Speech Analysis System

A comprehensive system for analyzing speech from video files, providing detailed grammar, pronunciation, and fluency analysis with HTML report generation.

Overview

This system processes video files to analyze speech patterns and language usage. It performs the following operations:

1. Extracts audio from video files
2. Transcribes the audio using OpenAI's Whisper model
3. Analyzes the transcribed text using Together AI's language models
4. Generates detailed HTML reports with analysis results

Prerequisites

- Python 3.8+
- FFmpeg installed on your system
- Required API keys:
    - OpenAI API key
    - Together AI API key

Installation

```
pip install -r requirements.txt
```

Required packages:

```
ffmpeg-python
openai
together
```

Project Structure

```
speech-analysis/
├── audio_extract.py      # Handles video to audio extraction
├── trans_utils.py        # Audio transcription utilities
├── audio_analysis.py     # Speech analysis logic
├── html_utils.py         # HTML report generation
├── main.py              # Main execution script
└── Input Video/         # Directory for input videos
    └── result/          # Generated analysis reports
```

Usage

1. Place your video files in the "Input Video" directory
2. Set up your environment variables:

Run the analysis:

```
python main.py
```

Module Documentation

audio_extract.py

Contains functionality for extracting audio from video files using FFmpeg.

Key Functions:

- `extract_audio_from_video(input_video_path: str) -> str`
    - Extracts audio from video files and saves as MP3
    - Returns path to extracted audio file
    - Creates an 'audio' subdirectory for storage

trans_utils.py

Handles audio transcription using OpenAI's Whisper model.

Key Functions:

- `transcribe_audio_with_openai(audio_file_path: str) -> str`
  - Transcribes audio files to text
  - Returns transcribed text
  - Uses Whisper-1 model with English language setting

audio_analysis.py

Implements speech analysis using Together AI's language models.

Classes:

- `SentenceAnalysis`
  - Dataclass for storing sentence-level analysis
  - Fields:
    - original_sentence: str
    - is_grammatically_correct: bool
    - corrected_sentence: Optional[str]
    - word_analysis: List[dict]
- `AudioAnalyzer`
  - Methods:
    - `analyze_sentence(sentence: str) -> SentenceAnalysis`
    - `get_overall_analysis(text: str) -> str`

html_utils.py

Generates HTML reports from analysis results.

Key Functions:

- `create_html_analysis(overall_analysis: str, sentences_analysis: List[SentenceAnalysis]) -> str`
  - Creates formatted HTML report
  - Includes styling and responsive design
- `save_html_analysis(html_content: str, output_path: str) -> bool`
  - Saves HTML report to file
  - Returns success status

main.py

Main execution script that orchestrates the entire process.

Key Functions:

- `analyze_audio_content(transcription: str) -> tuple[str, List[SentenceAnalysis]]`
    - Processes transcribed text
    - Returns overall and sentence-level analysis
- `main()`
    - Orchestrates the complete analysis workflow
    - Handles logging and error management

Output

The system generates HTML reports containing:

- Overall speech analysis
- Sentence-by-sentence breakdown
- Grammar corrections
- Word-level analysis
- Visual indicators for areas needing improvement

Error Handling

- Comprehensive logging system
- Error catching at each processing stage
- Graceful failure handling with informative messages

Technical Details

- Uses Meta-Llama-3-70B-Instruct-Turbo model for language analysis
- FFmpeg for audio extraction
- OpenAI Whisper for transcription
- Supports multiple video formats (mp4, mkv, avi)

Performance Considerations

- Process videos sequentially to manage resource usage
- Audio files stored in separate directory for organization
- Intermediate results logged for debugging