BTP - 2
Project Report

SPEECH TRANSCRIPTION VERIFICATION

Under the Guidance of Prof. Anil Kumar Vuppala
Mentor - Sai Aakarsh

Source Code - GitHub Repo Link

Srujana Vanka - 2020102005
Shreeya Singh - 2020102011

1 Objective

Develop an efficient Speech Transcription Validation System for Indian languages, incorporating automated audio segmentation, precise transcription generation, and a user-friendly interface with advanced features, including user authentication, transcription review, navigation, editing, synchronized audio playback, virtual keyboard support, and local storage.

2 Introduction

The conversion of speech into text plays a pivotal role in various applications, including automatic captioning, audio search, and speech recognition. However, transcribing lengthy audio files is often time-consuming and prone to errors. The accuracy of these transcriptions holds paramount importance for application effectiveness. Moreover, it's worth noting that, in the context of Indian languages, the availability of accurate ASR (Automatic Speech Recognition) models remains limited. This emphasizes the necessity for an efficient speech transcription verification system that ensures precision while reducing transcriber effort.

Our system addresses these challenges by streamlining the transcription verification process. Our primary objectives include improving transcription accuracy, reducing the workload on transcribers, providing flexibility for multiple languages, ensuring user-friendliness, and implementing robust security measures through user authentication.

3 Project Description

Our project focuses on creating an efficient Speech Transcription Verification System tailored for Indian languages. It accomplishes this by segmenting large audio files, converting segments into text using a speech-to-text API, and presenting them to users for review and correction.

This system features a user-friendly interface with advanced functionalities such as user authentication, enabling secure access and usage. Users can efficiently navigate through transcripts, ensuring seamless review and editing. Additionally, our system supports Indian languages by offering a virtual keyboard, enhancing accessibility and accuracy. Users have the option to save or discard each transcription, and the system ensures that audio segments are reviewed only once, optimizing the verification process. Our project's ultimate goal is to enhance the accuracy and efficiency of speech transcriptions while making the process user-friendly and accessible for transcribers.

4 Dependencies and Technologies

- **Programming Languages:** Python for backend development, JavaScript for frontend.

- **Front-End:** HTML, CSS, and JavaScript for the user interface, enhanced by React for dynamic content rendering.
- **Back-End:** Python for audio splitting and generating transcripts, Node.js for server-side logic and API development.
- **Caching and Data Storage:** LocalStorage API is employed for client-side caching to store transient data, ensuring data continuity.
- **Version Control:** Git was used for version control and collaborative development.
- **Docker:** Docker was employed to containerize the web application for efficient deployment.
- **Other Libraries and Frameworks:** Axios and Fetch API for save/discard, Pydub and WebRTC Voice Activity Detection (VAD) for audio chunking, react-router-dom for client-side routing and navigation, react-simple-keyboard for Telugu keyboard.

5 Project Implementation

Workflow of the project
Our project's core objective is to build a precise and user-centric Speech Transcription Verification System tailored for Indian languages. It all begins with segmenting extensive audio files into manageable parts, with each segment generating a corresponding transcript sent to the frontend platform. Users log in, select their language and the specific audio file for verification, and seamlessly transition to the verification page. Here, they can conduct transcription review, editing, synchronized audio playback, and utilize virtual keyboards, ensuring accuracy and ease of use.

Crucially, the system guarantees that each audio segment is reviewed only once, optimizing the verification process. Users have the flexibility to save or discard transcripts, move to the next one, and even log out, with the system preserving their progress. Additionally, administrators can access login data for system transparency, enhancing accountability.

6 User Interface - Frontend

The front end of our Speech Transcription Verification project is thoughtfully designed to be exceptionally user-friendly, ensuring that individuals with varying levels of technical expertise can easily navigate and utilize its features. With a straightforward and intuitive interface, even users with limited technical knowledge can seamlessly review, edit, and verify transcripts. In the subsequent sections, we will delve into each of these features in-depth.

6.1 Login Page

The login page is pivotal for transparency and data integrity. It has the following functionalities:

- **User Authentication:** The login page is pivotal for transparency and data integrity. It collects transcribers' names and email addresses.
- **File Selection:** Users utilize the login page to select the specific audio file or transcripts they intend to verify, offering flexibility and customization.
- **Language Selection:** The login page also enables users to designate their preferred language. This is used to generate the required virtual keyboard for Indian languages.
- **Data Integrity:** Robust validation checks are incorporated to guarantee data integrity. These checks include verifying the completeness of details and validating the accuracy of entered email addresses.
- **User Flexibility:** The user-centric login page grants users the flexibility to revisit it by logging out, allowing for smooth transitions between different audio files.
- **Admin Access:** It grants admins access to login data, enabling them to monitor and cross-verify information, thereby enhancing transparency and accountability within the system.

6.2 Functionalities and Features of the platform

The main page of our Speech Transcription Verification system is the central hub for users to conduct precise transcript verification efficiently. This section provides an in-depth overview of the various functionalities and features:

- **Display of Transcripts**: Users are presented with the transcripts corresponding to the audio chunks they are verifying.
- **Editing Capabilities:** Users can edit the existing transcripts in real-time, ensuring precision and accuracy in the verification process.
- **Audio Playback:** Users can listen to the audio segments corresponding to the displayed transcripts. Furthermore, they have the ability to pause the playback, and can even control the playback speed for a customized verification experience.
- **Save Button:** The "Save" button allows users to save any changes made to a transcript. Upon clicking, it accesses the transcript number from the LocalStorage and initiates an HTTP POST request via Axios to the local server. The server efficiently processes the request, ensuring the transcript and its corresponding audio file are saved within the local system.

- **Discard Button:** Similar to the Save button, the Discard button works in a similar way. It stores the transcript and its corresponding audio chunk in a discard folder.
- **Next Button:** The "Next" button allows users to seamlessly transition to the next transcript for verification.
- **Transcript Navigation:** The navigation panel allows users to navigate between the transcripts. Users can navigate to any transcript they wish to verify, and efficiently access specific transcripts using the search functionality. To enhance user productivity, the Navigation Panel exclusively displays remaining transcripts, automatically removing those that have been successfully verified.
- **Transcript Counts Display:** There is a dedicated section that provides a quick snapshot of the verification progress. It displays essential counts, including the number of remaining transcripts, saved transcripts, and discarded transcripts. This real-time feedback keeps users informed about their progress.
- **Virtual Keyboard Support:** The main page offers virtual keyboard support, particularly for Indian languages, facilitating ease of input and verification.
- **User-friendly Logout:** The interface also includes a convenient logout button, allowing users to log out at any time. Users can change the audio file they wish to verify or return later to pick up exactly where they left off. This ensures that their progress is never lost and provides flexibility in managing verification tasks.
- **Admin Data Submission:** The system includes a feature to send login data to the admin. This feature compiles logs of transcribers' login data. Admins can access this data, allowing them to review and check for any discrepancies or irregularities in the verification process.
- **Process Resumption:** The system prioritizes seamless verification continuity, allowing transcribers to return to their ongoing tasks without progress loss. Whether logging back in or reloading the application, the system maintains each transcriber's progress, eliminating the risk of duplicate verifications and ensuring that each transcript is reviewed only once.

7 Backend Functionality

The backend of our Speech Transcription Verification system is the engine that drives its core functionalities. It plays a pivotal role in ensuring a smooth and efficient experience for users, from audio processing to data management. In this section, we explore the fundamental components and processes that fuel the backend and, consequently, the entire system.

- **Server Setup:** Our backend's foundation is built on the Node.js and Express.js framework, providing essential support for the system's operations. To start the server, we execute the "server.js" file using the "node" command, ensuring a robust platform for the entire system.

- **Python Scripts for Audio Processing and Transcript generation:** We utilize the child_process module to execute Python scripts seamlessly. Specifically, two scripts, "audio-split.py" and "transcription.py," are initiated using the exec() function. These scripts are instrumental in handling tasks related to audio segmentation and transcription.

- **Python Script for utilizing user's folder for Transcript Verification:** A Python script is executed to convert the transcripts and audio chunks into the required format. This process is essential for aligning the data for verification on the frontend. A duplicate of the folder, now formatted and primed for transcription verification, is then transferred into the "original data" directory.

- **Handling Server Start-Up:** Once the Python scripts have completed their execution, the backend server takes charge. It listens for incoming requests on port 5000, leveraging the body-parser middleware to parse incoming requests with JSON payloads.

- **Handling Post Requests:** The backend efficiently manages incoming post requests, primarily focusing on saving or discarding transcripts and audio chunks.

8 Docker and Containerization

Within the context of our Speech Transcription Verification system, Docker plays a pivotal role in simplifying the deployment process and creating an environment where users, regardless of their technical proficiency, can comfortably engage with the application. Our choice of Docker aligns with our commitment to user-friendliness and accessibility. Additionally, Docker ensures consistency and portability, offering users a predictable and uniform experience across various environments.

9 Major Challenges and Future Scope

9.1 Challenges:

1. **Error Handling:** It's important to implement robust error handling, both on the client and server sides, to ensure a smooth user experience.

2. **Optimization:** Optimizing the application for performance, especially when dealing with large files, is crucial to ensure a smooth user experience.

3. **Cross-Browser Compatibility:** Ensuring that the application works correctly across different browsers can be a challenge, especially when dealing with HTML5 features.

9.2 Future Scope:

1. **User-Friendly UI/UX:** Enhance the user interface for a seamless and intuitive experience.
2. **Multi-Language Support:** Extend support for multiple languages in transcription and chunking.
3. **Feedback Mechanism:** Implement user feedback mechanisms to continuously improve transcription accuracy.
4. **Integration with Storage Solutions:** Integrate with cloud storage services like AWS S3 or Google Cloud Storage for scalable file storage.

10 Acknowledgment

We would like to express my sincere gratitude towards our project guide, Prof. Anil Kumar Vupalla, for his invaluable guidance, encouragement, and support throughout the project. His insights, suggestions, and constant motivation helped us to complete this project successfully. We would also like to extend my heartfelt thanks to our project mentor, Mr. Sai Aakarsh, who went above and beyond to answer our doubts, provide guidance, and help us whenever we faced any difficulties. Their presence and support have been instrumental in shaping our project. We would like to thank the panel members, for evaluating our project. Their feedback and suggestions were invaluable and will helps us in improving our project and taking it to the next level.

11 References

1. Create React App Documentation
   Link: `https://create-react-app.dev/docs/getting-started/`
2. WebRTC Voice Activity Detection (VAD) Python Library
   Link: `https://pypi.org/project/webrtcvad/`
3. Axios - HTTP Client for the Browser and Node.js
   Link: `https://axios-http.com/docs/intro`
4. HTTP Request Methods Explained
   Link: `https://www.freecodecamp.org/news/http-request-methods-explained/`

5. Local Storage in JavaScript: A Complete Guide
   Link: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage

6. react-simple-keyboard for Telugu keyboard
   Link: https://www.npmjs.com/package/react-simple-keyboard/

7. Build a Docker Image and Run a React Application via Docker in Linux
   Link: https://mentorcruise.com/blog/how-to-dockerize-a-react-app-and-deploy-it-easily/

8. Login Authentication to React Applications
   Link: https://www.geeksforgeeks.org/how-to-build-a-react-app-with-user-authentication/