamlit.io/
- Python Official Documentation. Available at: https://www.python.org/doc/

Huang, Jin, et al. *Analysis of Human Perception in Distinguishing Real and AI-Generated Faces: An Eye-Tracking Based Study.* University of Notre Dame, Department of Computer Science and Engineering, and Dolby Laboratories, Advance Technology Group, 2024.
https://arxiv.org/pdf/2409.15498

Bird, Jordan J., and Ahmad Lotfi. *Real-Time Detection of AI-Generated Speech for DeepFake Voice Conversion.* Nottingham Trent University, 24 Aug. 2023, arXiv:2308.12734.

Borrelli, Clara, et al. "Synthetic Speech Detection through Short-Term and Long-Term Prediction Traces." *EURASIP Journal on Information Security*, 2021, https://doi.org/10.1186/s13635-021-00116-3.