CMPE 492
Project Title

Ümit Can Evleksiz

Advisor:
Lale Akarun
Murat Saraçlar

Contents

Chapter 1

INTRODUCTION

1.1 Broad Impact

1.1.1 Learning Impact

This application provides learners with practical how-to knowledge for speaking standard English, focusing on pronunciation accuracy and clarity. By introducing users to the International Phonetic Alphabet (IPA) and phonetic concepts, the system builds domain familiarity, enabling learners to discuss and understand sound units systematically. The platform incorporates gamification elements and provides fast, friendly feedback to create a safe learning environment where users can practice without fear of judgment.

It is important to note that the objective is not to replace human-to-human interactions in language learning, but rather to provide accessible tools for individuals who may lack access to human tutors, native speaker friends, or formal language education resources. The system serves as a supplementary tool that empowers self-directed learning while acknowledging the irreplaceable value of human interaction in language acquisition.

1.1.2 Academic Impact

This project contributes to the academic community by demonstrating the practical application of fine-tuning deep learning models for specialized phonetic transcription tasks. The work combines core signal processing techniques with modern machine learning approaches, integrating multiple moving parts including audio preprocessing, phonetic transcription, forced alignment, and pronunciation assessment into a cohesive full-stack application. The system showcases how research-grade models can be adapted and deployed in real-world educational contexts, providing a reference implementation for similar pronunciation assessment systems.

1.2 Ethical Considerations

1.2.1 User Speech Data

The system handles sensitive user speech data, which raises important privacy and usage concerns. Audio recordings are used for inference during pronunciation assessment, but users must explicitly opt-in for any potential use of their data in model fine-tuning, voice cloning, or correct pronunciation generation features. Clear consent mechanisms and transparent data usage policies are essential to protect user privacy and maintain trust.

1.2.2 Prescriptive Approach in Linguistics

The application focuses on US English pronunciation, which may appear prescriptive. However, it is crucial to emphasize that all dialects, accents, and language variants are linguistically valid. US English is chosen not to minimize or devalue other accents, but because it is widely adopted and in high demand globally, particularly in professional and academic contexts. The system aims to improve comprehensibility and communication effectiveness rather than eliminate linguistic diversity. Users are encouraged to understand that accent reduction is optional and that the primary goal is clear communication, not accent elimination.

1.2.3 Monetization and Accessibility

Commercialization of educational technology raises concerns about potentially gatekeeping education from economically disadvantaged populations. While the system may include premium features or subscription models, careful consideration must be given to ensuring that core pronunciation assessment capabilities remain accessible. Free tiers, educational discounts, and open-source components can help mitigate barriers to access, ensuring that pronunciation learning tools do not become exclusive to those who can afford them.

Chapter 2

PROJECT DEFINITION AND PLANNING

2.1 Project Definition

This project develops a web-based application to assist English language learners in improving pronunciation through automated speech analysis and personalized feedback. The system enables users to record speech, receive detailed pronunciation assessments, and gain insights into English phonology.

Core functionality includes: (i) capturing and processing user speech recordings, (ii) analyzing pronunciation accuracy at the phonetic level using machine learning models, (iii) comparing actual pronunciation against target acoustic patterns, and (iv) providing educational guidance on English phonology.

The system focuses on English with US accent (potentially including UK), with initial development targeting Turkish-native English speakers. Future milestones will leverage users' native language to better detect phonetic units. The application is designed to be accessible and pedagogically sound, emphasizing comprehensibility over accent elimination.

2.2 Project Planning

2.2.1 Project Time and Resource Estimation

The project is developed as a solo effort with the following resource allocation:

Time Commitment:

- Development time: 15 hours per week
- Project duration: One academic semester (approximately 12 weeks)

- Total estimated effort: 180 hours

Infrastructure Costs:

- Full-stack server hosting: $5 per month
- Serverless PostgreSQL database: $5 per month
- Serverless GPU for ML job queue: $20-30 per month (moderate user activity)
- Total monthly infrastructure cost: $30-40 per month

Additional Resources:

- Development tools and frameworks (open-source)
- ML model training and evaluation datasets
- Audio processing libraries and APIs

2.2.2 Success Criteria

The project will be considered successful upon achieving the following objectives:

Technical Requirements:

- Full-stack secure web application with authentication and authorization
- System uptime of 99% or higher
- Acoustic ML model producing accurate phonetic-level transcriptions independent of prosodic variations
- Difference algorithm comparing actual pronunciation against target patterns with actionable feedback
- Scalable architecture supporting concurrent sessions and asynchronous audio processing

User Experience Requirements:

- Industry-grade UI/UX following modern web standards
- Intuitive workflow for recording, uploading, and reviewing feedback
- Clear presentation of pronunciation analysis results
- Responsive design supporting multiple devices

- Accessible interface adhering to web accessibility guidelines

Functional Requirements:

- Accurate real-time or near-real-time pronunciation assessment
- Educational content integration for English phonology
- Progress tracking and historical analysis
- Support for multiple audio formats and quality levels

2.2.3 Risk Analysis

Several risks have been identified that may impact project success:

Technical Risks:

- **ML Model Performance:** The machine learning model may not achieve desired accuracy, particularly with low-quality audio, background noise, or non-standard accents. Mitigation includes extensive training on diverse datasets, robust preprocessing, and fallback mechanisms for low-confidence predictions.
- **Audio Quality Variability:** Users may submit recordings with varying quality, device capabilities, and environmental conditions. The system must handle this through adaptive preprocessing and quality assessment.
- **Computational Resource Constraints:** Processing requirements may strain serverless GPU resources during peak usage. Load balancing, queue management, and resource scaling will address this.
- **Integration Complexity:** Coordinating multiple system components may introduce integration challenges. Comprehensive testing and modular architecture will mitigate these risks.

Pedagogical and Ethical Risks:

- **Language and Accent Limitations:** The application focuses on English with US accent (potentially UK). Other languages are not supported. Initial development targets Turkish-native English speakers, with future milestones leveraging native language for better phonetic detection. This limitation must be clearly communicated to users to avoid misconceptions about supported languages and accents.

- **Overly Prescriptive Approach:** The system may promote misconceptions about a single "correct" pronunciation, marginalizing linguistic variation. This is addressed by emphasizing comprehensibility over accent matching, acknowledging linguistic diversity, and transparently communicating system limitations.
- **User Expectations vs. Reality:** Users may have unrealistic expectations about pronunciation transformation. Clear communication about the tool's purpose as a learning aid is essential.
- **Data Privacy and Security:** Handling sensitive user audio data requires robust security measures and clear privacy policies. Users must explicitly opt-in for any use of their data in model fine-tuning, voice cloning, or pronunciation generation. Compliance with data protection regulations will be prioritized.
- **Accessibility and Monetization:** Commercial features may create barriers for economically disadvantaged users. Free tiers and educational discounts should be considered to ensure accessibility.

Project Management Risks:

- **Single Developer Constraints:** Limited capacity for parallel development and peer review. Careful time management and feature prioritization are essential.
- **Scope Creep:** The project may expand beyond initial scope. Potential scope creep includes voice cloning and generating correct pronunciation using the user's voice, which are explicitly out of scope. A clear feature prioritization framework and milestone-based development approach will maintain focus.

2.2.4 Team Work (if applicable)

This project is being developed as an individual effort. All aspects including requirements analysis, system design, implementation, testing, and documentation are handled by a single developer.

Chapter 3

RELATED WORK

This chapter reviews relevant research and technologies in speech recognition, phonetic transcription, and pronunciation assessment that inform the design and implementation of this system.

3.1 Speech Foundation Models and Phonetic Transcription

Several open-source deep learning models are available for phonetic transcription. Wav2Vec 2.0 [1] and WavLM [2] provide self-supervised speech representations that can be fine-tuned for phonetic recognition, though they require significant adaptation. For this application, POWSM (Phonetic Open Whisper-Style Speech Model) [13] is best suited, as it is specifically engineered for phonetic transcription with textual transcription context. The model is open to fine-tuning, which is valuable for adapting to Turkish-native English speakers. Connectionist Temporal Classification (CTC) [11] is commonly used for sequence alignment in phonetic transcription tasks.

3.2 Forced Alignment and Assessment

The Montreal Forced Aligner (MFA) [12] is an industry-standard tool for phoneme-level time alignment, supporting English out-of-the-box. Pronunciation assessment employs metrics such as Phoneme Error Rate (PER) and goodness of pronunciation (GOP) scores [16] to evaluate speech quality.

3.3 Commercial Applications

Commercial pronunciation assessment platforms demonstrate practical applications of these technologies. ELSA Speak [3] provides real-time pronunciation feedback for English language learners with personalized AI coaching. Pronounce AI [14] offers instant speech feedback, pronunciation practice, and AI speaking partners for professionals and language learners, supporting both American and British English accents.

Chapter 4

METHODOLOGY

This chapter outlines the technical approach, development methodology, and project management strategies employed in building the pronunciation assessment system.

4.1 Application Architecture

The system is built as an industry-standard web-based application with a focus on user experience. The full-stack architecture is dockerized for consistent deployment and development environments. The frontend utilizes React with TanStack Start for server-side rendering and routing, providing a modern, responsive user interface. The backend leverages TanStack Start's server functions for API endpoints and business logic.

Data persistence is handled through Neon, a serverless PostgreSQL database, providing scalable and reliable storage for user recordings, metadata, and analysis results. Audio files are stored in Cloudflare R2, a cost-effective object storage solution that integrates seamlessly with the application infrastructure. The entire stack is containerized using Docker, enabling reproducible deployments across development, staging, and production environments.

Modern server state management and caching are implemented using TanStack Query, which provides efficient data fetching, automatic background refetching, and intelligent caching mechanisms. This ensures optimal performance and user experience by minimizing unnecessary network requests and maintaining responsive UI updates.

Deployment is automated through GitHub Actions CI/CD pipelines, which handle building Docker images, running tests, and deploying to production platforms such as Railway or Fly.io. These platforms provide seamless Docker container orchestration, automatic scaling, and integrated monitoring capabilities.

4.2 Machine Learning Approach

The machine learning pipeline follows an experimental and iterative approach. Multiple open-source phonetic transcription models, including Wav2Vec 2.0, WavLM, and POWSM, are evaluated to identify the best-performing solution for the specific use case. Performance is assessed based on accuracy, inference speed, and compatibility with the target user population (Turkish-native English speakers).

Once a model is selected, fine-tuning is performed when necessary to improve performance on domain-specific data. This process involves collecting and curating training data from target users, adapting the model architecture, and iteratively improving accuracy. However, a significant challenge encountered is the lack of comprehensive documentation and standardized inference providers for many open-source phonetic transcription models, requiring extensive experimentation and custom implementation.

4.3 Development Methodology

The project employs parallel development tracks, leveraging prior professional software development expertise while building new knowledge in signal processing and deep learning domains. Signal processing components, including audio preprocessing, quality assessment, and feature extraction, are developed alongside deep learning model integration and fine-tuning pipelines.

Experimentation and prototyping are conducted using Python notebooks (Jupyter/IPython), enabling rapid iteration on digital signal processing (DSP) techniques, visualization of audio features, and deep learning model evaluation. These notebooks serve as both development tools and documentation of the experimentation process, capturing insights and learnings that inform the final implementation.

The development process emphasizes modularity and separation of concerns, with clear boundaries between frontend, backend, database, and machine learning components. This architecture facilitates independent development and testing of each component while maintaining system integration.

Chapter 5

REQUIREMENTS SPECIFICATION

5.1 Functional Requirements

Functional requirements define the specific behaviors and capabilities the system must provide. These are organized by feature area:

5.1.1 User Authentication and Authorization

- **FR-1.1: User Registration** The system shall allow new users to register accounts with email and password authentication or social sign-in (Google, Git, etc.).
- **FR-1.2: User Login** The system shall provide secure login functionality for registered users.
- **FR-1.3: Session Management** The system shall maintain user sessions and support automatic session expiration after periods of inactivity.
- **FR-1.4: User Profile** The system shall associate all recordings and progress data with authenticated user accounts.
- **FR-1.5: Admin Access** The system shall provide restricted administrative access for content management functions such as creating, editing, and deleting target texts and practice materials.

5.1.2 Speech Recording and Audio Input

- **FR-2.1: Browser-Based Recording** The system shall enable users to record speech directly through the web browser using the device microphone.
- **FR-2.2: Real-Time Audio Visualization** The system shall display real-time waveform visualization during recording to provide visual feedback.
- **FR-2.3: Recording Controls** The system shall provide start, stop, and pause controls for audio recording.
- **FR-2.4: Audio Format Support** The system shall accept audio recordings in WebM/Opus format from browsers and convert them to WAV format (16-bit, 16kHz, mono) for processing.
- **FR-2.5: Recording Duration Limits** The system shall enforce reasonable duration limits to prevent excessive resource consumption.

5.1.3 Audio Quality Assessment

- **FR-3.1: Basic Quality Check** The system shall perform basic audio quality assessment to ensure recordings meet minimum standards for analysis.
- **FR-3.2: Quality Feedback** The system shall provide feedback to users when recordings fail quality checks, suggesting improvements when possible.

5.1.4 Phonetic Analysis and Transcription

- **FR-4.1: Phonetic Transcription** The system shall generate International Phonetic Alphabet (IPA) transcriptions of user speech using acoustic ML models.
- **FR-4.2: Prosody Independence** The system shall produce phonetic transcriptions that are independent of prosodic variations (stress, intonation, rhythm).
- **FR-4.3: Phone-Level Alignment** The system shall align phonetic transcriptions with audio timestamps at the phone level.
- **FR-4.4: Target Comparison** The system shall compare user phonetic transcriptions against canonical IPA transcriptions of target pronunciations.
- **FR-4.5: Error Classification** The system shall classify pronunciation errors into categories: substitution, deletion, and insertion.

5.1.5 Pronunciation Feedback and Assessment

- **FR-5.1: Error Identification** The system shall identify and highlight specific phoneme-level pronunciation errors.
- **FR-5.2: Error Visualization** The system shall present pronunciation errors in a clear format showing the target vs. actual pronunciation.
- **FR-5.3: Overall Score** The system shall calculate and display an overall pronunciation accuracy score based on error rate.

5.1.6 Educational Content and Phonology Guidance

- **FR-6.1: Basic Guidance** The system shall provide basic guidance on English phonology concepts relevant to identified errors.
- **FR-6.2: Target Text Management** The system shall allow administrators to create, edit, and manage target texts for pronunciation practice.

5.1.7 Progress Tracking and History

- **FR-7.1: Recording History** The system shall maintain a history of user recordings with timestamps and metadata.
- **FR-7.2: Basic Progress Tracking** The system shall allow users to view their recording history and basic progress information.

5.1.8 Data Storage and Management

- **FR-8.1: Audio Storage** The system shall store processed audio files (16-bit, 16kHz, mono WAV) in persistent storage for long-term access.
- **FR-8.2: Metadata Storage** The system shall store recording metadata (user ID, file path, timestamps, analysis results) in a PostgreSQL database.

5.1.9 Asynchronous Processing

- **FR-9.1: Job Queue** The system shall process audio analysis tasks asynchronously using a job queue system to handle concurrent requests.
- **FR-9.2: Processing Status** The system shall provide status updates to users during audio processing.
- **FR-9.3: Error Handling** The system shall handle processing failures gracefully, providing clear error messages.

5.1.10 User Interface and Experience

- **FR-10.1: Responsive Design** The system shall provide a responsive user interface that works effectively on desktop and mobile devices.
- **FR-10.2: Intuitive Navigation** The system shall provide clear navigation and workflow for recording and viewing results.
- **FR-10.3: Visual Feedback** The system shall provide immediate visual feedback for user actions, including loading states and error notifications.
- **FR-10.4: Language Support** The system interface shall clearly communicate that it supports English with US accent (potentially UK), targeting Turkish-native speakers, and that other languages are not supported.

-

Chapter 6

DESIGN

6.1 Information Structure

ER Diagrams.

6.2 Information Flow

Activity diagrams, sequence diagrams, Business Process Modeling Notation.

6.3 System Design

Class diagrams, module diagrams.

6.4 User Interface Design (if applicable)

Chapter 7

IMPLEMENTATION AND TESTING

7.1 Implementation

The implementation follows the architecture and methodology outlined in previous chapters. The source code is organized in a structured repository [8] with clear separation between the full-stack application, documentation, and experimental work.

The main application codebase is located in the `app/` directory [4], containing the React frontend, TanStack Start backend, database schemas, and API endpoints. The application structure follows modern web development practices with TypeScript for type safety, component-based architecture, and modular design patterns.

Signal processing experiments and deep learning model evaluations are conducted in the `sig/exp/` directory [9], which contains Jupyter notebooks for audio preprocessing, feature extraction, model training, and visualization. These notebooks document the experimentation process and serve as reference implementations for the production pipeline.

The repository includes comprehensive documentation in the `doc/` directory, with learning materials, technical notes, and project reports. The main README file [7] provides an overview of the project structure, setup instructions, and development guidelines. Environment variables and configuration are documented to ensure reproducible deployments across different environments [5].

7.2 Testing

7.3 Deployment

The application is deployed using Docker containers, ensuring consistency across development, staging, and production environments. The deployment process is automated through GitHub Actions CI/CD pipelines [6], which handle building Docker images, running test suites, and deploying to production platforms.

Production deployment is hosted on Railway [15] or Fly.io [10], both of which provide seamless Docker container orchestration, automatic scaling based on traffic, and integrated monitoring capabilities.

The Docker configuration includes multi-stage builds for optimization, environment variable management, and health checks for container monitoring. The deployment process supports zero-downtime updates through rolling deployments and automatic rollback mechanisms in case of deployment failures.

Building instructions, environment setup, and deployment procedures are documented in the repository README [7], providing clear guidance for developers and system administrators. The deployment architecture supports horizontal scaling to handle increased user load and ensures high availability through redundant container instances.

Chapter 8

RESULTS

Chapter 9

CONCLUSION

Citation examples: [?], [?], [?], [?]. Footnote example[1]. Referring to figures and tables: Figure 9.1. Table 9.1.

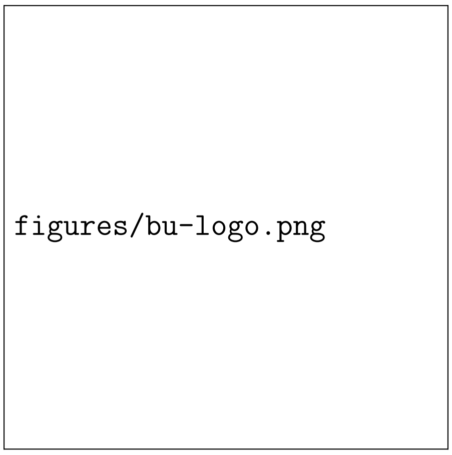

Figure 9.1: Figure caption

Table 9.1: Table caption.

	Header 1	**Header 2**
Row 1	100	300
Row 2	200	400

[1]More details: `https://www.cmpe.boun.edu.tr/`

Bibliography

[1] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460, 2020.

[2] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

[3] ELSA Corp. Elsa speak: Ai-powered english pronunciation coach. `https://elsaspeak.com/en/`, 2024. Accessed: November 2025.

[4] Evleksiz, Ümit Can. Application source code directory. `https://github.com/umitcan07/senior/tree/main/app`, 2024. Full-stack application codebase. Accessed: November 2025.

[5] Evleksiz, Ümit Can. Environment configuration. `https://github.com/umitcan07/senior/tree/main/app`, 2024. Environment variables and configuration files. Accessed: November 2025.

[6] Evleksiz, Ümit Can. Github actions ci/cd workflows. `https://github.com/umitcan07/senior/tree/main/.github/workflows`, 2024. Continuous integration and deployment pipelines. Accessed: November 2025.

[7] Evleksiz, Ümit Can. Project readme. `https://github.com/umitcan07/senior/blob/main/README.md`, 2024. Project documentation and setup instructions. Accessed: November 2025.

[8] Evleksiz, Ümit Can. Senior project repository. `https://github.com/umitcan07/senior/`, 2024. Boğaziçi University Computer Engineering Senior Project. Accessed: November 2025.

[9] Evleksiz, Ümit Can. Signal processing experiments directory. `https://github.com/umitcan07/senior/tree/main/sig/exp`, 2024. Jupyter notebooks for DSP and deep learning experiments. Accessed: November 2025.

[10] Fly.io. Fly.io: Deploy app servers close to your users. `https://fly.io/`, 2024. Global application deployment platform. Accessed: November 2025.

[11] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. *Proceedings of the International Conference on Machine Learning*, pages 369–376, 2006.

[12] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. *Proceedings of Interspeech*, pages 498–502, 2017.

[13] Yifan Peng, Muhammad Shakeel, Yui Sudo, William Chen, Jinchuan Tian, Chyi-Jiunn Lin, and Shinji Watanabe. Owsm v4: Improving open whisper-style speech models via data scaling and cleaning. *arXiv preprint arXiv:2506.00338*, 2024.

[14] Pronounce Inc. Pronounce ai: Ai-powered speech feedback and pronunciation practice. `https://www.getpronounce.com/`, 2024. Accessed: November 2025.

[15] Railway. Railway: Deploy and scale applications. `https://railway.app/`, 2024. Docker container deployment platform. Accessed: November 2025.

[16] Silke M. Witt and Steve J. Young. Phone-level pronunciation scoring and assessment for interactive language learning. *Speech Communication*, 30(2-3):95–108, 2000.

Appendix A

SAMPLE APPENDIX

Contents of the appendix.