BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

SPEECH TECHNOLOGY APPLICATION IN PRONUNCIATION TRAINING AND FOREIGN LANGUAGE LEARNING

VYUŽITÍ ŘEČOVÝCH TECHNOLOGIÍ PŘI VÝUCE VÝSLOVNOSTI CIZÍCH JAZYKŮ

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR — **Bc. ŠTĚPÁNKA BAROTOVÁ**
AUTOR PRÁCE

SUPERVISOR — **IGOR SZÖKE, Ing., Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2020

Brno University of Technology
Faculty of Information Technology

Department of Computer Graphics and Multimedia (DCGM) Academic year 2019/2020

Master's Thesis Specification

Student: **Barotová Štěpánka, Bc.**
Programme: Information Technology Field of study: Intelligent Systems
Title: **Speech Technology Application in Pronunciation Training and Foreign Language Learning**
Category: Speech and Natural Language Processing

Assignment:

1. Get familiar with the basics of comparison of two audio examples (using DTW).
2. Study provided class implemented in JavaScript, which compares two audio examples.
3. Design a method for training of pronunciation and evaluate it.
4. Improve the method. Aim at feedback to a user.
5. Discuss achieved results and future work.
6. Make an A2 poster and a short video presenting your work.

Recommended literature:

- According to supervisor's recommendation

Requirements for the semestral defence:

- Items 1, 2 and part of item 3.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor: **Szőke Igor, Ing., Ph.D.**
Head of Department: Černocký Jan, doc. Dr. Ing.
Beginning of work: November 1, 2019
Submission deadline: June 3, 2020
Approval date: May 27, 2020

Abstract

This diploma thesis deals with automatic English pronunciation assessment and error detection based on the Dynamic Time Warping (DTW) algorithm. It focuses on the improvement of an existing pronunciation training application and it proposes three areas of improvement: user interface, algorithm and corrective feedback. After various methods used for pronunciation assessment are discussed in the first part, the new design is introduced, the proposed system is described and three sets of experiments are performed. The experiments focus on phoneme-level error detection, syllable-level primary stress error detection and word-level intonation assessment and they are designed to be able to provide corrective feedback to the user. The last part of the thesis describes how all three areas of improvement were tested.

Abstrakt

Tato diplomová práce pojednává o využití algoritmu Dynamic Time Warping (DTW) pro automatické hodnocení výslovnosti anglického jazyka. Práce se zaměřuje na vylepšení již existující aplikace pro výuku výslovnosti, a to ve třech oblastech: uživatelské rozhraní, samotný algoritmus a korektivní zpětná vazba uživateli. První část se věnuje přehledu technik používaných v této oblasti, následně je představen nový design uživatelského rozhraní, popsán navržený systém a experimenty. Experimenty se zaměřují na problematiku detekce chyb na úrovni fonémů, na detekci chyb v primárním důrazu na úrovni slabik a na hodnocení intonace na úrovni slov. Všechny použité metody jsou navrženy tak, aby poskytovaly korektivní zpětnou vazbu uživateli. V poslední části je popsáno, jak byly všechny tři vylepšené oblasti aplikace otestovány.

Keywords

speech recognition, pronunciation assessment, pronunciation error detection, pronunciation training, foreign language learning, dynamic time warping, stress error detection, intonation assessment

Klíčová slova

rozpoznávání řeči, hodnocení výslovnosti, detekce chyb ve výslovnosti, trénování výslovnosti, výuka cizích jazyků, dynamic time warping, detekce chyb v důrazu, hodnocení intonace

Reference

BAROTOVÁ, Štěpánka. *Speech Technology Application in Pronunciation Training and Foreign Language Learning.* Brno, 2020. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Igor Szöke, Ing., Ph.D.

Rozšířený abstrakt

S rostoucím rozvojem počítačových technologií se násobí příležitosti k jejich praktickému využití. Světová globalizace způsobuje vysokou poptávku po efektivní výuce cizích jazyků a právě zde se otevírá prostor pro využití řečových technologií k vytvoření alternativního způsobu výuky jazyků: levnějšího, potenciálně přesnějšího a dostupného prakticky kdykoli a odkudkoli. V posledních desetiletích se na trhu objevilo mnoho aplikací pro výuku jazyků, ale co stále zůstává ve svých počátcích, je oblast výuky správné výslovnosti.

Výuka výslovnosti je komplexní téma, které zahrnuje porozumění problematice zpracování řeči, zvládnutí problému rozpoznání výslovnostních chyb a dobrou aplikaci pedagogických přístupů. Zejména přesné rozpoznání výslovnostních chyb, jak fonetických, tak na úrovni prozódie, je největší výzvou aktuálního výzkumu v této oblasti.

Cílem této práce je vylepšit dodanou webovou aplikaci pro výuku výslovnosti. Většina přístupů pro detekci chyb ve výslovnosti využívá metod založených na systémech automatického zpracování řeči (ASR - Automatic Speech Recognition), které využívají komplexních modelů řeči a vyžadují trénování na velkém množství řeči nativních i nenativních řečníků. Dodaná aplikace ovšem pracuje na jiném principu a využívá algoritmu Dynamic Time Warping (DTW).

DTW pracuje se dvěma nahrávkami řeči, s referenční promluvou a s promluvou studenta, a zarovnává jejich jednotlivé řečové segmenty (slova, fonémy) na sebe. Pro hodnocení kvality výslovnosti algoritmus využívá skóre akustické podobnosti dvou segmentů. Jeho výhodou je jednoduchost a rychlost, nenáročnost na data, a možnost nasazení i na mobilní zařízení, na kterých by ASR systém běžet nemohl, a to bez nutnosti implementace klient-server architektury. Nevýhodou DTW je zejména jeho neschopnost zachytit celou variabilitu správných výslovností.

Dodaná aplikace byla vylepšena ve třech směrech: bylo vytvořeno zcela nové uživatelské rozhraní, byl vylepšen a rozšířen algoritmus pro hodnocení výslovnosti a jako poslední byl navržen a implementován způsob korektivní zpětné vazby uživateli, který poskytuje zpětnou vazbu potřebnou k tomu, aby se uživatel mohl z chyb učit.

Co se týče samotného algoritmu hodnocení výslovnosti, byly v rámci práce provedeny tři sety experimentů: detekce vložení a vypuštění fonémů, detekce chyb v primárním důrazu a detekce intonačních chyb. Detekce vložení a vypuštění jednotlivých fonémů ve slovech byla navržena jako detekce určitých vzorů ve výsledné cestě DTW matice, ale přesnost takového algoritmu se ukázala být velmi nízká. Hlavní důvod, proč nelze fonémové chyby jen za pomocí DTW algoritmu detekovat, je, že DTW provádí deformace referenční i studentovy nahrávky tak, aby se na sebe za jakýchkoli okolností zarovnaly. Takové deformace způsobují v mnohých případech nepřesná zarovnání fonémů na sebe, a to zejména pokud se v daném místě nachází nějaká výslovnostní chyba. Čím více výslovnostních chyb, tím méně přesné zarovnání fonémů na sebe, a tím méně pravděpodobné, že algoritmus chyby detekuje.

V případě detekce primárního důrazu ve slovech bylo z algoritmu DTW využito pouze finální zarovnání řečových segmentů. To bylo využito pro zarovnání energií extrahovaných z obou nahrávek a energie jednotlivých slabik byly porovnávány vzhledem k celému slovu. Takovým způsobem bylo docíleno poměrně slušné přesnosti detekce chyb v primárním důrazu. Podmínkou funkčnosti takového algoritmu je předem provedená segmentace referenčních nahrávek na slabiky (ta byla v této práci dodána vedoucím). Navržený algoritmus lze dále zlepšit přidáním základního tónu do příznaků použitých k detekci. Podle odborných zdrojů je totiž důraz ve větě charakterizován nejen zvýšenou energií, ale také právě základním tónem, případně i prodlouženou délkou zdůrazněných fonémů. V takovém případě by již detekce primárního důrazu vyžadovala kombinaci více příznaků, a pravděpodobně

by pak bylo vhodnější použít pro detekci některý algoritmus strojového učení, který ovšem vyžaduje trénování na větším množství dat.

Hodnocení intonace ve větě bylo provedeno podobným způsobem jako hodnocení důrazu. Cílem tohoto experimentu bylo zjistit, zda intonaci lze vůbec nějakým způsobem za pomocí DTW zarovnání hodnotit. Jako příznak pro detekci byl zvolen základní tón (F0). Hodnocení intonace bylo založeno na tzv. intonačních vzorech používaných v lingvistice, jako například intonace stoupavá nebo klesavá. Tyto vzory byly pro každé slovo v referenční i testovací promluvě určeny pomocí jednoduchého klasifikačního algoritmu a dvojice vzorů porovnány mezi sebou. Takto bylo možné zjistit, kde student udělal chybu v intonaci oproti referenční nahrávce, kterou měl napodobit. Hodnocení intonace lze tedy provést pouze na základě nějaké referenční intonace, nikoli globálně. Přesnost navrženého algoritmu se ukázala jako ucházející a celkově může být řečeno, že hodnotit intonaci z DTW zarovnání a za pomocí základního tónu lze. Jednoznačným návrhem pro zlepšení tohoto algoritmu je zdokonalit oblast klasifikace intonačních vzorů ze segmentů křivky základního tónu. Navržený algoritmus je založen pouze na jednoduchých rozhodovacích pravidlech, které berou v potaz tvar křivky základního tónu, její sklon či celkovou změnu výšky tónu v rámci slova. Lepší klasifikace by zcela jistě mohlo být docíleno natrénováním samostatného klasifikátoru intonačních vzorů na určitém množství anotovaných dat pomocí některého z algoritmů strojového učení.

Algoritmus implementovaný do výsledné aplikace pak obsahuje pouze oblast detekce chyb správné fonetické výslovnosti na úrovni slov a detekci chyb v důraze na úrovni slabik. Implementace hodnocení intonace byla mimo rozsah této práce. Algoritmus byl navržen tak, aby nad ním bylo možné postavit mechanismus korektivní zpětné vazby. Ten byl navržen jako seznam chybných slov, ve kterém každé slovo obsahuje i popis konkrétní chyby a možnost porovnat si referenční výslovnost se svou vlastní.

Všechny tři vylepšené oblasti aplikace byly otestované. Uživatelské rozhraní i korektivní zpětná vazba byly uživateli hodnoceny velice pozitivně. Na druhou stranu, z testování samotného algoritmu expertem na výuku anglického jazyka vyplynulo, že jeho přesnost není dostatečně vysoká.

Budoucím směřováním by mohla být právě snaha o další vylepšení hodnotícího algoritmu. Z provedených experimentů se zdá, že algoritmus DTW se dá poměrně slušně použít pro hodnocení prozódie řeči (důrazu a intonace). Ovšem ukázalo se, že pro detekci výslovnostních chyb na úrovni fonému DTW není vhodný. Proto by výsledné aplikaci nejvíce prospělo, kdyby byl DTW algoritmus nahrazen algoritmem založeným na tradičním ASR systému.

Tento text obsahuje stručný úvod do technologií rozpoznání řeči v kapitole 2. Kapitola 3 pak obsahuje přehled existujících přístupů pro hodnocení výslovnosti. V kapitole 4 je popsána dodaná aplikace a redesign jejího uživatelského rozhraní. Sběr dat je popsán v kapitole 5. Dále text pokračuje kapitolou 6, která se zabývá vylepšeními hodnotícího algoritmu. Návrh a implementace korektivní zpětné vazby je uvedena v kapitole 7 a poslední kapitola 8 popisuje, jak byla aplikace otestována.

Speech Technology Application in Pronunciation Training and Foreign Language Learning

Declaration

I hereby declare that this Master's thesis was prepared as an original work by the author under the supervision of Ing. Igor Szöke, Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Štěpánka Barotová
June 1, 2020

Acknowledgements

I would like to thank Ing. Igor Szöke, Ph.D., for the supervision of my work and for all the advice he provided. I also want to thank Mrs. Zaina Ramji for her professional help.

Contents

Chapter 1

Introduction

Increasing globalization contributes to a greater demand for effective foreign language learning methods. Even though learning from human teachers is still considered to be the most effective and the most reliable, computer technology is slowly being incorporated into foreign language learning, too.

As opportunities and development in the field of computer science and technology continue to grow, speech technology can be utilized to create a more accessible and cheaper solution for language learning. In the last two decades, there has truly been a great interest in utilizing speech technology in the field of foreign language learning and recently, computer-assisted pronunciation training in particular has received considerable attention. There is plenty of existing research in this area and a number of commercial applications as well. However, correctly assessing pronunciation quality and detecting pronunciation errors is often a challenging task even for a human teacher. Therefore, computers are no exception and there are still numerous opportunities for improvement in this area, especially in providing corrective feedback to the student.

The main goal of this thesis is to improve an existing pronunciation training application provided by supervisor, so that it is able to give a useful corrective feedback for the English language learner. The pronunciation assessment algorithm used in this application is based on the Dynamic Time Warping (DTW) algorithm.

Originally, the DTW algorithm is used for automatic alignment of two utterances. As far as segmentation of the reference utterance to words and phonemes is provided, the algorithm can be also used to obtain segmentation to words and phonemes of the student's utterance. Then, acoustic similarity of the pairs of utterances can be computed on top of the alignment. However, the original algorithm itself is unable to provide any detailed information necessary to give the student a meaningful feedback so that the student can learn effectively. The aim of this work is to change that.

This work deals with three areas of improvement of the original application: user interface, algorithm and corrective feedback. Regarding algorithm improvements, three subproblems are explored in this work: prosodic error detection, stress error detection and intonation assessment.

Firstly, Chapter 2 gives a brief overview of the speech recognition technology. Then the text continues with a description of its application in foreign language pronunciation training in Chapter 3, where particular methods used for assessment and error detection, including the above-mentioned DTW method, are explained.

The second part of this thesis deals with the actual improvements of the application. Chapter 4 describes the original application in detail and discusses how the user interface

was redesigned. Furthermore, Chapter 5 describes how data from native and non-native English speakers was collected using the application. In Chapter 6, improvements of the pronunciation assessment algorithm are discussed. Finally, in Chapter 7, design of the new corrective feedback is described, and Chapter 8 describes how the application was tested.

Chapter 2

Speech Recognition

Speech recognition is the process of translating spoken speech into words and sentences using computers. The process is performed by automatic speech recognition (ASR) systems and an overview of the speech technology will be provided in this chapter, because most of the existing research concerning automatic pronunciation assessment is based on ASR.

This chapter describes how automatic speech recognition systems and their components work. Before this chapter dives into the details of speech recognition systems, it gives a brief overview of how speech is produced in human vocal tract and how specific types of speech sounds are formulated.

2.1 About Speech

In order to understand speech recognition technologies and o be able to utilize them successfully for the purposes of foreign language learning systems, it is important to know where human speech originates and how different sounds are pronounced. This section is mainly based on information from [21] and explained from the point of view of the English language.

On a physical level, speech is a continuous signal (except for pauses) but on a psychological level, speech is perceived by humans as made up of discrete sounds. However, it is only because of their language knowledge that people are able to divide the continuous speech signal into discrete units.

The production of speech is air-driven, completely dependent on the stream of air that goes through the vocal cords, the pharynx and out of the mouth or nose. Different sounds are created by obstructing the air stream in different ways.

Vocal cords change the sound from **voiceless** (for instance *s*, *f* or *p*) when they are completely open to **voiced** (such as *z*, *v* or *r*) when they are closed and vibrate.

If the air from the lungs is allowed to enter the nasal tract at the soft palate (velum), the sounds pronounced are called **nasal**, such as the first sounds of the words such as *meal* or *Neal*. This means that if a person has a cold, accuracy of computer speech algorithms is negatively affected because of the person's distorted articulation of nasal sounds.

If the air is not allowed to go to the nasal tract, the resulting speech sounds are called **oral**. Most English sounds are oral. Air passing through the oral tract can be altered by the tongue, teeth and lips. Their various positions result in different speech sounds.

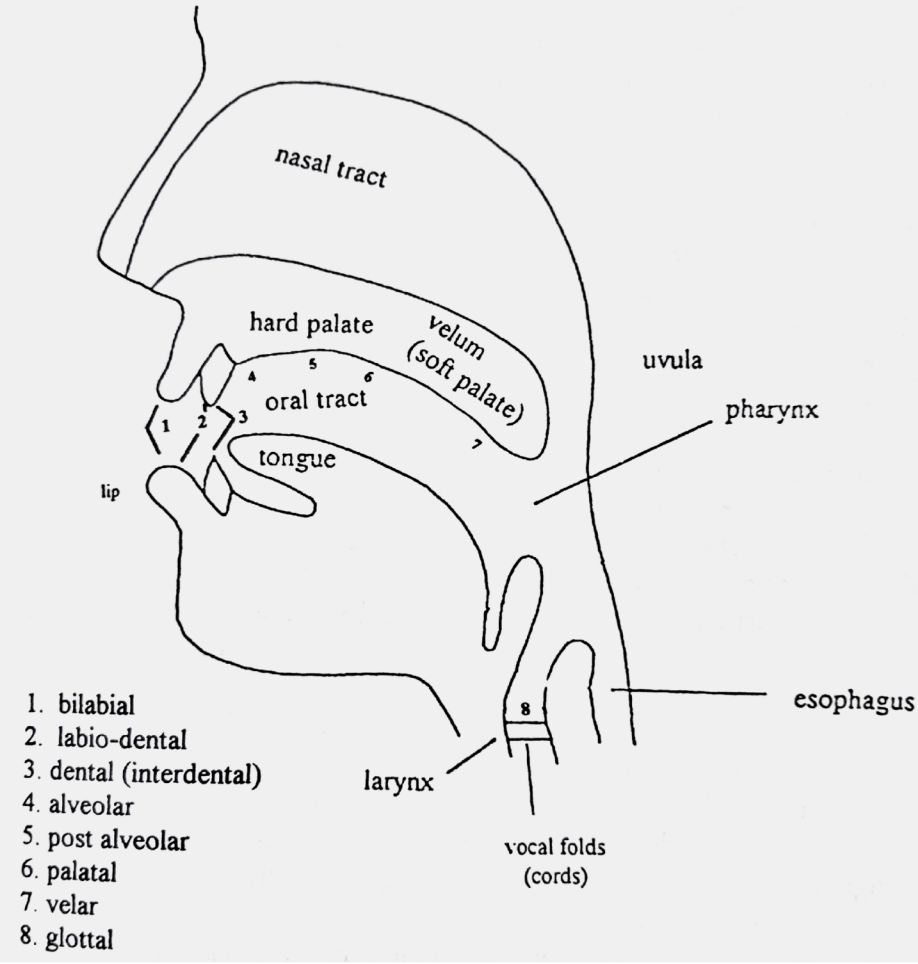


Figure 2.1: Vocal tract and places of articulation. Source: [21].

2.1.1 Types of Speech Sounds

Phonetics studies the two groups of speech sounds: consonants and vowels. *Articulatory phonetics* focuses on the physical process of human speech production and divides the parts of speech-producing anatomy into active articulators (vocal cords, soft palate and lips) and passive articulators (teeth, alveolar ridge, hard palate and velum). The whole human vocal tract is displayed in Figure 2.1.

Consonants

Consonants are defined as sounds that are produced when articulators touch or come close to each other. For instance, *b*, *s* or *r*. The particular sound of a consonant is determined by three factors: the **place of articulation** (the place where the obstruction is placed), the **manner of articulation** (how articulators are positioned and whether they are active or passive during the obstruction) and **voicing**. Voicing refers to the state of the vocal cords; when they are separated, a **voiceless** consonant is produced, whereas when they are together, a **voiced** consonant is produced.

Based on these factors, there are a lot of consonant types, such as a fricative, a stop, a dental consonant and others. All types are summarized in Appendix B.

Vowels

Vowels are sounds that are produced when articulators do not block the airstream. Vowels in English include *a*, *e*, *i*, *o*, *u* and sometimes *y*. These sounds resonate in the vocal tract and tongue and lips are used to alter the shape of the vocal tract so that it produces different sounds. Vowels are classified by the position of the tongue and the lips.

Based on tongue height (i.e. its distance from the roof of the mouth), we can distinguish **high**, **high-mid**, **mid**, **low-mid** and **low** vowels. Tongue backness (i.e. distance of the tongue from the teeth) allows us to discriminate **front**, **central** and **back** vowels.

When pronouncing vowels, lips may be either **rounded** or **unrounded**.

Diphthongs are sounds that begin with one vowel and gradually change into another one (*bite*, *bout*, *boil*).

2.1.2 Prosody

Intonation, stress, length and pitch are all *prosodic* features of speech. In some cases, their use changes meaning, in other cases they are used to convey emotional or other information. **Intonation** is the overall pitch contour of a sentence or phrase. For instance, some questions may have a rising intonation.

Syllables in English have a primary **stress** (the largest stress in a word) and a secondary stress (the second largest stress in a word). Stress is significant on the level of individual phrases (*a lighthouse keeper* vs *a light housekeeper*) as well as on the sentence level.

While vowel length is distinctive in some languages, it is not the case in English. The same applies for the relative pitch on a syllable. In tone languages, pitch may change the meaning of a word, but not in English.

These speech properties have to be taken into account in speech recognition systems to make speech recognition effective. Incorrect intonation or stress in the English language can be an indicator of a pronunciation error. Therefore, it might be meaningful to try to detect such errors using automatic pronunciation assessment algorithms.

2.1.3 Acoustic Phonetic Features

Acoustic phonetic features are the physical properties of sound waves of speech studied by *acoustic phonetics*, such as wave frequency or amplitude. Frequency determines the pitch of the sound in hertz (Hz) and amplitude determines the sound intensity in decibels (dB). Intensity of human speech ranges from whispering at 30 dB to loud shouting at 80 dB.

Frequencies are one of the most important features used to differentiate between certain sounds. The reason for that is that each vowel contains certain frequency bands that are much higher in energy than other frequency bands in the spectrum. These significant frequency bands are called **formants** and are different for each vowel type. The first formant (F1) of a sound is the lowest frequency band that is significantly high in energy or amplitudes. There are formants called F2, F3 and higher but usually only the first two or three are significant for speech recognition.

Formants can be detected from the sound by spectrographic analysis. The result is a spectrogram, a graph with time on the x-axis and frequencies on the y-axis, and the amplitude or energy is represented by a colour scale (the darker the colour, the higher the amplitude). Example of a spectrogram and formants can be seen in Figure 2.2.

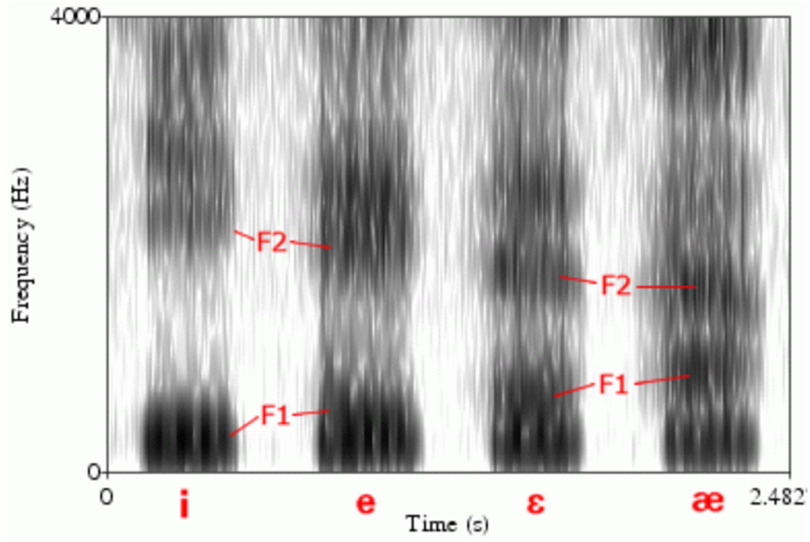


Figure 2.2: Example of a spectrogram and formants. Source: [22].

2.1.4 Phonology

In the context of languages, a **phoneme** is a basic unit in the sound system of a language. Each word can be transcribed to a sequence of phonemes. Even though one phoneme describes one sound, it may also sound slightly different based on the context (its two surrounding phonemes). Also, one phoneme may have different alternative pronunciations (allophones) that do not create a meaningful change in the word. However, two allophones in one language can represent different phonemes in another language. This is exactly the point where native language phonology affects the types of pronunciation errors made by a learner of a second language. Certain sounds in the native language of the learner might be perceived as the same phoneme, whereas in the target language, they create a significant change. A similar issue may arise with sounds of the target language that do not exist in the learner's native language.

The following section describes the basic components and functionality of an ASR system.

2.2 Automatic Speech Recognition Systems

The goal of an ASR system is to find the sequence of words that most likely corresponds to the input speech signal. This section is mainly based on [3].

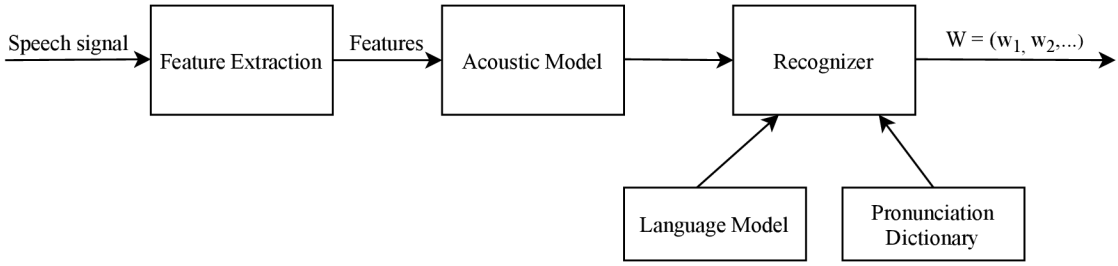


Figure 2.3: Main components of an ASR system.

2.2.1 Feature Extraction

The first step of the speech recognition process is feature extraction. After the input speech signal is preprocessed in order to decrease the influence of background noise, different distortions etc., features are extracted from the speech signal. Resulting features are later used in the rest of the speech recognition process.

Features are extracted from short time intervals (frames) of the signal using windowing. Usually, the window length is 10-25ms [3]. There is a wide range of features, such as MFCC (Mel Frequency Cepstrum Coefficients), LPC (Linear Predictive Coefficients), filter-bank coefficients, or bottleneck features and others. Basically, MFCCs are suitable for ASR systems based on HMM-GMM (Hidden Markov Models with Gaussian Mixture Models) acoustic models, and filter-banks are more suitable for acoustic models that utilize neural networks. Bottleneck (BN) features are extracted from one hidden layer of a multi-layer perceptron neural network trained to predict monophone states. BN features compress useful information, such as phoneme classes, while suppressing noise, speaker ID and other information. Bottleneck features are used in this work and the corresponding bottleneck feature extraction process is displayed in Figure 2.4.

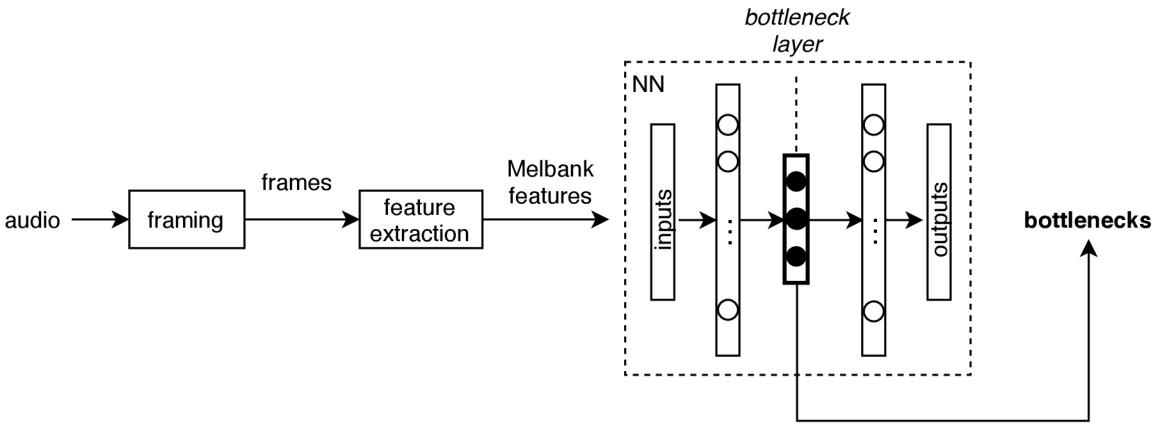


Figure 2.4: Schema of bottleneck feature extraction.

2.2.2 Acoustic Model

Once features are extracted, they are used to determine what the uttered sound to which they correspond was. Acoustic model is used to model individual monophones or triphones. A monophone is a sound of a phoneme independent on the context of the phoneme. Triphone, on the other hand, is a context-dependent sound. For example, sound *a* in the word *candle* will constitute a different triphone than in the word *handle* because its surrounding phonemes are different.

Acoustic model is trained using the same features that are extracted from the input speech signal. It outputs probabilities for each monophone or triphone that the input feature vector corresponds to the model.

Mostly, acoustic model would be created using HMMs but neural networks can be used, as well. In the following section, functionality of HMM-based acoustic models will be described.

Hidden Markov Models

HMM can be understood as a finite state machine with states and transitions between them that emit output symbols (phonemes). The HMM basically tries to find the sequence of transitions between states that most likely generate the sequence of phones $Y = (y1, y2, ...)$ that the speaker has uttered.

A HMM is defined by: [3]

1. A set of states. Usually one or three states per monophone/triphone are used.
2. Transition probabilities for all combination of states that are linked with a transition. They are determined in the HMM training phase.
3. A probability density function describing an output symbol for each transition. A probability density function determines the probability $P(y_i|t)$ of the current input y_i being emitted at transition t. Probability density functions can be modelled by GMMs or neural networks and are trained on the ASR features.

The reason why HMMs are called „hidden“ is that the actual sequence of states is unknown. The only known information are the *a posteriori* probabilities.

The output of the acoustic model at each time unit is the probability $P(y_t, s_t|s_{t-1})$ of each possible transition from (an unknown) state s_{t-1}. These probabilities are inputs to the recognizer, which decides on the final sequence of phonemes and words. [3]

Language Model

A language model is specific for each language and it provides some *a priori* knowledge about the language, such as the words most frequently used together. It provides a wider context that can significantly improve speech recognition.

Language models often use *N-grams*, i.e. sequences of $N-1$ preceding words which can be used to estimate the probability of a word w_i. The value of N is usually between one and three. [3]

Thus, if the acoustic model, for some reason, is not able to recognize one word within a sentence, the language model can compensate for that by providing probabilities based on the previous recognized word, so that the word can be inferred based on the knowledge from the language model.

Pronunciation Dictionary

Pronunciation dictionary defines how each word of the language is transcribed to a sequence of phonemes. There may be alternative pronunciations of the same word so the dictionary may contain more correct pronunciations for a single word.

2.2.3 Recognizer

The recognizer is the last piece of an ASR system. Its task is to decide on the final output sequence of words $W = (w1, w2, ...)$ using a sequence of probabilities from the acoustic model, and knowledge from the language model and pronunciation dictionary.

Probabilities from the acoustic model are stored and when all observations are processed, the best path (the one that has the highest probability) is found using a search algorithm. The final path corresponds to the most likely sequence of phonemes and words that fit the input speech signal. [3]

Chapter 3

Automatic Pronunciation Assessment and Error Detection

This chapter deals with the application of speech recognition technology in automatic foreign language learning and pronunciation training. It begins with explaining the current opportunities and challenges in automated foreign language learning, continues with an overview of methods used for automated pronunciation scoring and error detection, and finally, gives a detailed description of the method used within this work.

3.1 Types of Pronunciation Errors

Learners of foreign languages make pronunciation errors for two main reasons: either they are unaware of the correct pronunciation, or their attempt to pronounce the utterance correctly is influenced by their native language (L1) phonology. Native language transfer means that there may be sounds in the foreign language (L2) that do not exist in the learner's L1, or there may be two sounds that are perceived as different in L2 but are not distinguished in the learner's L1. [11]

However, the concept of pronunciation error, is rather difficult to define, as there is not a clear definition of right or wrong pronunciation. Thus, instead of quantifying pronunciation, a scale ranging from unintelligible speech to native-sounding speech is used. Does the utterance sound almost native-like, or does it sound strongly nonnative? [31]

There are two types of pronunciation errors: *phonemic* (or segmental) and *prosodic*. Nevertheless, all errors are closely linked. Pronunciation is an important part of foreign language learning, as it is vital for the learner to sound intelligible in order to be understood.

3.1.1 Phonemic Errors

Phonemic errors occur in relation to specific phonemes. There are three main types of phonemic errors that may be distinguished, namely:

- Phoneme substitution. The learner substitutes the correct phoneme for a different one.
- Phoneme deletion. The learner completely omits a phoneme from the utterance.
- Phoneme insertion. The learner inserts an extra phoneme in between two others.

Yet another case of a phonemic error is called „phoneme mispronunciation“. This error takes place when the correct phoneme is more or less pronounced, but it sounds differently enough that it is possible to tell that the speaker has an accent. [31]

Logically, correct pronunciation from the phonemic point of view is the basics of language learning and it is necessary to sound intelligible. However, phonemic correctness alone is not enough to master pronunciation of a language.

3.1.2 Prosodic Errors

Prosodic errors are usually measured on the sentence or phrase level. These errors are made in elements such as stress, rhythm, duration, timing, pauses or intonation. [31]

As stated in [1], prosody even has equal or greater effect than phonemic correctness on the speaker's comprehensibility. Teaching only the correct phonemic pronunciation does not significantly improve comprehensibility [9], while teaching prosody does [6].

Stress

There are two levels of stress: *pitch accent* (or *sentential stress*) and *lexical stress* [19].

Pitch accent concerns the correct placement of stress on the most prominent syllable within a sentence. It is characterized by an increase in pitch followed by fall in pitch [25]. It is usually influenced by a wide range of factors, such as type of sentence, emotional status of the speaker, context, or the speaker's intention. Therefore, pitch accent is very difficult to assess automatically, which is why most of the following information will be related to lexical stress.

Lexical stress, focuses on stressing the correct syllable within a word according to word stress patterns that are defined in dictionaries. It is characterized as an increase in duration and energy but not in pitch [25].

English has a stress-timed rhythm. That means that unstressed syllables between consecutive stressed syllables are reduced (their pronunciation changes), in contrast with syllable-timed rhythm, where stressed syllables have a longer duration but there is no vowel reduction (for example, French or Italian). [18]

Lexical stress patterns in English are not predictable. In other words, there are various stress patterns that differ for each word. Many other languages have predictable stress patterns (for example every first syllable of a word is stressed), but this is not the case in English. For example, while the word *citizen* has primary stress on the first syllable, *relationship* has primary stress on the second syllable, plus its last syllable carries a secondary stress. [17]

There are three levels of stress a syllable can have: **primary**, **secondary** and **non-stressed** [14]. Primary stressed syllables carry the major pitch change and there is only one primary stressed syllable in each tonal group (phrase). Usually, primary stressed syllables are found in words that carry some important information. Secondary stressed syllables are also stressed but less than primary stressed syllables. Finally, non-stressed syllables do not bear any stress at all. In English, every word has one or more stressed syllables but not all of them are realized phonetically [18]. For example, function words such as prepositions or articles are usually not stressed within a sentence.

According to [14], stressed syllables in English are characterized by power level, pitch, duration add vowel quality. Researchers in [17] agree but do not mention vowel quality.

Because vowels are longer than consonants, they carry most of the information related to stress [26]. Consequently, since vowel sounds can be changed depending on whether the vowel is stressed or not, stress errors can cause lower comprehensibility.

Stress errors that cause the greatest issues in comprehensibility are errors in primary stress. The learner may misplace the primary stress from one syllable to another and in some cases, the error may even change the very meaning of the word.

Intonation

Intonation is strongly related to the fundamental frequency in speech. Fundamental frequency is basically the frequency of vocal cords.

Intonation is used to convey emotional meaning, express attitude, communicate intentions and it helps the listener recognize grammatical structures of a sentence. It may even act as marker of personal or social identity (mother, lover, doctor, lawyer, ...). Very often, intonation may suggest exactly the opposite meaning than the words used by the speaker. [2]

In linguistics, a number of sentence intonation patterns exists for the English language. There are two categories of intonation patterns: falling tones and non-falling tones. Among **falling tones** we can find high fall (HF), low fall (LF), rise-fall tones; the **non-falling tones** comprise high rise (HR), low rise (LR), mid-level and fall-rise tones. In English, there is basically no rule for which sentence type has which intonation pattern, though some generalizations exist. A fall is typical for statements, exclamations, wh- questions, yes-no questions or commands. A fall-rise typically appears in statements and commands including polite corrections, partial statements or warnings. A rise usually occurs in encouraging statements, wh- questions, commands, yes-no questions or interjections, for instance. [29]

Wrong intonation patterns can distinguish a native speaker from a non-native speaker. However, if a speaker misapplies an intonation pattern, then, instead of noticing an error, it is more likely that their audience will understand the utterance differently than what the speaker's intention was [5].

3.2 Computer-Assisted Pronunciation Training

There are many existing pronunciation training computer programs. In the past, they often did not provide corrective feedback to indicate specific weaknesses (i.e. concerning the place of the error and advice on what should have been done differently), which means this software could only aid the assessment of pronunciation, not fully replace a human rater. If a student uses such software, much of the pronunciation learning task is left for them and that is called *self-assessment of pronunciation.*

However, a study on the reliability of self-assessment of pronunciation conducted in [7] confirms that self-assessment is not a reliable method of assessing pronunciation. In the majority of cases, L2 learners need of a teacher's help in identifying inaccurate sounds. Thus, it is possible to conclude that pronunciation teaching software without a reliable, specific and detailed assessment of the student's pronunciation will not be effective.

In other words, Computer-Assisted Pronunciation Training (CAPT) software that has a reliable method of assessing the student's pronunciation *and* is able to give a specific feedback, can significantly improve independent foreign language pronunciation learning. Until then, pronunciation training programs will leave most of the space for self-assessment to be done by the independent learner, which, again, is not effective in teaching pronunciation.

Nowadays, CAPT applications that provide corrective feedback to the user already exist. For instance, Elsa Speak (https://elsaspeak.com) or SpeechAce (https://www.speechace.com/), to name a few.

3.2.1 Advantages

Traditional methods of foreign language learning require the teacher's full attention devoted to a student, especially when it comes to pronunciation training. What a student needs the most in learning pronunciation, is an immediate feedback and correction. Computer-based pronunciation training (CAPT) programs that can reliably detect specific errors and provide meaningful corrective feedback have the potential to offer a much cheaper alternative to a human teacher. Moreover, CAPT programs can be accessible at any time and potentially at any place [11]. However, the necessary requirement for such a system to prove successful is its ability to accurately identify the exact errors within words [30].

3.2.2 Challenges

Today, the main challenge in CAPT systems is how to achieve accuracy high enough so that the feedback is not misleading to the student in any way, so that the system can be trusted and perceived as reliable [8]. This issue is particularly challenging for those sounds that are often substituted by L1 sounds or mispronounced. Moreover, false positives (telling the student there was an error where there was none) may be harmful to the student's learning process.

There are two terms which are not to be confused with each other: *pronunciation scoring* and *pronunciation error detection*. Pronunciation scoring concerns rating a sentence, phrase, word or a phoneme, according to how well it was pronounced. It can be used for assessing pronunciation fluency, for example.

Pronunciation error detection, when combined with a corrective feedback, can be more useful to the student, because it deals with detecting specific errors at the phoneme level. On the other hand, it is much more difficult than pronunciation scoring, because the shorter the unit, the greater the variability of the pronunciation assessment [31]. The challenge here is, how to detect such errors precisely.

There are, of course, many other challenges in automated pronunciation error detection. Ideally, it is desired for the CAPT system to be independent of the learner's L1 (L1-independence), to be text-independent (able to work with unconstrained speech), to be able to detect both phonemic and prosodic errors, and to be capable of providing meaningful corrective feedback. [31]

There is a lot of existing research on specific sub-problems of pronunciation error detection but, as mentioned in [31], "a successful system will require a combination of many different techniques".

An overview of existing methods used for pronunciation error detection is provided in the next section.

3.3 Methods

Research on automated pronunciation error detection and pronunciation scoring started in the 1990's [31]. Although it slowed down at the beginning of the 21st century, research interest was renewed a couple of years later. Most of the work primarily dealt with specific

sub-problems of the pronunciation error detection task but some of them tried to combine several approaches to build more robust solutions.

3.3.1 Classification of Methods

In general, the approaches to automatic error detection in pronunciation learning can be classified into several categories. There are **L1-dependent** and **L1-independent** approaches. Then, we may distinguish **text-dependent** and **text-independent** approaches. Finally, we differentiate **likelihood-based** methods and **classifier-based** methods.

L1-dependent vs L1-independent Approaches

In order to minimize the commercial implementation challenges, it is preferable to have an L1-independent solution. However, L1-dependent systems tend to achieve higher accuracy. L1-dependent solutions are designed to address the most common errors for the specific L1-L2 combination. [8]

Text-dependent vs Text-independent Approaches

Text-dependent solutions are bound to a specific learning material. For instance, letting the student exercise their pronunciation only within a limited number of sentences or words. Text-independent solutions are able to work with unconstrained speech and would be desirable, for example, for conversational learning systems. Not much research has been done in the latter approach.

Likelihood-based vs Classifier-based Approaches

Likelihood-based approaches have been used since the 1990's and methods that fall into this category include the HMM-based log-likelihood posterior score, which has become a standard in pronunciation scoring, and is based on the GOP (Goodness of Pronunciation) score. The advantage of these methods is that they are easy to compute and L1-independent. However, they are not capable of identifying the type of error that has occurred.

Classifier-based approaches, on the other hand, are able to do so, using classifiers trained on specific pairs of phonemes (correct-incorrect) that represent corresponding error types. [31]

3.3.2 Pronunciation Metrics

There are many metrics that can be used to measure pronunciation. Metrics for measuring **phonemic elements** include, for example: [31]

- Phone-level log-likelihood scores or GOP
- Acoustic-phonetic features
- Spectral features (formants)
- Phoneme or vowel durations

Metrics for **prosodic elements** (intonation, stress, fluency...) may include: [31]

- Distances between stressed and unstressed syllables

- Energy (power) within a word
- F0 contours (pitch)
- Rate of speech (words per minute) and articulation rate (phonemes per second)
- Measures of silence: mean pause time, silences per second, mean long silence duration
- Mean phoneme duration

3.3.3 Log-Posterior Probability Score

This score is computed for each phone segment and it is based on the acoustic model of the ASR system. If the score of a particular phone segment falls below a predetermined threshold, the phone is marked as mispronounced.

This score is based on the original GOP (Goodness of Pronunciation) algorithm, as defined in [30]. The GOP score was used as a standard method for a long time. Later, this type of score was further improved using new technologies, such as neural networks.

In order to compute the score, the canonical transcription (the expected content of the utterance, for example the phone-level transcription of a sentence that the learner should read) has to be known, as well as the previously obtained phonetic segmentation of the speaker's utterance. The segmentation can be generated by force aligning the student's speech against the canonical transcription.

First, for each frame y_t of the segment corresponding to a canonical phone q_i, the posterior probability $P(q_i|y_t)$ is computed as follows:

$$P(q_i|y_t) = \frac{p(y_t|q_i)P(q_i)}{\sum_{j=1}^{M} p(y_t|q_j)P(q_j)} \tag{3.1}$$

It represents the probability density of the frame y_t using the model of the q_i phone. In other words, the probability that the frame y_t actually belongs to the canonical phone q_i. The conditional phone distributions $p(y_t|q_i)$ can be modelled by Gaussian Mixture Models (GMMs), or neural networks trained with native speech. $P(q_i)$ is the prior probability of the phone q_i, which can be determined using frequency analysis, and M represents the number of phone models present in the HMM.

The posterior score for the phone segment q_i is then defined as the average of the logarithm of the segment's posterior probability:

$$\rho(q_i) = \frac{1}{d_i} \sum_{t=t_{i0}}^{t_{i0}+d_i-1} logP(q_i|y_t) \tag{3.2}$$

Where d_i is the duration of the phone q_i and t_{i0} is the first frame of this phone segment. Sources: [11], [10].

The Original Definition of the GOP Score

This is the original definition of the GOP (Goodness of Pronunciation) score from [30]:

$$\begin{aligned} GOP(p) &\equiv |log(P(p|O^{(p)}))|/d \\ &= \left| log(\frac{p(O^{(p)}|p)P(p)}{\sum_{q\in Q} p(O^{(p)}|q)} \right| /d \end{aligned} \tag{3.3}$$

Acoustic model is used to determine the likelihood $p(O^{(q)}|q)$ that an acoustic segment $O^{(q)}$ corresponds to phone q (Q is the set of all phone models in the acoustic model). $P(p|O^{(p)})$ is the posterior probability that the segment $O^{(p)}$ uttered by the speaker corresponds to the phone p. $P(p)$ is the prior probability of the phone p, given by the language model of the ASR system. And finally, d is the number of frames in the segment $O^{(p)}$.

Assuming that all phones in Q are equally likely and that the sum can be approximated by its maximum, the equation 3.3 can be simplified to the following:

$$GOP(p) = \left| log(\frac{p(O^{(p)}|p)}{max_{q \in Q} p(O^{(p)}|q)} \right| / d \tag{3.4}$$

If the GOP score is above a predefined threshold, it is rejected as a mispronunciation.

Adjustments

Some approaches used different thresholds for each phoneme and found that with this adjustment, the pronunciation error detection has improved ([30], [11]). Researchers in [30] explain that the reason for this is that some phonemes tend to have lower log-likelihoods than others, which in turn means that a higher threshold needs to be used for them.

In some cases, researchers decided to increase the detection accuracy by modelling the conditional phone distributions $p(y_t|q_i)$ by two models for each phone, one trained with the correct pronunciation, the other trained with an incorrect pronunciation ([11], [30]).

The approach described in [10] uses the score to rate the whole sentences.

Other researchers found out that HMMs are not powerful enough to distinguish between very similar sounds and decided to use classifiers to differentiate particular pairs of phonemes corresponding to specific pronunciation errors instead. [24]

Finally, this type of score can only be used to detect phonemic errors, so methods that want to deal with prosodic errors have to extend this approach by other speech features.

Some of these approaches will be described in the following subsections.

3.3.4 Extending the Recognition Network with Models of Incorrect Pronunciation

Acoustic phoneme models may be extended by models trained on incorrect pronunciation. That way, there will be two models for each phone; one trained with the correct, native-like, pronunciation, the other trained with mispronounced phonemes. In order to train the models of mispronounced speech, a phonetically transcribed database of nonnative speech is needed.

To detect a mispronunciation, the **log-likelihood (LLR) score** is computed for each phone q_i, using both pronunciation models, λ_C (correct) and λ_M (mispronounced). [11]

$$LLR(q_i) = \frac{1}{d} \sum t = t_0{}^{t_0+d-1} [logp(y_t|q_i, \lambda_M) - logp(y_t|q_i, \lambda_C)] \tag{3.5}$$

The score is normalized by the phone segment length in frames (d). Mispronunciation is detected when the score is above a predefined threshold. When thresholds are specific for each phone, it leads to error detection improvement, same as in the posterior score.

Research in [30] suggests that modelling each phone by two models improves the detection of those errors that are affected by the phonology of the learner's L1. That includes especially substitutions of L1 sounds for sounds of the target language that do not exist in

the L1. Incorporating models of L1 phonemes into the recognition network increases the accuracy of detecting such errors.

3.3.5 Features for Prosodic Error Detection

ASR-based scores described above can detect phonemic errors. Their advantage is that they are easy to obtain from the ASR system and can be calculated in a similar manner for each phone. However, it is not possible to use them to detect prosodic errors.

In order to detect prosodic errors, more general speech features have to be used. These suprasegmental features include pitch, duration or intensity (energy) and more. In this section, some of them are explained. Some of these features can also be used for improving the phonemic error detection.

Energy

All energy features are derived from the basic signal power (energy), that is defined in the following manner:

$$E = \sum_{n} |x(n)|^2, \tag{3.6}$$

where $x(n)$ is a signal frame.

Existing work on prosodic error detection uses raw energy, log energy or **root mean square (RMS) energy** [1], defined as:

$$RMSE = \sqrt{\frac{1}{N} \sum_{n} |x(n)|^2}, \tag{3.7}$$

An example of a feature derived from the energy feature is **rate of rise (ROR)**. ROR is the main feature of a method introduced in [28] and can be used to detect bursts (abrupt rises) of energy within segments. The research in [24] suggests that this measure can be used to discriminate plosives from fricatives, along with the **zero-crossing rate** measure that is defined as the number of times the signal crosses the x-axis.

The ROR measure is based on the RMS energy and is computed using a 24 ms window that is shifted over the acoustic speech signal every 1 ms. For each window, the window energy is measured by computing the logarithm of the log RMS energy over the window.

Pitch

The pitch of a signal may be described by **fundamental frequency (F0)**, which is the frequency of vocal cords. There is a number of methods for F0 extraction, such as the autocorrelation pitch detector of the cepstral F0 detection approach [33] but details of these methods are outside of the scope of this thesis. F0 contours are in Hertz (Hz). Final F0 contours can be used to assess stress or intonation.

Duration Scores

Duration of various segments of speech (phonemes, syllables) may also be measured.

[1] https://musicinformationretrieval.com/energy.html, accessed 2020-01-22.

To compute the segment duration score defined in [10], duration of all phone segments in a sentence is first obtained from the alignments and then it is normalized by the rate of speech (ROS). Finally, log-probability of the normalized duration is computed, which uses discrete duration distribution of the particular phone. That has to be trained on native speech.

$$D = \frac{1}{N} \sum i = 1^{N} log[p(f(d_i)|q_i)] \tag{3.8}$$

N is the number of phones in the segment we want to measure the score for. d_i is the segment duration in frames and $f(d_i) = d_i * ROS$. ROS is the average number of phones per time unit.

There can be other types of duration scores, for examples durations of the three hidden states of a triphone, or durations normalized for the articulation rate. [8]

Timing Scores

In English, stressed syllables are usually lengthened and others shortened. To measure rhythm and stress, some measures of timing can be used. The timing scores defined in [10] use a measure called **syllabic period**, which is defined as the time between the centres of vowels, normalized on rate of speech (ROS). The score is then defined as the average of the log-likelihoods of the normalized syllabic periods over a sentence and it is calculated using a discrete distribution of syllabic periods trained on native speech.

$$E_n = 20 \times log_{10}(\frac{rms_n}{0.00002}) \tag{3.9}$$

ROR is defined as the derivative of energy E_n:

$$ROR_n = \frac{E_n - E_{n-1}}{\delta t} \tag{3.10}$$

3.3.6 Classifier-Based Methods

Classifier-based approaches to pronunciation error detection combine several features together, in order to build more robust and accurate solutions. In classifier-based methods, ASR-based features are usually combined with some of the suprasegmental features mentioned earlier. Apparently, ASR-based scores seem necessary to be a part of the final algorithm, because using the other suprasegmental features alone does not lead to sufficient accuracy. The most significant characteristic of classifier-based approaches is that they are able to provide corrective feedback to the user, instead of only detecting the error.

In [10], neural networks and classification/regression trees are trained on a transcribed nonnative database in order to predict pronunciation scores a human rater would give. This approach only deals with pronunciation scoring, not with pronunciation error detection.

Research described in [8] trained a different SVM (Support Vector Machines) model for each phone in order to discriminate between specific phoneme pairs.

Finally, in [24], algorithms aimed at discriminating specific phones from each other are introduced. There is a decision tree based on acoustic-phonetic features, and the LDA (Linear Discriminant Analysis) algorithm, which assigns weights to all features to find the linear combination that best discriminates the phone classes.

3.3.7 Stress Error Detection

This section contains an overview of existing approaches to stress error detection.

As already mentioned in section 3.1, stressed syllables are best characterized by energy, pitch and duration. According to [23], the fundamental energy (F0) is not as reliable correlate as energy or duration but still, fundamental energy significantly improves detection accuracy. Most of the existing work takes F0 into account.

Approach in [14] uses two-stage recognition using HMMs trained on F0, energy and the MFCC features. In the first stage, the presence of stress in syllables is detected. In the second step, stress level is identified for stressed syllables.

In [17], machine learning algorithms are used to assess lexical stress patterns and detect lexical stress errors. They use acoustic features such as F0, log value of amplitude and normalized duration.

Researchers in [26] use F0, F0 slope, RMS energy and duration to detect syllable stress errors using a Bayes classifier. In [19], lexical stress and pitch accent detection method using Deep Neural Networks (DNNs) is suggested.

Most approaches use HMMs, NNs or other machine learning algorithms to detect stress errors. However, there is one approach ([2]) that takes into account the DTW algorithm. This approach uses log energy features and directly compares them using correlation between the test and reference energy curves. This is done on top of phoneme alignments from the DTW algorithm. Since this work utilizes DTW, too, and is built using lightweight technologies, a similar approach will be used here.

3.3.8 Intonation Error Detection

There is not much existing research on intonation error detection. Instead of robust intonation assessment which would require complex contextual information such as the speaker's emotional state or language nuances in meaning, existing approaches rather focus on comparing the student's intonation with a reference intonation pattern.

Research in [16] suggests intonation assessment on the syllable and sentence level. Their approach is based on classifying syllable pitch contours into 5 pitch types: low-high (LH), HL, LHL, HLH and „no equivalent“. The pitch intonation pattern on the sentence level is assessed using the mean pitch value for each syllable. Before F0 contour is used for classification, it is smoothed using a median filter. They conclude that pitch movement on the sentence level plays a more significant role in perceived intonational quality than on the syllable level.

The second existing approach to intonation assessment is based on aligning the reference and student utterances using the DTW algorithm and then comparing the aligned F0 contours frame-by-frame [2]. F0 contours are normalized and smoothed using a median filter and the comparison is done using correlation, with respect to 4 intonation patterns: high rise (HR), high fall (HF), low rise (LR) and low fall (LF). The aim of the approach is to decide whether two compared utterances were produced with the same falling-rising intonation pattern on the sentence level.

From the above-mentioned approaches, it seems reasonable to perform intonation assessment based on comparing the student's speech to a reference utterance produced with an expected intonation pattern. That way, it should be possible to detect intonation errors based on intonation pattern types used in linguistics.

3.4 Dynamic Time Warping (DTW)

The last section discussed different methods that are used for pronunciation assessment and error detection. Most of the methods, however, are ASR-dependent, which means that if they are practically implemented in a real application, they always need the whole ASR system to run in the background, which can be disadvantageous for mobile or web applications that are desired to be fast and lightweight. Unlike similar work in [12], which focuses on ASR-based error detection, the aim of this work is to develop an algorithm for foreign language pronunciation training purposes that would both fulfil the requirements for speed and lightweight design and still remain accurate enough. The core algorithm used in this work is the the Dynamic Time Warping (DTW) method and this work will explore how it can be utilized for the purposes of pronunciation error detection in English.

The DTW method was created in the 1960's and its main function is to adjust the length of the speaker's utterance and its phonemes to the length of the template it is being compared with. [21]

This algorithm basically aligns a sound signal with another sound signal. It aligns the lengths of two utterances and minimizes the influence of different lengths of phonemes and words and different speech rates on speech recognition. Generally speaking, it is a distortion algorithm that compresses and stretches parts of the utterance to force a match. However, its disadvantage is that if words of a speaker's utterance are compressed or stretched enough, they can fit the reference template, even if the actual words are different. [21]

3.4.1 Algorithm Description

The algorithm, as described in [4] and [15], is based on dynamic programming and its main objective in speech recognition is to successfully match words or phonemes in two time series despite wide variations in timing.

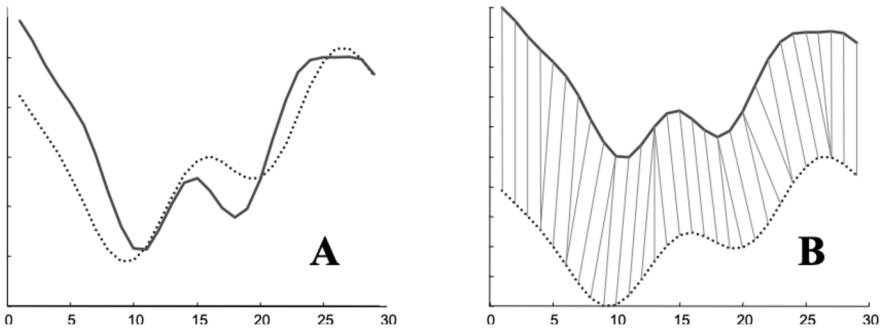


Figure 3.1: Example of two aligned time series. Source: [15].

Suppose we have two time series X, the reference, and Y, the speaker's utterance.

$$X = (x_1, x_2, ..., x_n) \tag{3.11}$$

$$Y = (y_1, y_2, ..., y_m) \tag{3.12}$$

Where n and m are the series lengths.

To align the two series, an $n \times m$ **distance matrix** δ is constructed. Each point (i, j) corresponds to the distance between elements x_i and y_j. There are many possible distance measures but the most frequently used is the Euclidean distance, so the matrix is defined as follows:

$$\delta(i, j) = (x_i - y_j)^2 \tag{3.13}$$

In addition to the distance matrix, a **cumulative distance matrix** γ is created as well. The elements of the cumulative distance matrix are defined as a sum of the current distance and the minimum of cumulative distances of the surrounding elements, as described in equation 3.14 and illustrated in Figure 3.2.

$$\gamma(i, j) = \delta(i, j) + min(\gamma(i-1, j), \gamma(i-1, j-1), \gamma(i, j-1)) \tag{3.14}$$

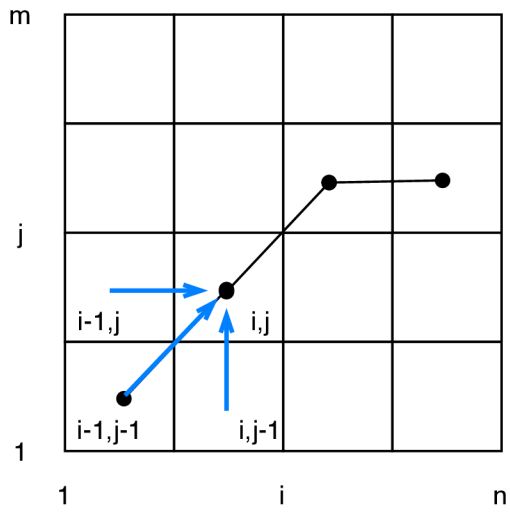


Figure 3.2: Cumulative distance matrix step pattern. (Illustration of Equation 3.14.)

The task of DTW is to find a **warping path** W, which aligns X and Y so that the distance between them is minimized.

$$W = (w_1, w_2, ..., w_k) \tag{3.15}$$

The time warping problem is formally defined as:

$$DTW(X, Y) = min_w(\sum_{k=1}^{p} \delta(w_k)) \tag{3.16}$$

Each element w_a of the path corresponds to one element of a cumulative matrix and the optimal path can be found by backtracking the complete cumulative matrix and choosing points with the lowest cumulative distances. The score of fit of a path is the path's length. When there is no timing difference between the two time series, the warping path is equal to the diagonal line $i = j$. An example of a warping path can be seen in Figure 3.3.

There are some constraints placed on the path in order to limit the search space: [4]

1. **Boundary conditions**. The starting and ending points of the path have to be $w_1 = (1, 1)$ and $w_k = (m, n)$, or an offset can be used.

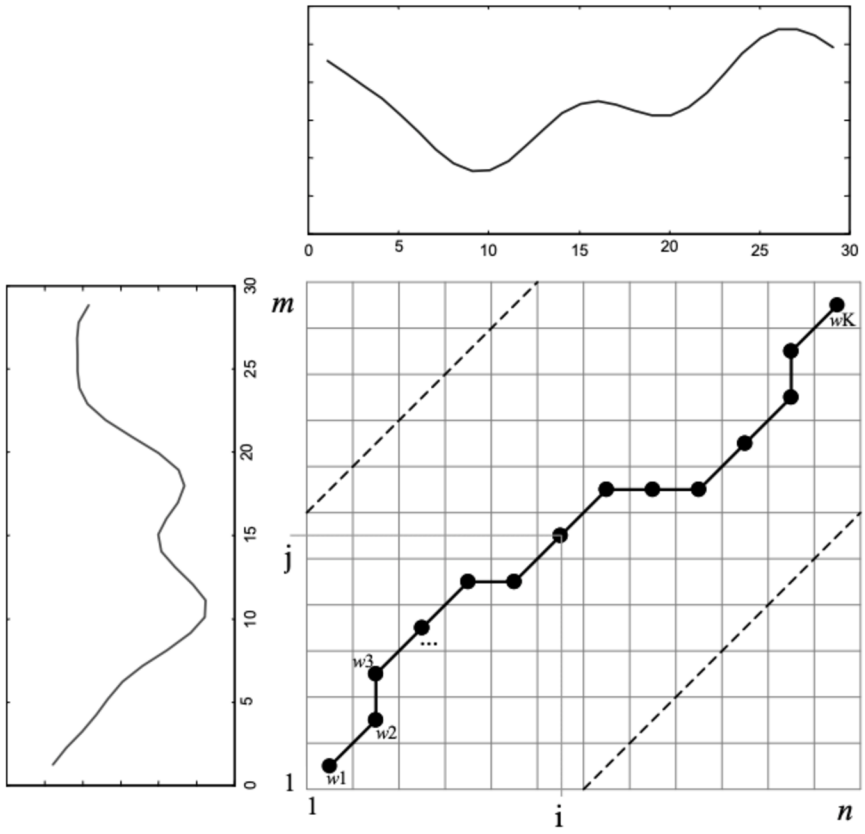


Figure 3.3: Example of a warping path. Source: [15].

2. **Monotonicity**. The path points have to be time-ordered. Given $w_k = (i, j)$ and $w_{k-1} = (i', j')$, then $i' \leq i$ and $j' \leq j$.

3. **Continuity**. The steps in the grid are restricted to neighbouring elements, including diagonally adjacent elements. Given $w_k = (i, j)$ and $w_{k-1} = (i', j')$, then $i - i' \leq 1$ and $j - j' \leq 1$.

4. **Warping window**. Allowable points of the path must fall within a warping window of width w: $|i - j| \leq w$

5. **Slope constraint**. Allowable paths can be constrained by limiting the allowable slope of the path.

The output of the DTW algorithm is the alignment of the student phonemes/words to the reference phonemes/words: $I(k) = i_R(k), i_S(k), 1 \leq k \leq K$, where $i_R(k)$ and $i_S(k)$ are frame indexes of the aligned utterances [2]. It is basically a forced match of two utterances.

3.4.2 Advantages and Disadvantages

The aim of this work is to improve an existing pronunciation training application provided by supervisor. The application uses DTW as the pronunciation assessment algorithm. As DTW works on the principle of comparing the student's speech to the one and only one reference recording, it will obviously never be able to catch the whole variety of correct

pronunciations that exist. However, the final product could potentially be used by teachers to teach a particular way of pronunciation of their choice, for example, British English, or even to teach pronunciation, including the rhythm and stress, of a poem, for instance.

The advantages of DTW are that the algorithm is trivial, ASR-independent, extremely lightweight and fast. Therefore, it will be easy to run solely on the client (such as a mobile device or a web browswer), without the need of a client-server architecture. Also, this method only needs little data to work and it is not dependent on a large database of perfect speech. What is more, to change the desired pronunciation to teach, it is enough just to replace the reference recording with a different one.

The disadvantages are that the accuracy will most likely be lower than the accuracy of ASR-based methods described earlier in this chapter and that, due to the distortion nature of the method, if the student says a sentence that is completely off from what was required from them, the algorithm might still be able to distort their speech to the extent that it will fit the reference recording, even though it would obviously be perceived as incorrect by a human. Our assumption is, however, that the speaker will strive for a correct pronunciation and will not deliberately feed the algorithm with unconstrained speech. Finally, as already mentioned, it might happen that the speaker pronounces one of the words using a different correct pronunciation than is actually present in the reference recording, and yet his correct pronunciation will be rejected because it is different than that particular realization of the word in the reference recording.

Even though the DTW method has a number of disadvantages, the advantages hold a lot of potential. This thesis will explore the potential of DTW to be the base algorithm for a more sophisticated pronunciation assessment and error detection algorithm. Most importantly, the DTW algorithm by itself obviously does not provide any feedback on the specific pronunciation errors. Therefore, the main task of this work will be to adjust the algorithm so that it also able to provide corrective feedback to the user.

The next chapter describes the application provided by supervisor and how it was redesigned.

Chapter 4

Redesign of the Original Application

As already mentioned at the end of the previous chapter, this work deals with the improvement of an existing pronunciation training application. The application was provided by supervisor and contains an implementation of the DTW algorithm. This chapter will talk about the original application in detail and describe how the application was redesigned both conceptually and regarding the user interface.

4.1 Original Application

The original application is a JavaScript web application implemented as a jQuery module that can be inserted into a website. The aim of the original application is to teach English pronunciation from educational videos. Its main component is a video player that stops after certain sentences in the video and asks the user to repeat them using a simple user interface with a microphone button. After the sentence is recorded by the user, the user is shown a feedback in the form of a similarity score in percentages. The second part of the feedback is that words with a high score are coloured in green, whereas words with low score are coloured in red. The user can also replay his own voice and compare it to the reference recording. The user interface can be seen in Figure 4.2.

Figure 4.1 displays the overall design of the original application. It contains the simple DTW implementation, botlleneck feature extraction and voice recording functionality.

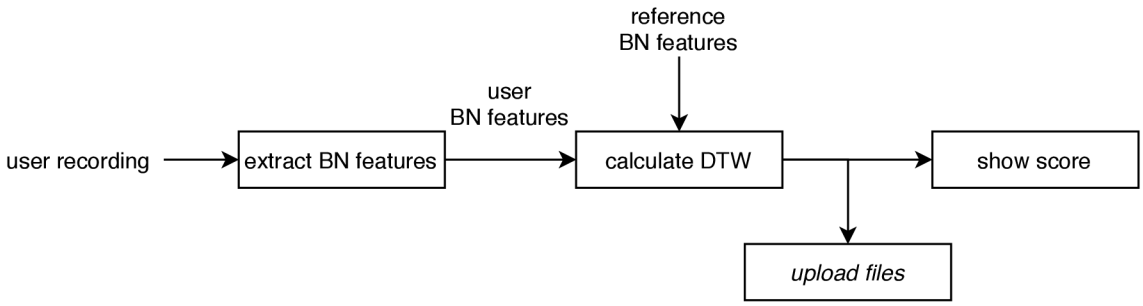


Figure 4.1: Original application design.

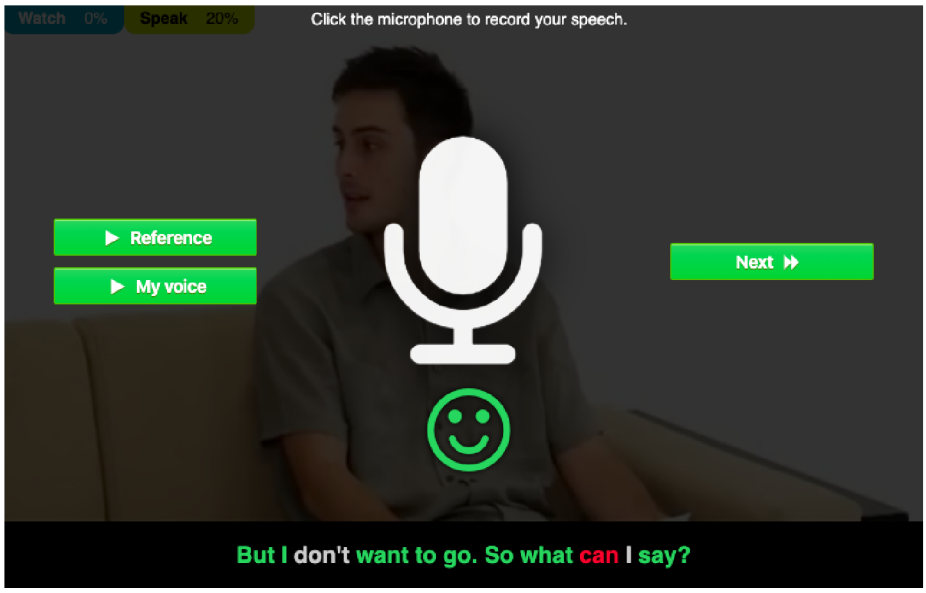


Figure 4.2: The old application UI.

4.1.1 Reference Data

The application requires a set of reference recordings and their segmentation to words and phonemes. This segmentation was prepared in advance using a pronunciation dictionary and an acoustic model of an ASR system. Thanks to the reference segmentation, the DTW algorithm will be able to segment the user speech to the corresponding words and phonemes.

4.1.2 Bottleneck Feature Extraction

The original system contains a neural network (NN) that is used to extract the bottleneck (BN) features from the recordings.

Bottleneck features are generated by a multi-layer perceptron (MLP) that is trained to predict phonemes [13]. The bottleneck neural network has multiple hidden layers, typically three. One of the hidden layers called the *bottleneck layer* has a significantly lower number of hidden units, compared to the other layers. The network creates a bottleneck of information in the bottleneck layer that provides features of low dimensionality [32]. Bottleneck features are direct outputs of the BN layer. Inputs for the BN network are melbank features extracted from speech.

BN features are a good representation of phonemic and prosodic information in the audio and have lower information redundancy than other features.

The neural network used in this work has three hidden layers. The third one is the bottleneck layer. The net is trained on native English speech to predict phonemes. The extracted BN features are inputs for the DTW algorithm.

In real time, bottleneck features are extracted only from user recordings. BN features of the reference recordings are prepared in advance to speed up the algorithm.

4.1.3 DTW

The original application contains an implementation of the DTW algorithm. Inputs for the algorithm are bottleneck feature vectors and its output is a similarity score described in the next section.

As already described in the previous chapter, the DTW algorithm takes the features extracted from the reference and user recordings and finds an alignment between them so that the acoustic similarity of the two recordings is as high as possible. The original application only takes into account the words, not the phonemes. Therefore, acoustic similarity is computed on the word level and the algorithm does not provide any information about what kind of pronunciation error has been made.

4.1.4 Similarity Score

The original application uses a simple similarity score of the reference and user's utterances. This score is displayed to the user in percentages as a very simple feedback. The similarity score is computed both globally (for the whole utterance) and for each word k. The acoustic similarity score is computed as follows, using the resulting DTW warping path:

$$score_k = 1 - (cumulativeDistance_k)/warpingPathLength_k, \tag{4.1}$$

where the word's cumulative distance is computed as the difference of the cumulative distances at the end of the word and at the beginning of the word.

The final score determines the acoustic similarity of the given pair of words: the reference and the user's.

All the following work described in this thesis is based on this application and utilizes, modifies and improves the existing functionality.

4.2 New Application

This work deals with three areas of improvement of the original application: user interface, algorithm and corrective feedback. This section will describe the first area: the complete redesign of the application's frontend.

4.2.1 User Interface

Not only was the user interface remade, but conceptually, the application was redesigned from a video player to a pronunciation training session that includes a number of pronunciation exercises. In each exercise, the user is asked to pronounce a single sentence. There are three possible exercise modes (the exercise mode can be set up in the module parameters):

1. **Read** mode. User is shown a sentence and is prompted to read it.
2. **Repeat** mode. User listens to a reference recording (plain audio) and is asked to repeat it.
3. **Repeat with subtitles** mode. User can see the sentence in text and can hear the reference audio at the same time. Then the user is asked to repeat the sentence.

At the end of each exercise, the user is shown the acoustic similarity score in percentages. Plus, if the exercise is in the *read* or *repeat with subtitles* mode, each word of the sentence

is coloured according to its similarity score. If the score is too low, the word is coloured in red, if the score is high enough, the word is coloured in green.

The user can also skip or repeat an exercise, listen to their own recorded voice and replay the reference recording. By hovering over a reference word, they can hear the word's reference pronunciation and by clicking on the word, the can hear their own pronunciation. This way, the user can compare their pronunciation with the reference one. Figure 4.3 shows the resulting user interface.

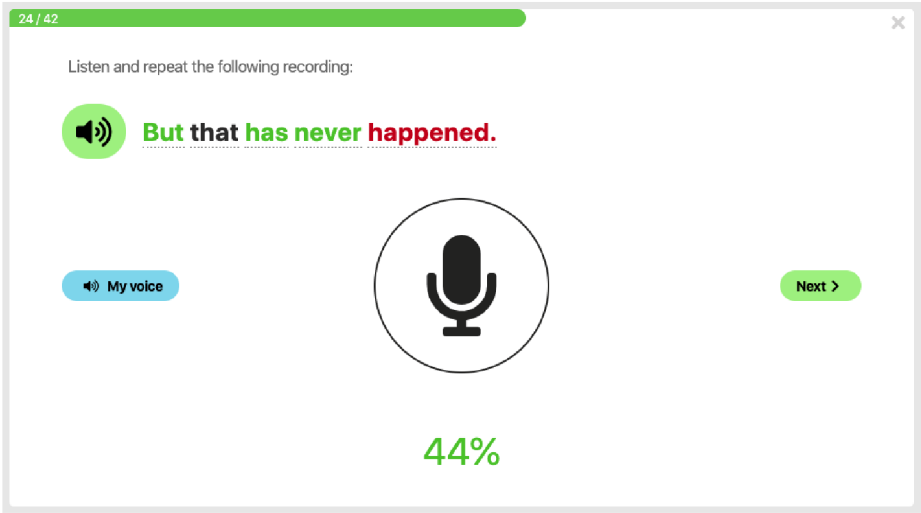


Figure 4.3: The new application UI (*repeat with subtitles* mode).

At the end of the pronunciation session, the user is given an overall score computed as the average of all exercise scores.

4.2.2 Customizability

The jQuery module is customizable through module parameters. The application can be run both on one and on a list of recordings. There are three possible exercise modes to choose from. Segments can be picked from a file randomly or sequentially. All parameters are summarized in Appendix C.

4.2.3 Reference Data

The new application requires the same data as the original application: reference recordings, reference segmentation and bottleneck features extracted from the reference audio in advance. The big advantage of this application is that if the reference segmentation to words and phonemes is provided, almost any kind of audio data can be used within the application.

In this phase, both the pronunciation assessment algorithm and the style of corrective feedback to the user stayed the same as in the original application. Before any other improvements of the application were made, the application was used to collect data from a variety of native and nonnative English speakers.

The following chapter (Chapter 5) describes the particular dataset of reference recordings that was used for the data collection, how the data was collected, and it summarizes some findings from the data.

Chapter 5

Data Collection

Along with the original application, a dataset of reference British English recordings was provided by supervisor, including reference segmentation to words and phonemes, and bottleneck features extracted in advance. This data was ready to use with the redesigned application and therefore, it was used as reference data for collecting a dataset of user recordings from native and non-native English speakers.

The aim of the data collection was to obtain a set of user utterances of the reference sentences, that would contain real pronunciation errors. Such data could then be annotated on phoneme-level errors and errors of stress and of intonation. The annotated data could be used for additional improvements of the pronunciation assessment algorithm.

Even though the data was successfully collected, it turned out to be unsuitable for the purposes of algorithm improvements. Unfortunately, many of the recordings were low-quality, incomprehensible or with a lot of background noise. Pronunciation errors in the recordings were often not clearly distinguishable and therefore not suitable as testing data for building a pronunciation error detection algorithm. Due to the low quality of many of the recordings and due to the lack of time needed to annotate such data, data annotation was eventually not done.

Instead, a small set of user recordings from one Czech female student was created, annotated by hand and used for building and testing the algorithm. Reason for not using some of the existing datasets, such as ISLE (Interactive Spoken Language Education) [20], is that it would be necessary to format them into the format required by the application, generate the segmentation and extract the features. This was already prepared by supervisor in the dataset of British English recordings. Also, the existing datasets only contain annotations of phoneme-level errors, whereas this work also focuses on prosodic errors, such as errors of stress and intonation.

This chapter describes the reference dataset provided by supervisor and how it was used to collect the data using the redesigned application. Next, it describes the collected dataset and what knowledge can be inferred from it.

5.1 Reference Dataset Description

The dataset of reference recordings provided by supervisor consists of over 400 recordings randomly chosen from YouTube educational videos. The dataset contains a wide range of learning material with varying quality, content and speaker style. For the purposes of creating a suitable set of reference sentences for data collection, a subset of reference

recordings was picked from this dataset. As it was desired for the reference speech to be as clear as possible, the following criteria were determined for selecting suitable data:

- British English
- no background music or noise
- adult voice
- clear speech
- meaningful, short and clearly separated sentences
- as few names and complicated words as possible
- as few filler words and interjections as possible

6 recordings were chosen for the application. 4 of them fulfils all of the selected criteria, one contains loud background music and one contains a child's voice.

5.2 Data Collection

A narrow selection of 7 particular sentences from the 6 recordings was chosen to be displayed to every single participant of the data collection. Each of these 7 sentences was presented in all three modes to the user: as a plain text sentence, as an audio, and as an audio with text. In addition to this small selection of sentences, another sample of 21 sentences randomly chosen from the 6 recordings were presented to each user (in a random mode, too). Thus, there were 28 distinct sentences shown to one user and a total of 42 pronunciation exercises.

In each exercise, the user either read the sentence or listened to the recording and then read or repeated it. Then the application displayed the simple acoustic similarity score of the user's pronunciation based on the DTW algorithm. Each recorded utterance was automatically uploaded to a server, along with the score and user data. User data collected by the application was: age, gender, whether or not the user is a native English speaker, native language and level of English according to the Common European Framework of Reference for Languages (CEFR).

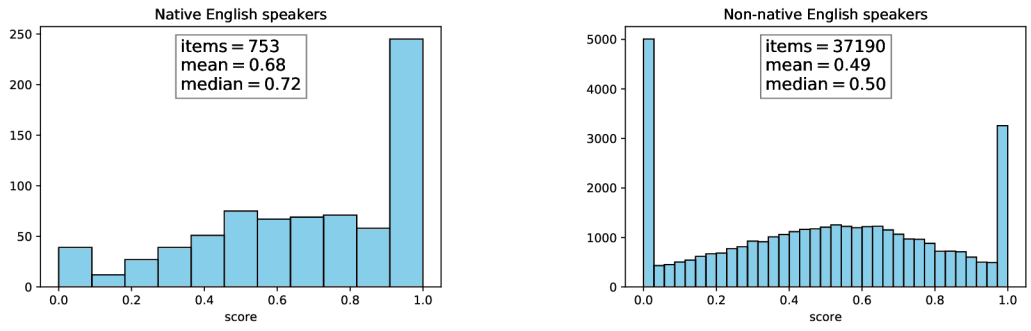
Data from approximately 875 people from different countries all around the world has been collected, which makes it over 37000 recordings. The number of participants is only approximate because user identifiers were not collected. One recording session contained 42 exercises with the possibility to either skip an exercise or to repeat an exercise. That is why the numbers of recordings per person differ. The majority of the participants were from India, and there were native English speakers from the United Kingdom and America, as well as nonnative speakers from Czech Republic, Germany, Hungary, Malaysia, Japan, Greece, France or African countries. Details are summarized in table 5.1.

5.3 Data Analysis

The collected data was analysed for score differences by certain data attributes to find out whether anything interesting can be inferred from the data. As expected, scores of native English speakers seem to be significantly higher than scores of non-native English speakers, as can be seen in Figure 5.1.

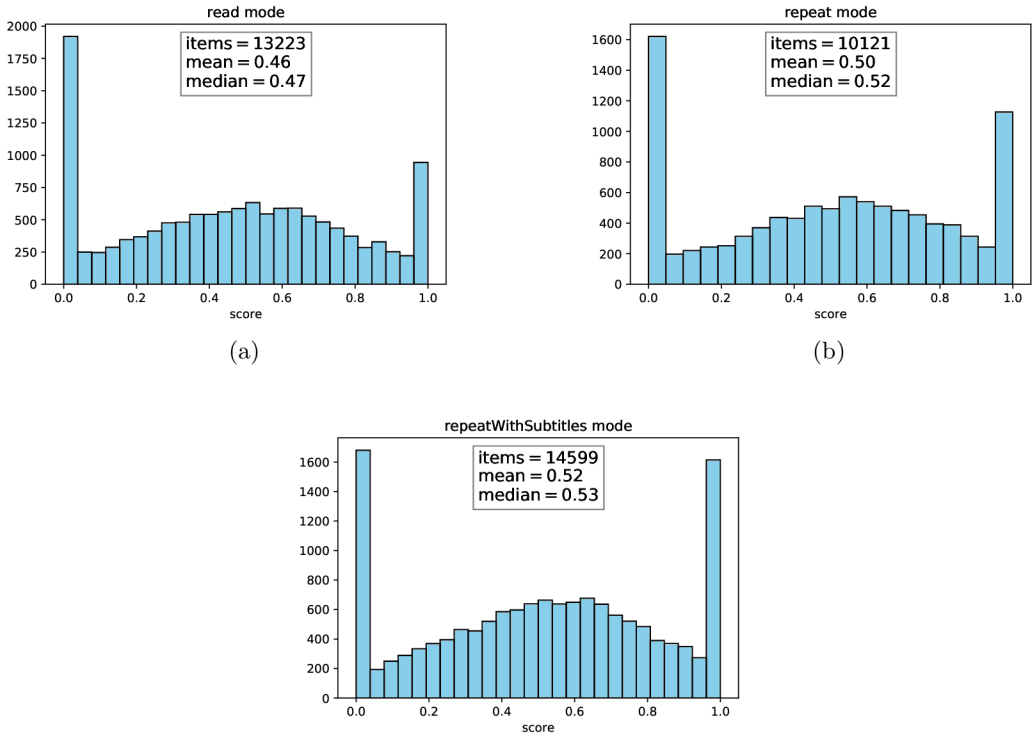
Table 5.1: Number of recordings obtained of people with different native languages.

Native language	**# of recordings**	**Approx. # of people**
Urdu	26987	630
Pashto	5577	125
English	906	20
Arabic	888	20
Punjabi	545	12
Czech	531	11
Persian	430	10
Hindi	128	3
Greek	98	2
Nepali	89	2
Slovak	85	2
German	75	2
Irish	61	1
Sindhi	57	1
Telugu	55	1
Hungarian	49	1
Chinese	44	1
Japanese	40	1
Basque	3	1
Other	1093	25
Unknown	202	4
Total	**37943**	**875**



(a) (b)

Figure 5.1: Histogram of scores from native (a) and non-native (b) English speakers.



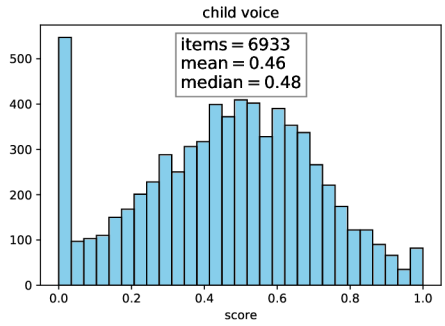
(a) (b) (c)

Figure 5.2: Histogram of scores from read mode (a), repeat mode (b) and repeat with subtitles mode (c).

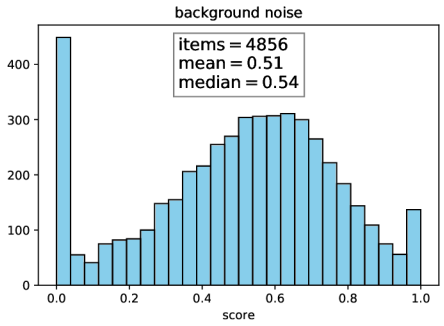
When scores were compared across the three different exercise modes, the resulting histograms in Figure 5.2 suggest that when participants are presented an exercise in the *read* mode, their scores tend to be slightly worse than in the *repeat with subtitles* mode, in which the scores were the best out of all three modes. It makes sense that if a user hears the correct pronunciation first, they are more likely to actually repeat the correct pronunciation. Reading exercises seem to be generally slightly more difficult for the users.

Histograms of scores from exercises where the reference voice was a child's voice or where there was a lot of background noise were compared with the rest of the recordings (see Figure 5.3). The comparison does not tell much about whether child voice or background noise have a direct influence on the score. Even though it would make sense if they had, more data and/or analysis would be needed to make any conclusions.

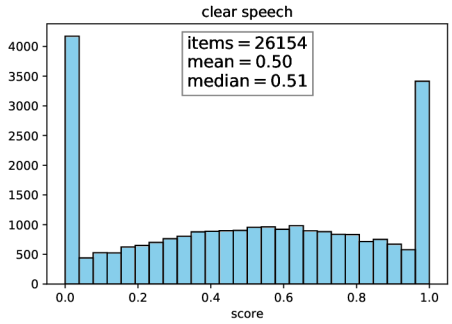
Not much difference in scores was observed between different CEFR levels of English, too. When comparing particular languages, some of the indian languages, such as Urdu or Punjabi seemed to have a significantly higher number of zero scores than, for example Czech. Figure 5.4 shows the difference between Punjabi and Czech. More context would be needed in order to make any general conclusions from the data, however.



(a)

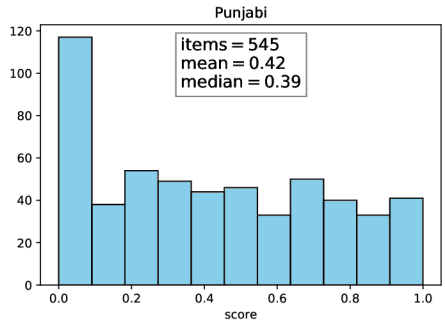


(b)

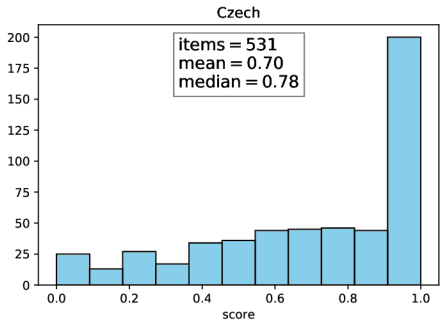


(c)

Figure 5.3: Histogram of scores from recordings with a child voice (a), a lot of background noise (b) and clear speech (c).



(a)



(b)

Figure 5.4: Histogram of scores from Punjabi speakers (a) and Czech speakers (b).

Chapter 6

Improvements of the Pronunciation Assessment Algorithm

After the user interface was redesigned and data was collected, improved pronunciation assessment algorithms based on the original DTW implementation were designed. The new algorithms were designed with the goal for the methods to be both accurate enough in pronunciation error detection, and able to output the specific pronunciation errors so that the user can be given a corrective feedback. A set of experiments were performed to test the designed methods. This chapter describes the new design of the pronunciation assessment system and then, it discusses the experiments and their results.

6.1 Design

The original pronunciation assessment system, that was described in Chapter 4, contains only bottleneck feature extraction and the DTW algorithm. The original output of the system was a similarity score of the two utterances in percentages. The original system was extended and new components were added.

Figure 6.1 shows the design of the new pronunciation error detection system. Stress error detection and intonation error detection blocks, along with feature extraction, were added to the system. Moreover, the DTW algorithm was slightly modified to give specific phoneme-level information about pronunciation errors.

6.1.1 DTW Modification

In this thesis, the original implementation of the DTW algorithm is adjusted so that it is able to provide more meaningful results and corrective feedback to the user, rather than simply a single number (similarity score) for each word and sentence.

First, the algorithm was modified to work not only on the word level but also on the phoneme level. The new algorithm takes into account acoustic similarities of specific phonemes. Second, phonemic error detection method was designed. The method was designed to detect phoneme insertions and phoneme deletions solely from the DTW warping path.

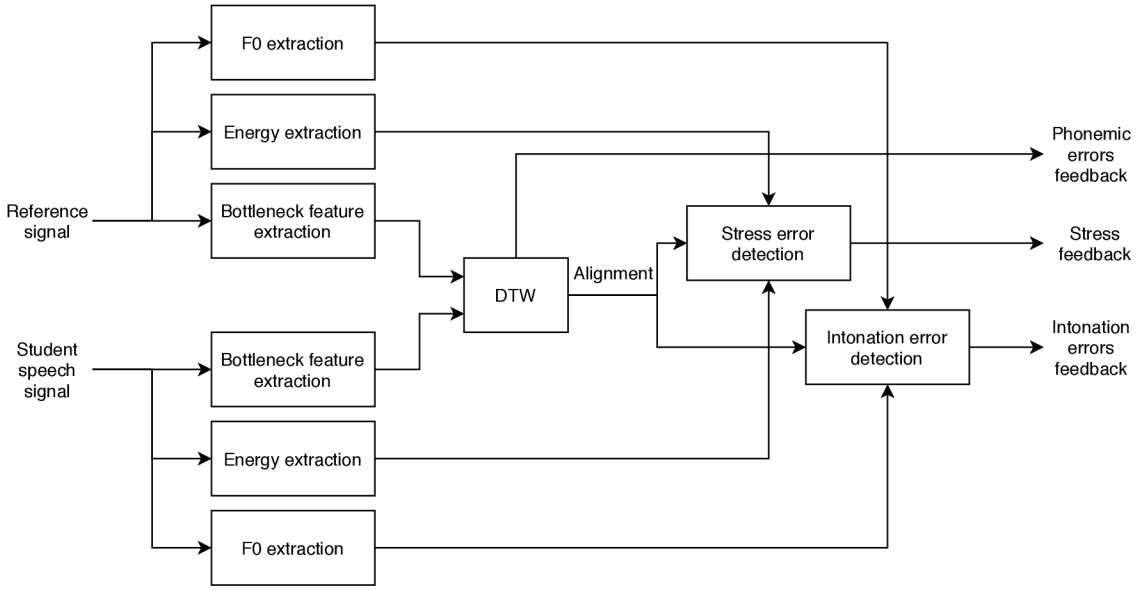


Figure 6.1: Design of the new pronunciation error detection system.

6.1.2 Energy Extraction and Stress Error Detection

The goal of stress detection in this work is to detect any primary stress errors and provide specific feedback to the user about where the error has occurred exactly and advice on how to correct it. It is designed to work on the syllable level.

Energy features used in the stress error detection method are extracted from the signal in the following manner: log energy is computed for each frame of the input signal. The resulting energy contours are compared for each pair of syllables in the reference and user's word. Primary stress is detected for each word and based on the comparison, stress errors are marked by the algorithm.

6.1.3 F0 Extraction and Intonation Error Detection

Intonation error detection is designed in a similar manner as stress error detection. First, the fundamental frequency (F0) in Hertz is estimated, and then intonation fluency is assessed on the word level. Because F0 extraction is performed outside of the designed system, using a Kaldi pitch extractor, this component of the system will not be a part of the final application. The goal of the intonation assessment experiments is to find out whether intonation assessment on top of the DTW algorithm is possible or not.

6.2 Experiments

In this section, improvements of the original algorithm are described in detail. Because pronunciation error detection is usually done using ASR-based algorithms, it is also discussed whether or not the designed methods based on the DTW algorithm without an ASR system could be used for pronunciation error detection and providing meaningful corrective feedback to the user.

Three areas have been explored during this phase of the work. Firstly, detection of particular phonemic errors in terms of phoneme insertions and phoneme deletions. Secondly,

detection of stress errors, in terms of primary stress correctness on the syllable level. And finally, experimental detection of intonation errors on the word level.

The experiments were evaluated in terms of whether the detection methods are **accurate** enough and whether they have the potential to provide the user with a **specific** corrective feedback that will help them learn.

6.2.1 Evaluation of Experiments

The following experiments were performed on a small set of reference recordings taken from the dataset of recordings from YouTube videos described in Chapter 5. As already explained, the non-native dataset collected during this project and described in Chapter 5 could unfortunately not be used to evaluate the experiments. For that reason, the following experiments were performed on a small set of test utterances with purposefully created pronunciation errors. These recordings come from one female speaker of Czech nationality and were manually annotated on the word, syllable and phoneme level. There were 154 recordings in total, 31 contained purposefully created phoneme deletion errors, other 31 contained phoneme insertion errors, 40 contained primary stress errors and the last 52 contained intonation errors.

The designed algorithms where evaluated in terms of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). True positive rate (TPR) and false positive rate (FPR) were computed for each experiment. The following equations were used [1]:

$$TPR = \frac{TP}{TP + FN} \tag{6.1}$$

$$FPR = \frac{FP}{FP + TN} \tag{6.2}$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.3}$$

6.2.2 Detection of Phonemic Errors

The DTW algorithm itself outputs an alignment of the student and reference utterances. That is, for each phoneme and/or word in the reference utterance, it finds the corresponding segment in the student's speech. It will always try to match the two audios, and in order for them to match, it will perform two types of distortion transformations along the way: it will stretch some parts of the utterances or it will shrink them.

Ideally, it should map all frames of a correctly pronounced phoneme on the whole reference phoneme, and pronunciation errors should be mapped only on the phonemes where the mistake has occurred. The assumption is that the similarity score of the correctly pronounced phonemes will be maximized, whereas the score of the errors will be minimized. If this assumption is correct, it should be possible to detect phone insertions and deletions solely from the DTW warping path. When there is a phone insertion error, all frames of the preceding and following phonemes should ideally be mapped on the reference phonemes, and all frames of the extra phoneme should be mapped on zero frames in the reference recording. The process should word in the opposite manner for phone deletion errors;

[1] https://en.wikipedia.org/wiki/Sensitivity_and_specificity

namely, all frames of the reference phoneme should be mapped on zero frames in the student's recording, where the deletion error has occurred.

Figure 6.2 shows the two types of patterns that should ideally reflect the corresponding phoneme errors. The following subsections describe the algorithms used for their detection, and their evaluation.

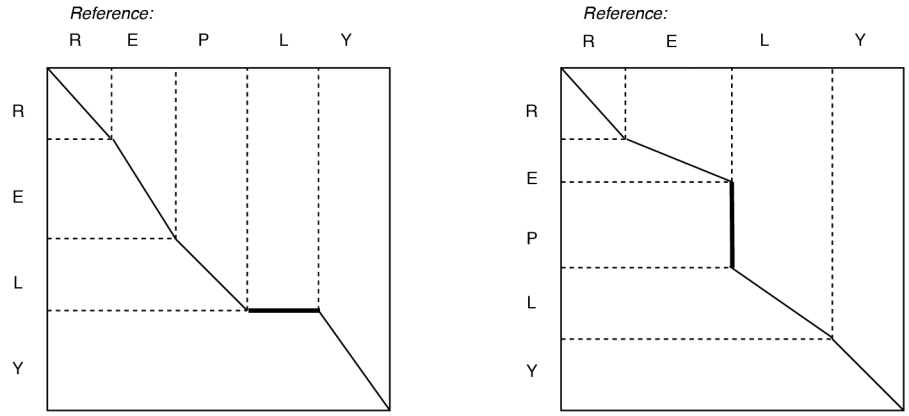


Figure 6.2: Illustration of the ideal warping path patterns for deletion errors (left) and insertion errors (right).

Phoneme substitution errors should be determined by the similarity score alone, given a specific threshold, so they will not be further discussed here.

Phoneme Deletions

The basic idea is that when a deletion error occurs, a large number of frames belonging to one reference phoneme are mapped on a very small number of frames of the student's utterance. In addition, logically, the deleted reference phoneme should be given a very low similarity score. Based on this characteristic and after the analysis of a number of warping paths for deletion errors, the resulting decision rules were established and used in Algorithm 6.1.

This algorithm was run on 31 short recordings containing deletion errors, which were recorded solely for this purpose. There were 1054 phonemes in total and out of these, 53 phonemes contained deletion errors. Table 6.1 summarizes the results. In can be seen that only a small part of the errors is actually detected and moreover, the number of false alarms is high. The true positive rate is not more than 22.6% and the false positive rate is quite high (3.9%).

Upon closer examination, the results revealed that the deletion of vowels is easier than the deletion of consonants because vowels are mostly much longer than consonants. Also, many errors were not detected at all because of the inaccurate segmentation to phonemes in the reference recording. Additionally, DTW often takes last frames of the phoneme before the deleted phoneme and first frames of the phoneme after it and aligns them to the deleted phoneme in the reference recording. So, not only is the detection more difficult, but similarity scores of the neighbouring phonemes are affected as well.

```
horizontalLines = extractHorizontalLines(warpingPath)
for line in horizontalLines:
    refPhoneme, exaPhoneme = getCorrespondingPhonemes(line)
    if len(line) >= 0.95 * len(refPhoneme)
        mark as mistake
    elif len(line) >= 0.5 * len(refPhoneme)
        if len(exaPhoneme) <= 1:
            mark as mistake
        elif len(exaPhoneme) <= 4:
            if exaPhoneme.score <= 0.4:
                mark as mistake
            else:
                mark as OK
        else:
            mark as OK
    else:
        mark as OK
```

Algorithm 6.1: Phoneme Deletions Detection Algorithm

Table 6.1: Results of the phoneme deletions detection algorithm.

Total errors	53
Total phonemes	1054
True positives (TP)	12
False positives (FP)	39
False negatives (FN)	41
True negatives (TN)	962
TP rate	**22.6%**
FP rate	**3.9%**

Phoneme Insertions

Detecting phoneme insertion errors is similar to detecting phoneme deletions. In phoneme insertion errors, frames of the inserted phoneme in a student's speech are mapped on a very small number of frames in the reference recording. The difference is that the length of the path belonging to the inserted phoneme or phonemes is unknown. Because of that, the average phoneme length of the student speech was computed, and it has been empirically determined that the path length has to be at least the average phoneme length. In addition, the vertical line has to start at the beginning of a phoneme or end at its end, with the tolerance of one frame. And finally, the ratio of frames in the student and the reference speech within the line must be at least 2:1. The final algorithm is described in Algorithm 6.2.

The algorithm was run on 31 recordings with insertion errors. These recordings were different from recordings used to evaluate the deletion error detection. There were 32 phoneme insertion errors in total. Table 6.2 shows that the algorithm was able to correctly detect only about one third of the errors (true positives). While the false positive rate

```
verticalLines = extractVerticalLines(warpingPath)
for line in verticalLines:
    refPhoneme, exaPhoneme = getCorrespondingPhonemes(line)
    if len(line) >= 0.98 * avgExaPhonemeLength:
        if lineStartsAtPhonemeStart(line, refPhoneme, exaPhoneme) or
           lineEndsAtPhonemeEnd(line, refPhoneme, exaPhoneme):
           if phonemeLengthRatio > 2:
            mark as mistake
    else:
        mark as OK
```

Algorithm 6.2: Phoneme Insertions Detection Algorithm

stayed as low as 0.7%, which is desirable, the true positive rate was only 31.2%. Even though this result is better than the result of deletion errors detection, it is still insufficient for the purposes of practical implementation.

Qualitative analysis of the results showed that this algorithm works better if a larger number of phonemes is inserted by the student. However, oftentimes the results are affected by inaccurate segmentation of the phonemes in the reference recording (this segmentation was provided by supervisor and created automatically using an ASR system). It was also affected by not accurate enough phoneme alignment done by the DTW algorithm, as mentioned in the evaluation of the deletions detection algorithm.

Table 6.2: Results of the phoneme insertions detection algorithm.

Total errors	32
Total phonemes	1054
True positives (TP)	10
False positives (FP)	7
False negatives (FN)	22
True negatives (TN)	1015
TP rate	**31.2%**
FP rate	**0.7%**

Conclusions

From the results of the phoneme error detection experiments, it can be concluded that neither of the two methods can be successfully used in phoneme error detection. While it might be possible to slightly increase the low accuracy of the methods by further improving the algorithms or by using more advanced algorithms such as machine learning, the main drawback of the methods is that they are heavily dependent on the results of the DTW algorithm. Since the main goal of DTW is to match two utterances in any possible way, it will distort the contents of the utterances. Based on the experiments, using such distorted data for the purposes of *accurate* phoneme error detection does not and cannot lead to successful results.

6.2.3 Detection of Stress Errors

Let's assume stress errors can be detected from a pair of reference and student utterances with the help of the final DTW alignments. Based on [2], stress is defined as a combination of loudness (energy), pitch and duration. Energy is the primary feature of stress and it is extremely easy to extract from audio. Adding pitch would be outside of the scope of this thesis, so energy was used as the only feature in this experiment. The assumption is that energy itself is enough to detect stress errors with a satisfactory accuracy.

Method

The detection method is based on comparing energy feature vectors extracted from the reference and student speech. First, energy is extracted from the audio in a form of log energy for each frame, and it is normalized into a fixed range between 0 and 1. After that, energy feature vectors are aligned according to the final DTW alignment. In English, stress is syllable-based [17], so data was first annotated in terms of segmentation to syllables. Each word has primary, secondary and non-stressed syllables. In this experiment, only primary stress is taken into account.

For each syllable, normalized energy vectors belonging to that syllable are compared between the reference and student utterances. For this step, the following three methods were tested:

(i) Correlation.

(ii) Euclidean distance of root mean square energies.

(iii) Simple detection of a syllable with the maximum energy peak within a word.

Correlation did not provide good results because it is dependent on segmentation accuracy. Once the start time of a speech unit was slightly shifted, correlation results were negatively affected. Root mean square energies gave less accurate results with decreasing size of the speech unit. For instance, on the phoneme level, it was significantly inaccurate in relation to phonemes at the beginning of a word because they usually contained both very low and very high energy values. Instead of emphasizing the high values, the approach averaged the values so primary stress was almost never correctly detected if located at the first phoneme.

Surprisingly enough, best results were obtained using the third and simplest method. For each word, the syllable corresponding to the highest energy peak within the word was found and marked as primary stress. Primary stress of reference and student speech was compared and, if different, marked as a stress error. This approach had a true positive rate as high as 82.3% when tested on 40 recordings with stress errors, created just for this purpose. False positive rate was 5.3%. Such results look promising.

In order for this algorithm to be used in real applications, the user's point of view was also considered. For the user, the less false positives, the better, because they will trust the application more. True negatives do not play such a great role here. It is worse to tell the user there was an error while there was none, than to be silent about some of their actual errors. It was found out that only 62% of the feedback did not contain any false positives. This rate was called *user friendliness*. 38% of the feedback the user would receive from such algorithm would contain false positives. This means that in 38% of cases the student

```
extractEnergy()
normalizeEnergy()
for word in utterance:
    for syllable in word:
        primaryStress, secondaryStress = detectStress()
        if primaryStress.ref - secondaryStress.ref < THRES or
           primaryStress.exa - secondaryStress.exa < THRES:
            mark as OK
            continue
        if primaryStress.ref.syllable != primaryStress.exa.syllable:
            mark as mistake
        else:
            mark as OK
```

Algorithm 6.3: Primary Stress Error Detection

would be told they made a mistake somewhere in the sentence although they did not. For that reason, it was further explored how the algorithm can be improved.

For a pronunciation error detection algorithm, it is better to decrease the number of false positives as much as possible, even if that means that the number of true positives would decrease as well (the algorithm would leave some pronunciation errors undetected). This will increase the credibility of the algorithm and provide for better user experience.

Table 6.3: Reasons for inaccurate stress detection in the no threshold method.

Reason	**Number of recordings**
Secondary stress has a higher energy	13
Inaccurate DTW alignment	6
Other	8

After qualitative analysis of the results, it was found that most false positive and false negative results were caused by the fact that sometimes secondary stress in a word has a slightly higher energy than the word's primary stress. The reason for this reality is that stress is not determined just by energy but also by pitch and duration. Since it was impossible to add pitch into the detection process, detection of secondary stress was added into the method. If the primary and secondary stress were too close to each other, the word was automatically marked as correct in terms of stress. A threshold was set up to determine how close the values can be. Summary of the final method is described in Algorithm 6.3. Table 6.4 displays the above-discussed results. Even though the overall accuracy (ACC) of the detection is quite high (over 80%), the TPR and FPR values are much more significant factors to consider before making any conclusions. In our case, the most important measures are the number of pronunciation errors actually detected by the algorithm and the number of false alarms determined by the algorithm.

Figure 6.3 displays the ROC curve of the stress detection algorithm, depending on the threshold, and Figure 6.3 show how user friendliness changes with different thresholds. 0.02, 0.06 and 0.08 seem to be meaningful values of the threshold. 0.02 has a high true positive rate while 70% of feedbacks are relevant to the user. This setting catches most of the pronunciation errors but gives many false alarms, too. The „user friendliness“ in case

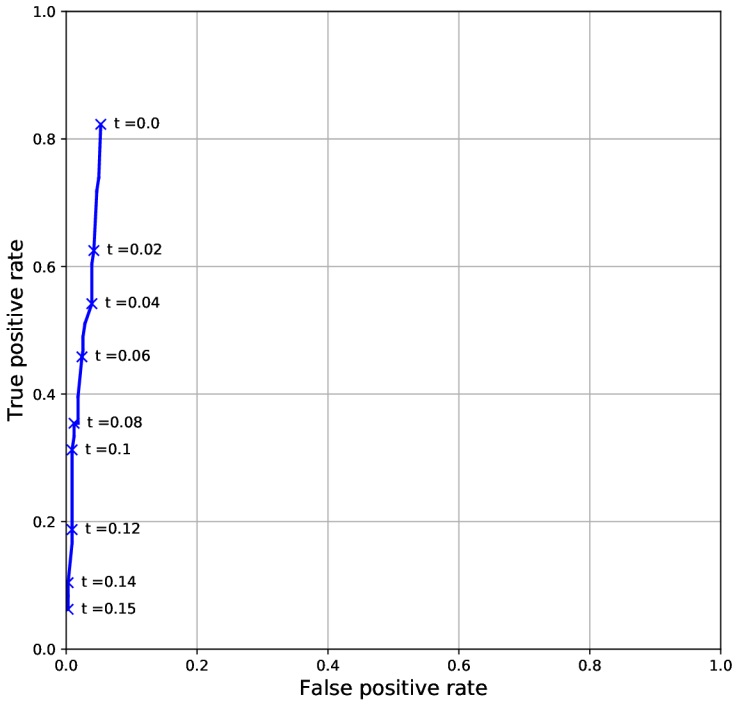


Figure 6.3: ROC curve for the stress detection algorithm, according to the threshold parameter.

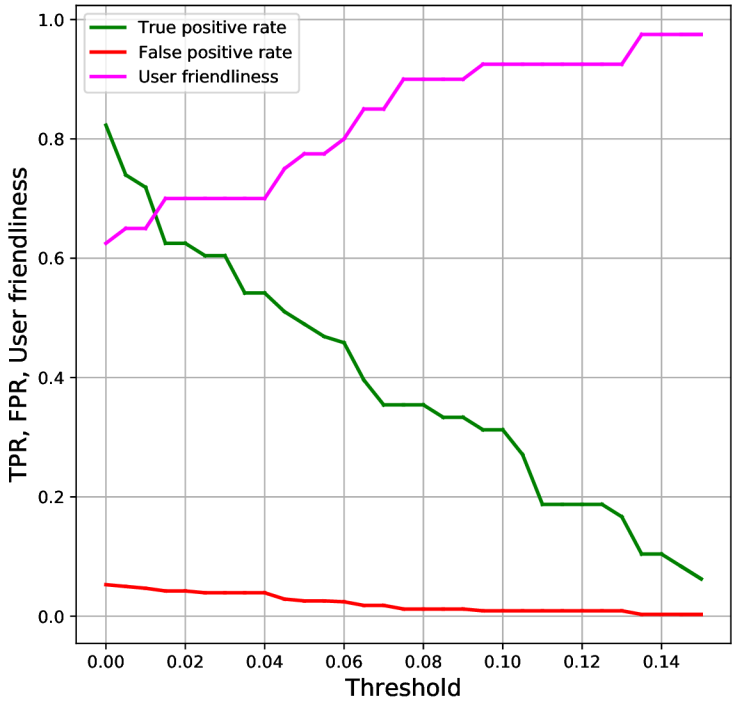


Figure 6.4: Dependency of user friendliness on the threshold for stress error detection.

of value 0.06 is even higher, while the true positive rate is still somewhat good. The value of 0.08 has the lowest false positive rate, which means that algorithm with this setting will be likely to provide a very small number of false positives, i.e. it will be most likely not to tell the user there has been a mistake while there was none.

Table 6.4: Results of the stress detection method with and without threshold.

	No threshold	**T = 0.02**	**T = 0.06**	**T = 0.08**
Total errors	96	96	96	96
Total syllables	475	475	475	475
True positives (TP)	79	60	44	34
False positives (FP)	35	28	16	8
False negatives (FN)	17	36	52	62
True negatives (TN)	344	351	363	371
TP rate (TPR)	**82.3%**	62.5%	45.8%	35.4%
FP rate (FPR)	5.3%	4.2%	2.4%	**1.2%**
Accuracy (ACC)	89.1%	86.5%	85.7%	85.3%
% of feedbacks without FP	62%	70%	80%	**90%**
% of feedbacks with FP	38%	30%	20%	**10%**

Conclusions

Based on the experiments discussed above, detection of primary stress errors based on energy features and DTW alignment is possible and gives good results. The best results were achieved using the simple energy peak detection method in combination with a threshold that allows modifications of the user-friendliness of the algorithm. The algorithm could be further improved by adding pitch (F0 contour) into the feature array.

6.2.4 Detection of Intonation Errors

In addition to previous experiments, an extra one was performed. The goal of this experiment was to find out whether intonation errors can be detected using simple algorithms and with the help of DTW alignments only.

Intonation is usually assessed using the pitch (F0) contour of the utterance. F0 contours were extracted from both reference and test utterances using a Kaldi pitch extractor that is not a part of the designed system. Intonation was assessed with regard to five different intonation patterns, based on intonation patterns used in linguistics [29]. The following intonation patterns were taken into account: rise (R), fall (F), rise-fall (RF), fall-rise (FR) and constant (C).

Instead of assessing intonation globally on the sentence level, each word was considered separately. It would be possible to assess intonation on the syllable level, too, but words are more meaningful in terms of providing feedback to the user. Moreover, more resources would be needed to annotate test data on the syllable level. Even though intonation assessment on the syllable level would be easier in terms of classification into intonation patterns (one syllable equals one intonation pattern), word-level assessment made it possible to create and annotate testing data in a reasonable time and without the need of an expert annotator. The disadvantage of assessing intonation on the word level, on the other hand, is that a

single word may contain a combination of intonation patterns, such as fall-rise-fall-rise so it is more difficult for classification.

The approach taken for this experiment is similar to approaches in [16] and [2]. First, pitch contours are smoothed using Hamming smoothing window as defined in [27] and then, F0 contour of each word is classified into one of the intonation pattern classes using several classification rules, as summarized in Algorithm 6.4. Finally, intonation patterns of the reference and test utterances are compared and intonation errors are determined. An intonation error occurs if intonation patterns that are being compared fall within the following set of pairs: $(R, F), (R, C), (F, C), (RF, C), (RF, C), (RF, F), (FR, F), (FR, RF)$.

The overall intonation pattern of a word is determined by extracting local minima and maxima of the pitch curve, ignoring those that are too close to each other, and using the rest for classification. If only two points are left, the slope of the line between them is taken into account, as well as the pitch change throughout the word. The parameters will distinguish between a C, R and F. Given that three points are left, the same parameters are used to determine a FR or RF. Unfortunately, using such a simple algorithm makes it impossible to take into account more complicated intonation patterns, but it still serves well as a proof of concept that such algorithm can work in practice if modified accordingly.

```
smoothF0Contours()
for word in utterance:
    points = filterRelevant(extractLocalMinimaMaxima())
    if len(points) == 2:
        if slope(points) < SLOPE_THRES and F0Diff(points) < DELTA_F0:
            return IP_CONSTANT
        elif slope(points) > 0: return IP_RISE
        elif slope(points) < 0: return IP_FALL
    elif len(points) == 3:
        if F0Diff1(points) > DELTA_F0 or F0Diff2(points) < -DELTA_F0:
            return IP_RISEFALL
        elif F0Diff1(points) < -DELTA_F0 or F0Diff2(points) > DELTA_F0:
            return IP_FALLRISE
        else: return IP_CONSTANT
    else: return IP_UNKNOWN
```

Algorithm 6.4: Intonation Error Detection

The algorithm was evaluated on a set of 52 recordings with intonation errors that were annotated on the word level. Each word annotation contained information about whether or not the word contains an intonation error in comparison with the reference utterance.

Parameters $SLOPE\_THRES$ and $DELTA\_F0$ have been set to 1.5 and 60 respectively. Such setting proved to give the best results: FPR is 19% and TPR is 70.4% (see Table 6.5). It can be seen that the FPR is quite high. The reason for that may be that the algorithm is more accurate than a human rater in some cases. Also, the reason for detection errors definitely is the classification algorithm itself. Logically, a set of simple rules will probably not be enough for accurate classification. For that reason, it would be better to train a machine learning classifier solely on the intonation pattern classification

from pitch contours. Also, it might help increase the accuracy if data are annotated by an expert.

To sum up, detection of intonation errors based on direct comparison between a reference and test utterances using the DTW alignment works, but there is a lot of room for improvement regarding the intonation pattern classification algorithm itself.

Table 6.5: Results of the intonation error detection algorithm.

Total errors	81
Total words	239
True positives (TP)	57
False positives (FP)	30
False negatives (FN)	24
True negatives (TN)	128
TP rate	**70.4%**
FP rate	**19%**

6.2.5 Summary of Results

A number of experiments with the DTW algorithm was discussed in this chapter. It was found that due to the distortion character of DTW, accurately detecting phoneme-level errors, such as phoneme deletions and phoneme insertions solely from the warping path is not possible. The reason for that is that the primary goal of DTW is to lengthen or shorten two utterances *to any possible extent* so that they match. Results of DTW will most likely always contain inaccuracies, so it is not possible to accurately detect phoneme-level pronunciation errors from them.

Results of stress error detection experiments, on the other hand, look more promising. Given energy features, the algorithm reached 82.3% true positive rate. With an additional threshold, it is possible to further decrease the number of false alarms (correct pronunciations classified as pronunciation errors), which are most harmful to the user. Along with false positives, true positives decrease, too. The threshold enables us to find balance between high TP rate and low FP rate. Stress detection could be further improved by adding pitch into the feature array.

The same applies for intonation error detection based on direct comparison between the reference and test utterances. Given F0 contours, it is possible to classify contours into intonation patterns and draw conclusions about the correctness or incorrectness of intonation based on comparison between the two patterns. The algorithm could be significantly improved by changing the classification rules to a classifier trained using a machine learning algorithm. In this case, the algorithm was not so much affected by alignment errors from the DTW algorithm.

To sum up, the smaller the unit, the less accurate any detection performed on top of the DTW algorithm. Phoneme-level errors are very difficult to detect because of the distortion characteristic of DTW. Syllable-level and word-level pronunciation errors can be fairly well detected using algorithms that use DTW only to align the utterances. By improving the simple detection algorithms described in this chapter, the accuracy could rise even further.

6.3 Implementation

Not all of the above-mentioned methods were implemented into the application. The final pronunciation assessment algorithm consists of only two parts: the improved word-level acoustic similarity assessment that takes into account the acoustic similarities of particular phonemes, and energy error detection on the syllable level. Due to its low accuracy, phoneme insertion and deletion detection was not implemented. As already explained, assessing intonation would require the application to be able to extract F0 contours, which was outside of the scope of this thesis. The overall design of the final application can be seen in Figure 6.5.

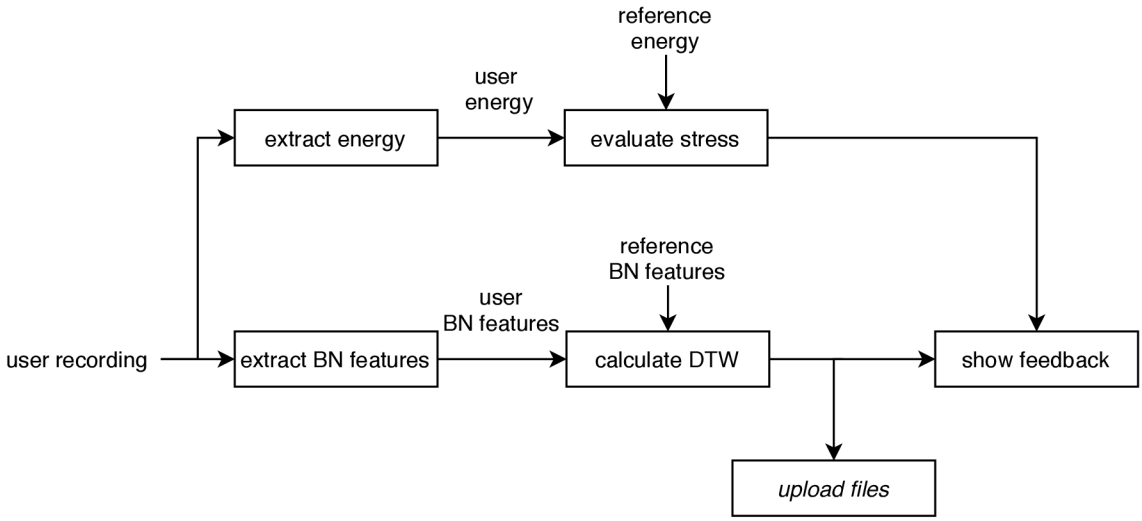


Figure 6.5: New application design.

In **acoustic similarity assessment**, similarity score is derived from the warping path of the DTW algorithm the same way as in the original application, as described in section 4.1.4. However, the similarity score is computed on the phoneme level. Then, each word is analysed and marked as either correctly pronounced or mispronounced. A number of parameters have been set that help determine whether the word is going to be marked as mispronounced:

1. There is at least one phoneme with a similarity score lower than 10%. OR
2. There are at least 2 phonemes with a similarity score lower than 20%. OR
3. The overall similarity score of a word is less than 30%.

Stress error detection is more straightforward. The stress error detection algorithm marks all syllables where primary stress differs from the reference recording. That means that both the missing stress and the extra stress are marked by the algorithm. Therefore, each stress error consists of two parts: the incorrectly stressed syllable and the syllable that should have been stressed instead. This setting allows the application to colour the former in red and underline the latter. The threshold of the stress detection algorithm has been set to **0.06** (for more information see Table 6.4).

6.3.1 Reference Data Requirements

In order to speed up the algorithm as much as possible, energy features of the reference recordings have to be provided beforehand, the same as BN features. For each recording, that may contain several sentences, there has to be one energy feature file and one BN feature file. This is a complete list of the data that has to be provided for each reference recording:

- A **WAV file** with the reference voice.
- A **bottleneck feature file**.
- An **energy feature file**.
- An **XML file** containing segmentation of the recording. This has to be prepared ahead (by an ASR system) and has to contain segmentation to segments (one segment is usually one or a few short sentences), words, syllables and phonemes. For this thesis, all segmentation files were provided by supervisor.

This chapter described improvements made to the original pronunciation assessment algorithm. In the application, outputs of the improved method are displayed to the user in the form of a corrective feedback. The following chapter describes how the corrective feedback was designed.

Chapter 7

Corrective Feedback

Corrective feedback is the third and last area of improvement of the original application. It is probably the most important part of the application from the user point of view because it is not a perfect algorithm what helps the user learn but a smart computer-to-human communication. Of course, it would be impossible to present a meaningful corrective feedback to the user without having a good error detection algorithm, but a great detection algorithm without any feedback at all would be literally useless to the user.

Given the algorithms described in the previous chapter, the goal is to present their outputs in the application's frontend in a simple and clear way that will help the user learn. In this chapter, the design of the corrective feedback is discussed.

7.1 Algorithm Outputs

As already mentioned, the improved error detection algorithms were designed with regard to the corrective feedback. The output of the acoustic similarity assessment consists of information about whether or not each word has been pronounced correctly. Then, the stress error detection algorithm provides information about primary stress correctness for each syllable of each word. That means that each syllable is assigned one of the following three classes: correct primary stress, missing primary stress or extra primary stress.

7.2 Displaying Specific Errors

The new feedback is more sophisticated than the feedback in the original application. In addition to colouring the correctly pronounced words in green and incorrectly pronounced words in red, more specific feedback is provided. Given the algorithm outputs, the corrective feedback was designed as a list of erroneous words. For each word, there is a written explanation of what exactly the error was, and there are buttons enabling the user to compare the word's reference pronunciation with their own. Apart from the written explanation, stress errors are also displayed visually. Syllables that should have been stressed and were not are underlined, and syllables that were incorrectly stressed by the user (they contain an extra primary stress) are coloured in red. Theoretically, if phoneme-level error detection was possible, this kind of corrective feedback would be able to display erroneous phonemes in a similar manner as it displays stress errors. Similarity score in percentages was completely removed from the application. An example of a list of specific mistakes is displayed in Figure 7.1.

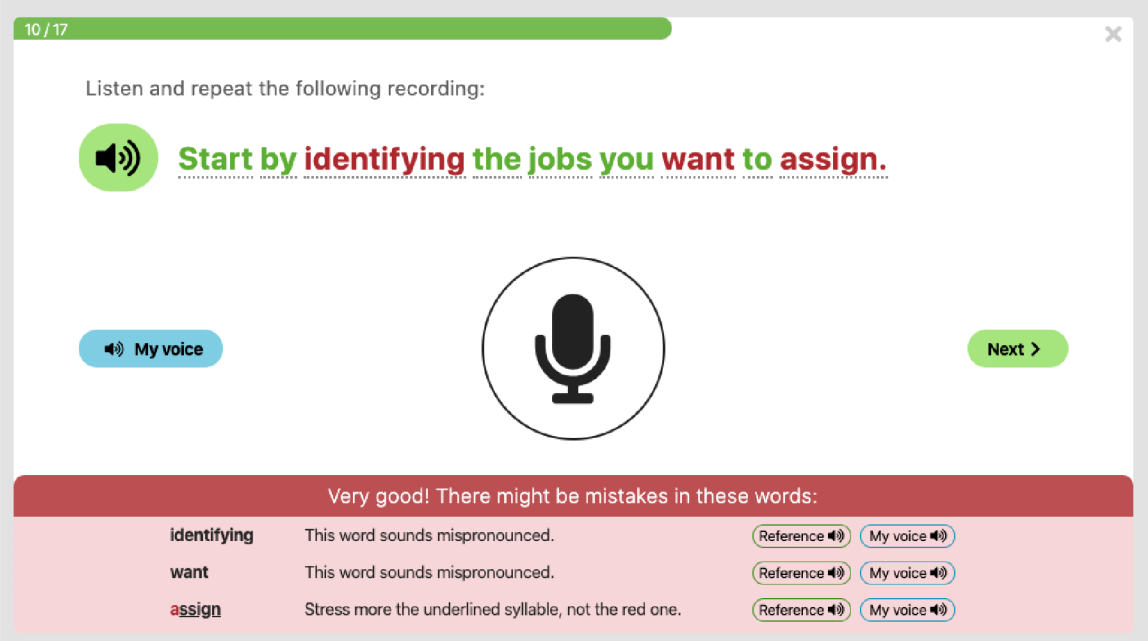


Figure 7.1: Application UI showing a list of pronunciation errors.

7.3 Global Feedback

In addition to directly displaying the outputs of the detection algorithms, each sentence is also evaluated globally. In fact, the list of mistakes is showed to the user only if there are 1-4 erroneous words in the sentence. If the sentence has been pronounced correctly (there was no mistake), the user is shown an encouraging message and may continue to the next exercise. On the other hand, if there are more than 4 mistakes, the whole sentence is marked as unintelligible and the user has to repeat the exercise in order to proceed.

Overall, the user can repeat an exercise anytime by clicking on the microphone button again. When the user is shown a list of mistakes and their descriptions, they can learn from them and try again. However, if they do not wish to repeat the exercise, they may still proceed to the next exercise, even though the algorithm has detected errors in the utterance. The reason for allowing the user to continue and not forcing them to repeat the exercise until their pronunciation is perfect is that in some cases, the algorithm is simply not right. Also, from the teaching point of view, it would not make sense to expect the student to be able to correct all their mistakes immediately. This is also the reason why the user is allowed to proceed after their pronunciation was evaluated as unintelligible three times. Figures 7.2 and 7.3 show how the corrective feedback looks like in case of a correctly pronounced sentence and an unintelligible sentence.

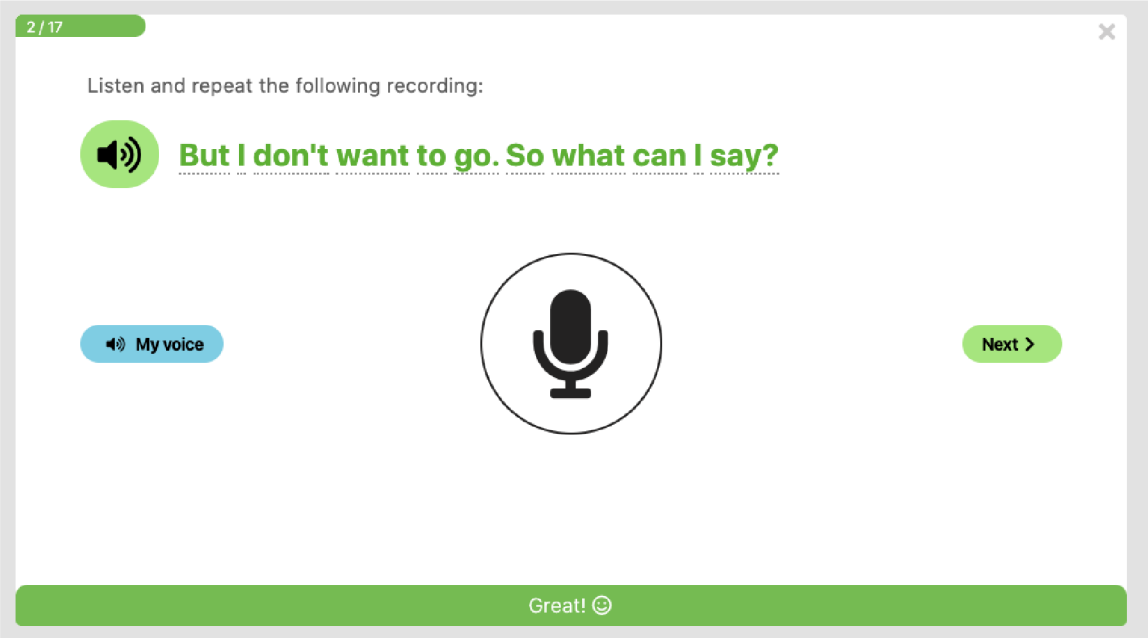


Figure 7.2: UI showing feedback on a correctly pronounced sentence.

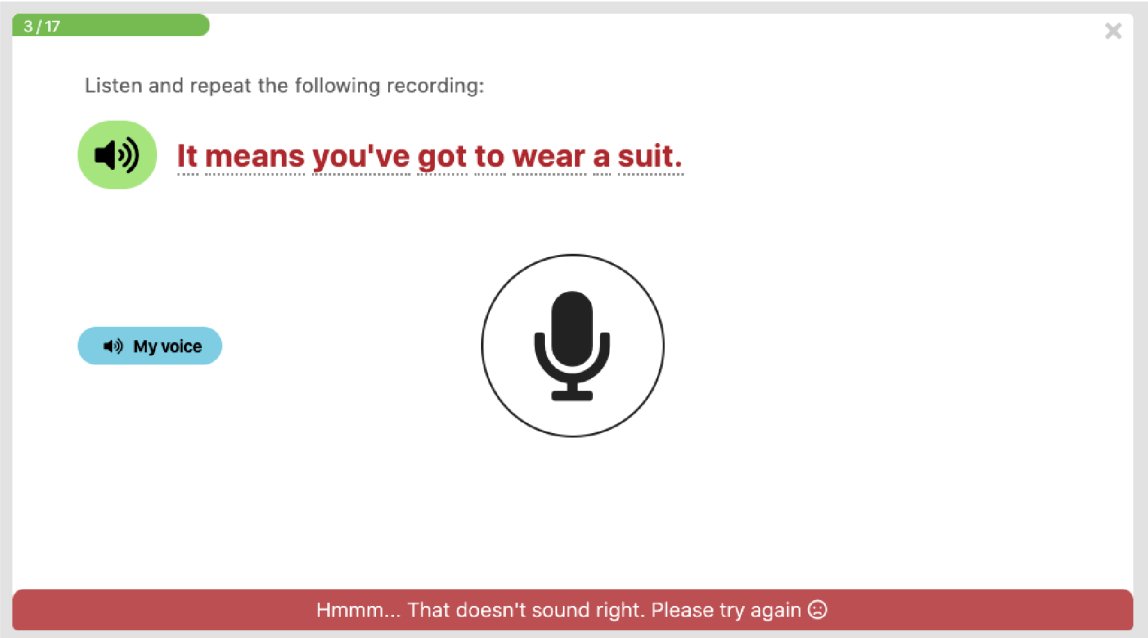


Figure 7.3: UI showing feedback on an unintelligible sentence.

Chapter 8

Testing

All three areas of improvement discussed in this thesis were tested. First, user testing was performed to find out how users interact with the UI. Second, the similarity assessment and stress error detection algorithms were tested by an expert on teaching the English language in order to obtain accuracy on real data. And finally, corrective feedback was evaluated on users to be able to conclude whether or not it meets its goal to help users learn.

8.1 User Interface Testing

User testing was performed in an early stage of the work - during the collection of the non-native speaker dataset. In this stage, new user interface was already implemented, but contained the old style of corrective feedback. The new corrective feedback was not yet developed. For that reason, the user testing did not include testing of the corrective feedback UI section.

User testing was performed on 8 users and it was found that overall, users interacted with the application fairly smoothly. Thanks to the testing, a number of design issues were discovered, and the corresponding parts of the new UI were redesigned.

Initially, some people did not know what to do at all, even though it was stated in the instructions at the top of the page. Therefore, the colour of the instruction text was changed from grey to black and letter spacing was increased, so that the instructions stand out more and attract the attention of the user better.

Secondly, 4 people out of 8 were confused about how to stop the recording and they did not realise that they have to click on the microphone button for a second time to end the recording. However, most of them were able to learn this behaviour after a couple of exercises, thanks to the help message that was displayed at the bottom, saying „Click the microphone when you finish your speech“.

Furthermore, the third design issue that was discovered during testing, was that some users were confused about the meaning of black-coloured words in the feedback. The meaning of green words (correct pronunciation) and red words (definitely mispronounced) was well understood, though. The black colour was a part of the original application's feedback. It used three colours: green for correct pronunciation, red for very bad pronunciation and black for a somewhat good pronunciation. Based on this finding, the word colouring was redesigned to use only two colours: green and red.

Finally, almost none of the 8 users clicked on the underlined words in the sentence even though the indicator in the form of underline was supposed to tell users that they can click

on the words. This feature would replay the user's pronunciation of the word. Additionally, on hover, the word's reference pronunciation would be replayed. This is one of the reasons why the final improved corrective feedback contains separate buttons for replaying and comparing the pronunciations for each erroneous word directly in the list of mistakes.

In addition, 21 people were asked about how easy to use the application was. Figure 8.1 shows that the feedback was mostly very positive.

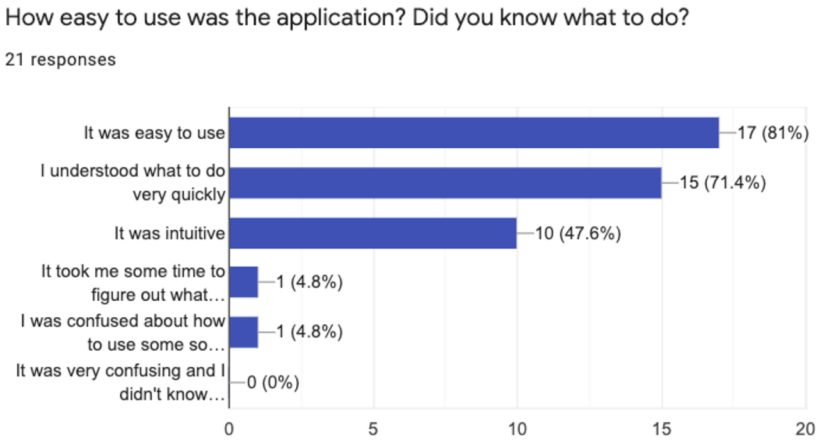


Figure 8.1: Most people stated that the application was easy to use.

8.2 Algorithm Testing

Because testing on the real data that was obtained from native and non-native English speakers earlier in the project was not possible, another means of testing had to be performed. For that reason, a separate evaluation tool was developed that enabled an English language expert to test the pronunciation evaluation algorithm on different pronunciation errors. The tool is displayed in Figure 8.2. Thanks to the expert, information about the real accuracy of the pronunciation evaluation algorithms was obtained, using the evaluation tool.

8.2.1 Expert Testing

The expert's task was to feed the application with utterances containing different pronunciation errors. This was possible because the expert was very familiar with the types of errors students make and knew how to create them. After the expert was shown the algorithm's result in the form of corrective feedback, they could correct the algorithm by simply clicking on the particular words. If they thought the algorithm has made a mistake, they could correct it by clicking on the particular red word and it would change colour to green, and the other way around. Sentences were evaluated on the word level. There was an attempt to evaluate the stress detection algorithm on the syllable level, too, using a checkbox which the expert was supposed to check in case the detailed feedback about the place or stress error was not correct. Unfortunately, the expert did not use the checkbox at all, so it was not possible to evaluate the algorithm on the syllable level.

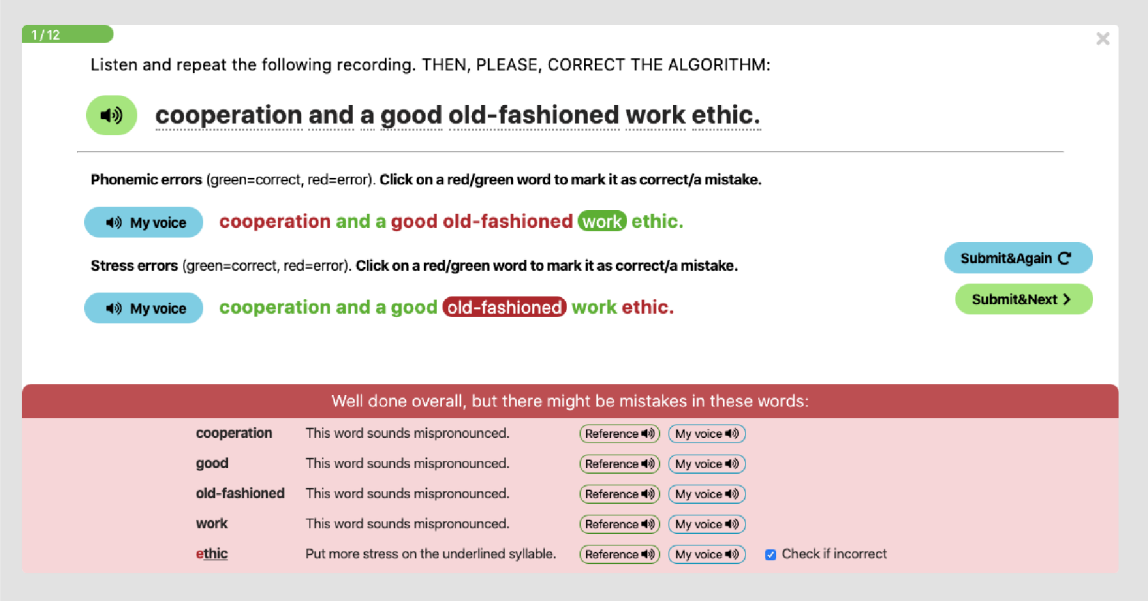


Figure 8.2: Expert tool used to evaluate the algorithm by experts.

75 sentences were evaluated on the word level using the evaluation tool. Table 8.1 shows detailed results of this evaluation. The accuracy on real data is, of course, lower than the results of experiments in Chapter 6. True positive rate of the stress detection algorithm did not exceed 24.7% (FPR was 5.3%) and the acoustic similarity pronunciation assessment had true positive rate not exceeding 37.1% (FPR was 2.8%). The table also shows the overall accuracy of the algorithms, but for assessing the results, TPR and FPR values are much more relevant.

Table 8.1: Results of expert testing on the word level.

	Similarity	**Stress**	**Together**
Total errors	97	73	-
Total words	932	932	-
True positives (TP)	36	18	-
False positives (FP)	44	24	-
False negatives (FN)	61	55	-
True negatives (TN)	791	835	-
TP rate (TPR)	**37.1%**	**24.7%**	-
FP rate (FPR)	5.3%	2.8%	-
Accuracy (ACC)	88.73%	91.52%	-
Sentences total	75	75	75
Feedbacks with FP	26	19	37
% of feedbacks without FP	**65.33%**	**74.67%**	**50.67%**
% of feedbacks with FP	34.67%	25.33%	49.33%

About 65% of feedback of similarity assessments alone did not contain any FP values. It was about 74% in case of error detection alone. When combined, however, only 50% of all feedback shown to the user did not contain any misleading information, which is quite

low. Ideally, the user should trust the assessment system, but if in 50% of cases they are given a feedback that contains a false alarm, will they trust such application?

8.2.2 User Testing

Feedback from real users on their perception of the algorithm's accuracy was also collected. This is a qualitative feedback obtained from 21 users using a questionnaire. From the graph in Figure 8.3, it can be seen that most users perceived that the algorithm catches their pronunciation mistakes fairly well. 76.2% of users rated the ability of the algorithm to detect pronunciation mistakes by number 4 or 5 on a 5-point scale. 23.8% of users rated the algorithm with a point 3.

Figure 8.3: Graph showing the users' evaluation of the application's ability to detect errors.

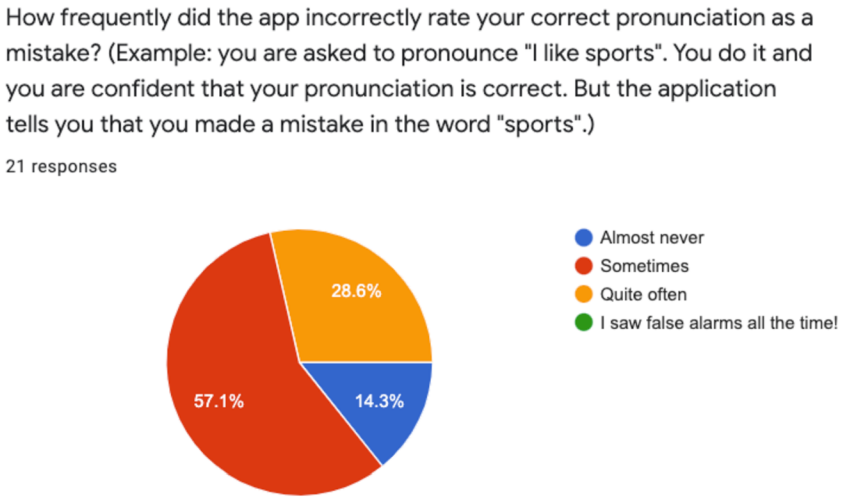


Figure 8.4: Graph showing that most users noticed the application sometimes gives false alarms.

On the other hand, 85.7% of users stated that the application displayed false alarms, as seen in Figure 8.4. Only 14.3% of users said they did not notice any false alarms.

8.3 Corrective Feedback Testing

The usefulness of the new corrective feedback was tested on 21 users using a questionnaire. The users were presented with the new application and were asked to test it. Then, they were presented with the older version of the application used for data collection. The older version used similarity score in percentages as a feedback instead of the list of mistakes. The users were asked to rate the helpfulness of both approaches and to compare the two styles of corrective feedback.

The vast majority (81%) of users rated the new corrective feedback as more useful than the old one (see Figure 8.5). Only 19% of participants did not think the new feedback was more useful. Figures 8.6 and 8.7 show how users rated the old and the new feedback on a 10-point scale.

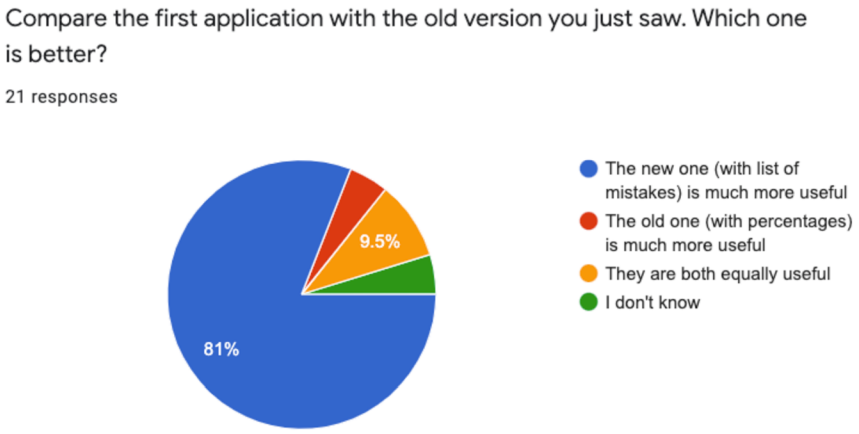


Figure 8.5: Most users found the new corrective feedback to be more useful than the old one.

According to Figure 8.8, users found the list of the specific words to be the most useful part of the corrective feedback. However, both the written explanation of the mistake and the buttons for pronunciation comparison were also rated as useful by the users.

In addition to user testing, the British English language expert who helped evaluate the algorithm also provided her opinion on the corrective feedback. She said the new corrective feedback was a big improvement compared to the previous one. She especially appreciated the list of specific pronunciation errors and the more detailed description of what exactly was wrong.

8.4 Conclusions

Based on the testing summarized above, it seems that while the user interface and corrective feedback were well-accepted by users, there is a lot of room for improvement regarding the error detection algorithm. While the algorithm does work to certain extent and the feedback on its accuracy is rather positive than negative, it would be desirable for the true positive

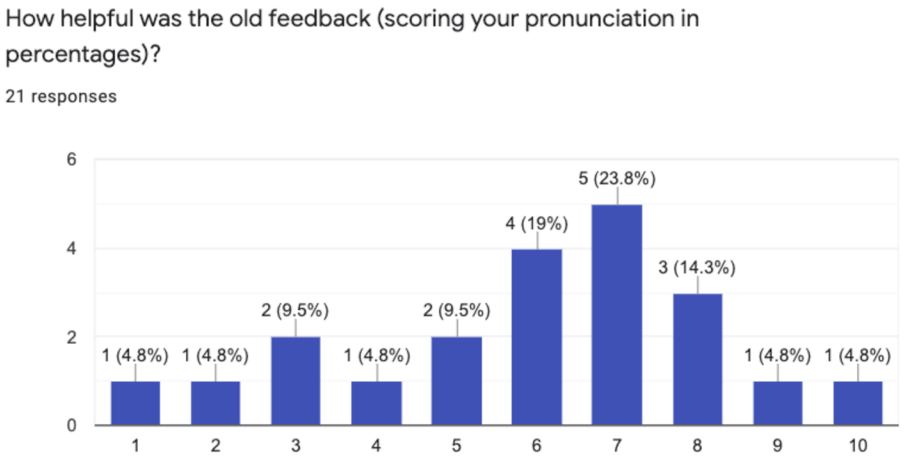


Figure 8.6: Ratings of the old corrective feedback. (1 = not useful, 10 = extremely useful)

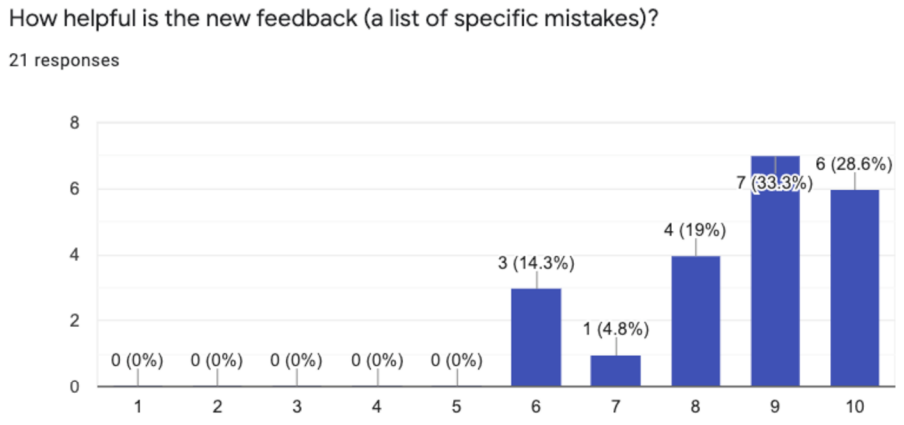


Figure 8.7: Ratings of the new corrective feedback. (1 = not useful, 10 = extremely useful)

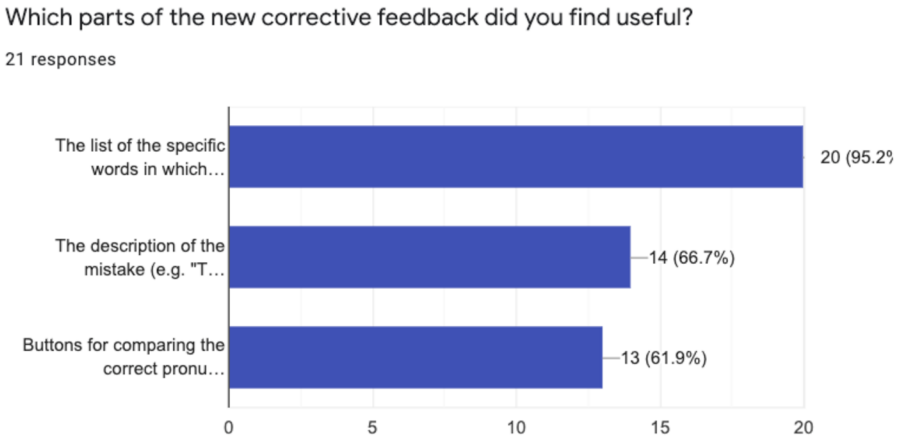


Figure 8.8: The list of specific words seems to be the most useful part of the correcie feedback.

rate to be higher and for the false positive rate to be lower. Based on the testing, the number of false alarms seems to be too high.

My suggestion is to improve the stress error detection algorithm by adding pitch into the feature array and by using some machine learning method. For assessing prosody, DTW seems to be a sufficient solution. However, regarding the similarity assessment method, I would suggest to replace the DTW algorithm with an ASR-based approach to phoneme-level pronunciation error detection. DTW seems to be able to provide only approximate results on pronunciation correctness, while efficient pronunciation teaching requires the algorithm to be able to detect specific errors accurately.

The user interface could be further improved by redesigning the microphone button so that recording it is more intuitive. Also, it might be worth changing the indicator in the form of an underline below words to something else due to the fact that users rarely used the feature.

And finally, if the algorithm was improved to be able to detect phoneme-level errors, corrective feedback could be extended to cover phoneme deletions, insertions and substitutions as well, which might be extremely helpful to the user.

Chapter 9

Conclusion

In this thesis, automatic pronunciation assessment and error detection techniques were discussed, and improvements of an English pronunciation training web application provided by supervisor were presented. The application was improved in three areas.

First, user interface was redesigned from a video player to a set of pronunciation exercises. It was tested on users and overall, the new user interface was very well accepted by users. In addition, the redesigned application was used to collect data from more than 800 native and non-native English speakers.

Second, the pronunciation assessment algorithm based on DTW was improved. The pronunciation assessment method in the original application consisted of the basic DTW algorithm and an acoustic similarity score. In this work, three areas of algorithm improvement were considered and corresponding experiments were performed: phoneme-level error detection, stress error detection and intonation assessment. All methods were designed with the goal to have an accurate error detection system that would be able to output specific errors that could be presented to the user in the form of a corrective feedback.

It was found that while stress and intonation assessment worked fairly well, it was very difficult to detect phoneme-level errors using DTW only. While using DTW for prosodic assessment might be sufficient, it seems that for accurate phoneme-level error detection, it is better to use ASR-based methods that provide more lexical and language-related information for phoneme recognition. This could be the greatest improvement of the application for the future. Regarding stress error detection, it could be further improved by adding pitch to the feature array. Intonation assessment could work better if more sophisticated algorithms, such as machine learning algorithms for intonation pattern classification, are utilized.

The final application contains word-level assessment of pronunciation and syllable-level stress error detection. Testing of the algorithms by an English language expert confirmed that there is still a room for improvement.

The third and last area of improvement is the corrective feedback. The corrective feedback presents to the user the results of pronunciation assessment and it is designed as a list of specific mistakes and their descriptions. It was tested on users and it was found that in comparison with the old corrective feedback, which used percentages only, the new feedback was rated by users as much more useful. The application's ability to communicate specific words where error has been made was most appreciated part of the corrective feedback.

To sum up, in this work, the application was improved in all three areas. User feedback on the user interface and the corrective feedback was very positive. However, even though

it was improved, the pronunciation assessment algorithm turned out to be still not accurate enough. The biggest improvement of the application for the future would be in the area of the algorithm.

Bibliography

[1] Anderson Hsieh, J., Johnson, R. and Koehler, K. The relationship between native speaker judgments of nonnative pronunciation and deviance in segmentais, prosody, and syllable structure. *Language learning.* 1st ed. Wiley Online Library. 1992, vol. 42, no. 4, p. 529–555.

[2] Arias, J. P., Yoma, N. B. and Vivanco, H. Automatic intonation assessment for computer aided language learning. *Speech communication.* 1st ed. Elsevier. 2010, vol. 52, no. 3, p. 254–267.

[3] Becchetti, L. P. *Speech Recognition: Theory and C++ Implementation.* 1st ed. John Wiley & Sons, 1999. ISBN 0-471-97730-6.

[4] Berndt, J. Using dynamic time warping to find patterns in time series. In: Press, A., ed. *AAAIWS'94: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining.* 1994, p. 359–370.

[5] Cutler, A. Errors of stress and intonation. In: Fromkin, V., ed. *Errors in linguistic performance : slips of the tongue, ear, pen, and hand.* San Francisco: Academic Press, 1980. ISBN 0-12-268980-1.

[6] Derwing, T. M. and Rossiter, M. J. The effects of pronunciation instruction on the accuracy, fluency, and complexity of L2 accented speech. *Applied language learning.* 1st ed. ERIC. 2003, vol. 13, no. 1, p. 1–17.

[7] Dlaska, C. Self-assessment of pronunciation. *System.* 1st ed. Elsevier. 2008, vol. 36, no. 4, p. 506–516.

[8] Doremalen, C. S. H. van. Automatic detection of vowel pronunciation errors using multiple information sources. In: IEEE. *2009 IEEE Workshop on Automatic Speech Recognition & Understanding.* 2009, p. 580–585. ISBN 978-1-4244-5478-5.

[9] Elliott, A. R. On the teaching and acquisition of pronunciation within a communicative approach. *Hispania.* 1st ed. JSTOR. 1997, no. 1, p. 95–108.

[10] Franco, L. D. V. R. O. Combination of machine scores for automatic grading of pronunciation quality. *Speech Communication.* 1st ed. 2000, vol. 30, 2-3, p. 121–130.

[11] Franco, L. R. M. B. H. Automatic detection of phone-level mispronunciation for language learning. In: International Speech Communication Association. *Sixth European Conference on Speech Communication and Technology.* 1999.

[12] Gazdík, P. *Automatické hodnocení anglické výslovnosti nerodilých mluvčích.* 2019. Master's thesis. Vysoké učení technické v Brně, Fakulta informačních technologií. Supervisor Ing. Kateřina Žmolíková.

[13] Grézl, F., Karafiát, M., Kontár, S. and Cernocky, J. Probabilistic and bottle-neck features for LVCSR of meetings. In: IEEE. *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07.* 2007, vol. 4, p. IV–757. ISBN 1-4244-0727-3.

[14] Imoto, K., Tsubota, Y., Raux, A., Kawahara, T. and Dantsuji, M. Modeling and automatic detection of English sentence stress for computer-assisted English prosody learning system. In: International Speech Communication Association. *Seventh International Conference on Spoken Language Processing.* 2002.

[15] Keogh, M. J. Derivative dynamic time warping. In: SIAM. *Proceedings of the 2001 SIAM international conference on data mining.* 2001, p. 1–11. ISBN 978-0-89871-495-1.

[16] Kim, C. and Sung, W. Implementation of an intonational quality assessment system. In: *Seventh International Conference on Spoken Language Processing.* 2002.

[17] Kim, Y.-J. and Beutnagel, M. C. Automatic assessment of American English lexical stress using machine learning algorithms. In: International Speech Communication Association. *Speech and Language Technology in Education.* 2011.

[18] Lee, G. G., Lee, H.-Y., Song, J., Kim, B., Kang, S. et al. Automatic sentence stress feedback for non-native English learners. *Computer Speech & Language.* 1st ed. Elsevier. 2017, vol. 41, no. 1, p. 29–42. ISSN 0885-2308.

[19] Li, K., Mao, S., Li, X., Wu, Z. and Meng, H. Automatic lexical stress and pitch accent detection for L2 English speech using multi-distribution deep neural networks. *Speech Communication.* 1st ed. Elsevier. 2018, vol. 96, no. 1, p. 28–36.

[20] Menzel, W. and Atwell, e. a. The ISLE corpus of non-native spoken English. In: Gavrilidou, M., ed. *Proceedings of LREC 2000: Language Resources and Evaluation Conference.* April 2000, vol. 2, p. 957–964. ISBN 2-9517408-6-7.

[21] Rodman, R. R. *Computer Speech Technology.* II.th ed. Artech House, Inc., 1999. ISBN 0-89006-297-8.

[22] Russell, K. *Identifying sounds in spectrograms.* 2005 [cit. 2020-05-19]. Available at: https://home.cc.umanitoba.ca/~krussll/phonetics/acoustic/spectrogram-sounds.html.

[23] Sluijter, A. M. and VAn HEUVEn, V. J. Acoustic correlates of linguistic stress and accent in Dutch and American English. In: IEEE. *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96.* 1996, vol. 2, p. 630–633. ISBN 0-7803-3555-4.

[24] Strik, K. d. W. F. C. C. Comparing different approaches for automatic pronunciation error detection. *Speech Communication.* 1st ed. Elsevier. 2009, vol. 51, no. 10, p. 896–905.

[25] Tamburini, F. Prosodic prominence detection in speech. In: IEEE. *Seventh International Symposium on Signal Processing and Its Applications, 2003. Proceedings.* 2003, vol. 1, p. 385–388. ISBN 0-7803-7946-2.

[26] Tepperman, J. and Narayanan, S. Automatic syllable stress detection using prosodic features for pronunciation evaluation of language learners. In: IEEE. *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.* 2005, vol. 1, p. I–937. ISSN 1520-6149.

[27] The SciPy community. *SciPy v0.19.1 Reference Guide*. 2017 [cit. 2020-05-19]. Available at: https://docs.scipy.org/doc/scipy-0.19.1/reference/generated/scipy.signal.hamming.html.

[28] Weigelt, L. F. Plosive/fricative distinction: The voiceless case. *The Journal of the Acoustical Society of America*. 1st ed. 1990, vol. 87, no. 6, p. 2729–2737.

[29] Wells, J. C. *English intonation PB and Audio CD: An introduction*. 1st ed. Cambridge University Press, 2006. ISBN 978-0-521-86524-1.

[30] Witt, S. J. Phone-level pronunciation scoring and assessment for interactive language learning. *Speech Communication*. 1st ed. 2000, vol. 30, no. 2, p. 95–108. ISSN 0167-6393.

[31] Witt, S. Automatic Error Detection in Pronunciation Training: Where we are and where we need to go. In: Engwall, O., ed. *Proceedings of International Symposium on automatic detection on errors in pronunciation training*. 2012, vol. 1. ISBN 978-91-7501-402-9.

[32] Yaman, S., Pelecanos, J. and Sarikaya, R. Bottleneck features for speaker recognition. In: *Odyssey 2012-The Speaker and Language Recognition Workshop*. 2012, p. 105–108.

[33] Zhao, X., O'Shaughnessy, D. and Minh Quang, N. A processing method for pitch smoothing based on autocorrelation and cepstral F0 detection approaches. In: IEEE. *2007 International Symposium on Signals, Systems and Electronics*. 2007, p. 59–62. ISBN 1-4244-1448-2.

Appendix A

Content of the Storage Medium

This is the list of content on the storage medium enclosed to this thesis:

- Text and source files of this document
- Source code of the improved application
- Source code of the expert evaluation tool
- Source code of all experiments performed in this work
- Example subset of the collected dataset
- README.txt

  The README.txt file contains a more detailed explanation of what is included.s

Appendix B

Consonant Types

Consonants can be classified into types based on their place of articulation and manner of articulation.

There are 8 places of articulation in the English language and all of them are displayed in Figure 2.1. Each place of articulation determines a consonant type.

1. **Bilabial** consonants are created when the two lips momentarily come together and obstruct the stream of air. Examples: *p*, *b*, *m*.

2. **Labio-dental** consonants are produced when the lower lip is raised toward the upper teeth (*f*, *v*).

3. **Dental** consonants are created by obstructing the air stream using the tongue tip and the teeth (*th*).

4. **Alveolar** consonants. The front of the tongue is raised toward the alveolar ridge (behind the upper front teeth). Examples: *t*, *d*, *n*, *s*, *z*, *l*...

5. **Post alveolar** consonants. Created by moving the tongue toward the post-alveolar part of the hard palate (*sh*, English *r*).

6. **Palatal** consonants. Obstruction is made in the palatal area and the examples are the initial sounds of *year*, *church*, *judge* or *hue*.

7. **Velar** consonants. Obstruction is at the soft palate (*k*, *"hard c"* in *coat*, *"hard g"* in *goat*).

8. **Glottal** consonants. Obstruction is at the vocal cords (*h* in *hat*, and sounds like *"uh"*, *"oh"*).

There are 7 manners of articulation, each representing one consonant type.

1. **Stop consonants (stops)**. There is a momentary but complete blockage of the air stream (initial sounds of *ball*, *doll*, second sounds of *spill*, *still*).

2. **Aspiration**. The stop is held so long that air pressure is built up behind the obstruction and then released (*pill*, *till*).

3. **Nasal stop**. Air is blocked in the mouth but allowed to flow out the nose (final sounds of *beam*, *bean*, *bing*).

4. **Fricative**. Produced when two articulators are very close to each other but not touching and they cause turbulence (*f*, *v*, *th*, *s*, *z*, *sh*, *h*).

5. **Approximant**. Two articulators are close to each other but not close enough to cause turbulence (*y* in *yet*, *w* in *wet*, *r* in *red*).

6. **Affricate**. This is the combination of the he alveolar stop *t* and the palatal fricative *sh* (beginnings of *chump*, *jump*). This characteristic implies an interesting fact: *white shoes* said together without a pause sound almost the same as *why choose*.

7. **Flap**. Flap is produced when the tongue taps against the alveolar ridge. Occurs in American English as the middle consonants of *rider*, *writer*, *latter*, *ladder*.

Appendix C

Module Parameters

Table C.1 describes all customizable parameters of the final JavaScript application (jQuery module). The module can be included into a HTML website.

Table C.1: Overview of module parameters.

Parameter	**Description**
`recordingJsonpUrl`	the JSONp URL to load recording data from
`uploadUrl`	URL for uploading recorded segments
`width`	glplayer element width; if 0 then 100%
`height`	glplayer element height; if 0 then 100%
`autoStart`	if true, audio starts playing without user's interaction
`mode`	exercise mode: 'read' | 'repeat' | 'repeatWithSubtitles'
`randomModes`	if true, modes will be picked randomly for each segment
`selectionStrategy`	selection of segments from recordings: 'sequential' | 'random' | 'datacollection
`speakScoreCalibration`	1 - very hard, 2 - hard, 3 - medium, 4 - easy, 5 - very easy
`syllables`	true if recording files contain segmentations to syllables
`showHints`	if true, instruction bubbles will be shown to the user
`multipleRecordings`	true if more files with recordings will be used
`recordingJsonpUrlArray`	array of URLs to load recording data from (if multipleRecordings is set)
`segmentsFromEachRecording`	number of segments taken from each recording data; if 0, all we be taken