Improving Accent and Pronunciation

by

Jasmeet Singh
B.Tech., Kurukshetra University, 2012

A Project Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Improving Accent and Pronunciation

by

Jasmeet Singh
B.Tech., Kurukshetra University, 2012

Supervisory Committee

George Tzanetakis, Supervisor
(Department of Computer Science)

Yvonne Coady, Departmental Member
(Department of Computer Science)

Supervisory Committee

George Tzanetakis, Supervisor
(Department of Computer Science)

Yvonne Coady, Departmental Member
(Department of Computer Science)

ABSTRACT

Improving accent and pronunciation is a key factor for anyone learning a new language. This project is aimed to help people improve their accent and pronunciation. It is a web application that lets users record an audio word and compare it against the words stored on the server. The words stored on the server are audio files recorded by a person fluent in English. The comparison is a Dynamic time warping score showing how close or far is the user's pronunciation is when compared to the one stored on the server. The web application is designed in such a way that people can fork the code and use it for other languages easily. This project can also help people who are new to the English language and people who have speech disabilities.

Contents

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

my supervisor George Tzanetakis, for his continuous support, guidance, mentoring and for giving me this wonderful opportunity.

my parents Sukhdev Singh and Gurjeet Kaur, for everything.

my idol and my sister Aman, for fueling my ambitions.

'Do or do not, there is no try.'
- Master Yoda

DEDICATION

I dedicate this to my teachers, my family and friends.

Chapter 1

Introduction

Learning a new language is difficult. Harder yet is the accent and pronouncing the words correctly, especially as an adult with a busy life, or when the language is based on sounds that are very different than a speaker's native language. Pronouncing words correctly helps people become socially comfortable and easily present their ideas in front of others. Incorrect pronunciation can also lead to an incorrect message. Unfortunately, most of the stress when learning a new language is given on grammar and adding more words to the vocabulary. People often try to use complex words with difficult pronunciations rather than just using the simple words to convey the message. Because most of the stress while learning a language is given on reading, writing and adding more words to the vocabulary there are countless free and paid resources on the internet to improve them. Consequently, pronunciation and accents which form the basis of communication do not have much dedicated easy-to-use resources on the internet. In order to fill this gap, this project is an effort to provide a free easy-to-use tool that can help people to improve their pronunciation and learn a particular accent. The project which is a web application lets the user play audio words, record their pronunciation of the word and give a score depending on how close or far their accent is from the one in the dictionary. The project uses dynamic time warping (DTW)[2] to compute the similarity between two audio files.

1.1 Structure of the Project Report

This section provides information on what each Chapter of this Report will discuss:

Chapter 1 gives an overview of what this web application is all about.

Chapter 2 talks about the problem being considered in this project and related work.

Chapter 3 discusses the technical design of the web application.

Chapter 4 explains the application architecture, implementation challenges and the algorithm used.

Chapter 5 evaluates the application and talks about its limitations and future work.

Chapter 6 concludes the purpose and implementation of the project.

Chapter 2

Problem and Related Work

Language attitude studies has shown that language is a powerful social force that does more than convey intended referential information [3][4]. Research shows that "an American may think that a stranger is 'cultured' and 'refined' simply because his or her accent is recognized as British" [3]. Such beliefs can create bias in social interaction. There are studies that confirm that accents have a great impact on listeners' attitudes or perceptions toward speakers [4]. Many studies in the field of language attitudes have indicated that a speaker's accent can influence their chances of success in an employment interview. In other words, an accent can play an important role in the perceptions and attitudes of a person's characteristics. Generally, the results of the studies that were conducted have indicated that standard accented speakers are favored for prestigious jobs, whereas the nonstandard accented speakers are favored for less desirable jobs [4]. One study shows that Canadian listeners preferred German-accented speakers for high-status jobs compared to those with West Indies and South Asian accents, although British speakers were predictably more preferred to German-accented speakers [5]. Studies have also demonstrated that foreign accented applicants, compared to local accented applicants, were rated less suitable for higher-status jobs, and were said to be more suitable for the lower-status job. The accent may have an effect on students perception of a teacher's performance in the classroom. One study showed that Korean students did not consider Korean accented English as a good model for learning English. They termed non-native accent as "bad accent" and some of them felt that they did not need to understand Indian, Singaporean, or Filipino English and the results showed that they could not distinguish these varieties [4]. All these studies shows how accent plays an important role in our day to day life. Through this project, an effort is made to build a tool which people

can easily use to learn and practice particular accents.

This project is a small step to build a tool which can be used by non-native English speakers to improve and practice the native English accent. The project does not deal with sentences but words. It uses the Dynamic time warping[1] algorithm to calculate how close or far two audio files are from each other. The DTW algorithm is also used in various other fields such as such as data mining, information retrieval and bioinformatics [2]. The application takes input from the user in the form of audio. That audio is compared to the audio recorded by a native English speaker. The comparison is the DTW score. The audios are first converted in the Mel-frequency cepstral coefficients (mfcc)[6] feature representation and then fed into the DTW algorithm. The algorithm gives a score which tells how close or far two audio files are in terms of similarity. A score of zero means the two audio files are identical, a score close to zero means the two files are similar and a large score means the two files are quite different.

Since all the code is on GitHub[7], the interested audience can also use it for other languages by keeping the audio dictionary files of the language on Amazon's S3[8] Bucket. Also, the REST[9] APIs are built so that they can be consumed on the web, and any future mobile applications. The backend is a Docker[10] container which can be helpful to the interested audience to use the code easily on any machine.

Chapter 3

Design

3.1 Technologies Used

One of the easiest ways for comparing two accents is to give a score of the difference between two such accents. A higher score means a greater difference between the user's accent and the dictionary accent. A lower score means the user's accent more closely matches the dictionary accent. The final target was to develop an application that could be easily accessible to the interested audience and is easily scalable in the future. Therefore, building a web application with dedicated REST[9] APIs was a logical step.

One of the most popular front-end frameworks for making web applications is Angular[11]. Angular 5 along with HTML, CSS, JavaScript, and TypeScript were used to build the frontend. A well known third-party software was used to record the audio on the frontend. Flask[12], a popular python framework was used to build the rest APIs. The sole purpose of creating rest APIs was to let others use them in their own web or mobile applications. To compute how close or far two audio signals are, we used Librosa[13] another famous python framework. Github[7] was used for the version control and the REST[9] APIs are contained inside a Docker[10][14] image. Anyone can download the code from the GitHub[7], which is kept open, and run it on their own machine with simple docker commands.

3.2 Data

The audio dictionary used in the backend to compare against the user's audio is a collection of 4876 audio files. The files are recorded by a female speaker fluent in English. All the files are in the FLAC format (Free Lossless Audio Codec). These files are downloaded from shtooka.net and all the files are under the license "Creative Commons Attribution 3.0 United States". The license allows its users to

- **Share** — copy and redistribute the material in any medium or format
- **Adapt** — remix, transform, and build upon the material, for any purpose, even commercially.

3.3 Application

The web application is a single page application. It lets the user search for the word in the dictionary and listen to the pronunciation of the available word. Playing the audio file can help the user to learn new and complex word pronunciation. The application lets the user record their own version of the pronunciation of the word. For this purpose, we used RecordRTC[15] library. The user can play their recordings and record new ones. The recorded audio can be compared against the one stored in the dictionary through the REST[9] API running on a Flask server. The app takes an audio file as input and returns the DTW[2] score, which tells how similar or different two audio files are. A DTW score of zero means two files are identical, close to zero means files are more similar and a value far from zero means two files are quite distinct.

To calculate the DTW score, first, we need to extract features from the sound files. In sound processing, the mel-frequency cepstrum[16] (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum[17] on a nonlinear mel[18] scale of frequency[16]. Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC. They are derived from a type of cepstral[19] representation of the audio clip[16]. Once features are extracted from the audio files they are fed into the Dynamic time warping algorithm which gives the total alignment cost called the DTW score.

To store the dictionary audio files AWS S3[8] bucket was used. To fetch these audio files required by the REST APIs, running on the flask server, the AWS S3 rest

APIs were used. All the code for the frontend[20] and backend[21] is kept on Github[7] and is public. The backend code is contained inside the Docker so that the interested audience can clone the code and run on their own machine.

The Figure 3.1 shows the screen-shot of the web application running on a browser and Figure 3.2 is the distance between the mfcc[6] features of two audio files over the frames. The Docker container contains rest API for both DTW score and spectrum shown in figure 3.2, but the web application returns the DTW score on the screen, to make things simpler for the general user.

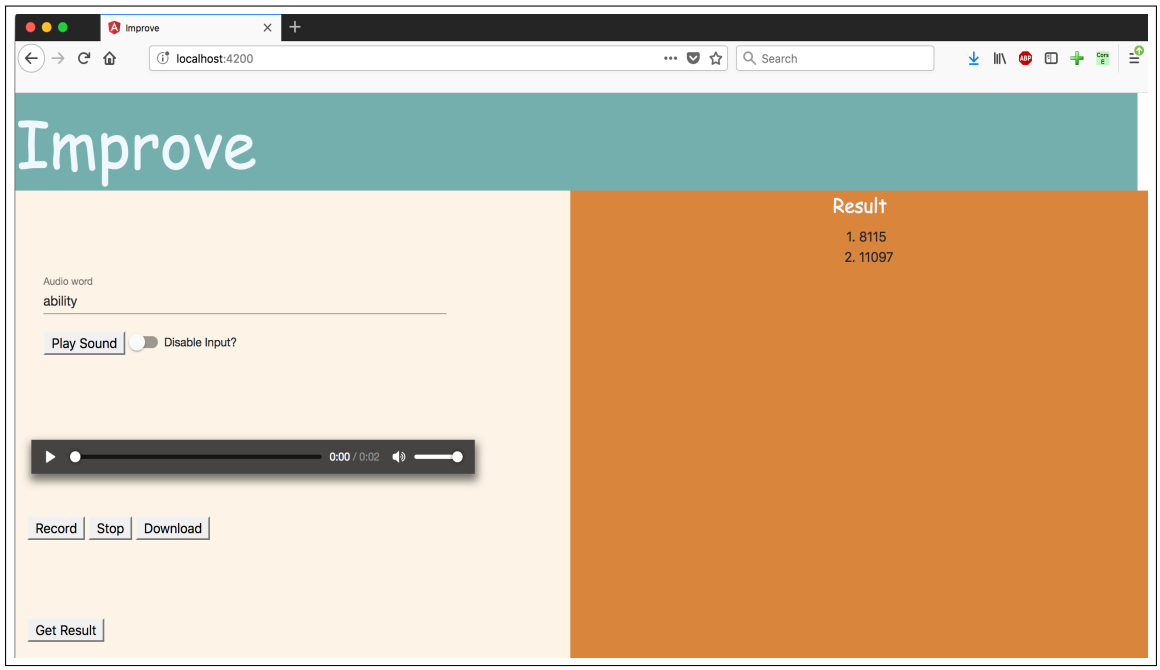

Figure 3.1: Web Application screen shot

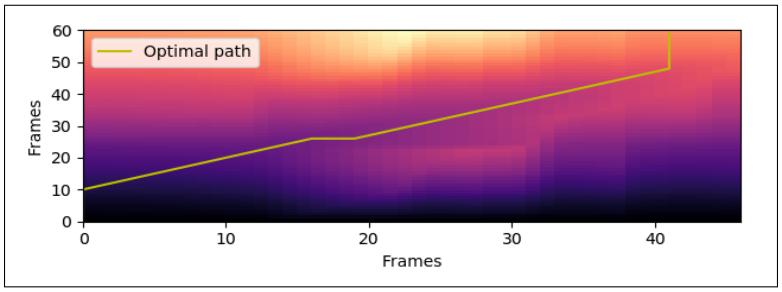

Figure 3.2: Line showing distance between mfcc features of two audio files

Chapter 4

Implementation

4.1 Application Architecture

The front end of the application is built using Angular 5[11], Node, Typed Script, HTML, and JavaScript. The front-end app uses RecordRTC[15] library, which helps to record the audio files and let users play, upload and download those audio files. To keep the UI simple and modifiable material design has been used from the Angular's official documentation. Package.json file in Angular lists all the dependencies used in the front-end.

For the backend a number of tools and technologies are used. The REST APIs were made using the Python framework Flask. The APIs are designed in such a way that they can be consumed from the web and mobile devices easily. Python's Librosa framework was used to help calculate the difference between two audio files. Librosa[13] provides a lot of features and works with a large number of different audio file formats. Figure 4.1 shows the architecture of the application.

The Rest APIs compare the audio file sent by the user with the corresponding audio file stored on the server. There are total 4878 audio files (spoken dictionary words) stored on the server. For better implementation of the rest API and to use the web resources properly, we stored all the audio files on Amazon S3[8] because it is a better way to manage file when compared to storing files as a BLOB (Binary large object) in a database. The backend extensively uses AWS web services to fetch the audio file from the Amazon S3 bucket when required.

The purpose of the whole application is to let other people use the code and contribute to the application easily. For this purpose, all the code has been put on

Github[7] with proper README files which has helpful information for setting up the development environment. Since the backend has been built using the Docker, all the code is machine independent. Docker can help others easily run the code on the local machine and modify the backend for other languages by adding proper audio dictionary files[10].

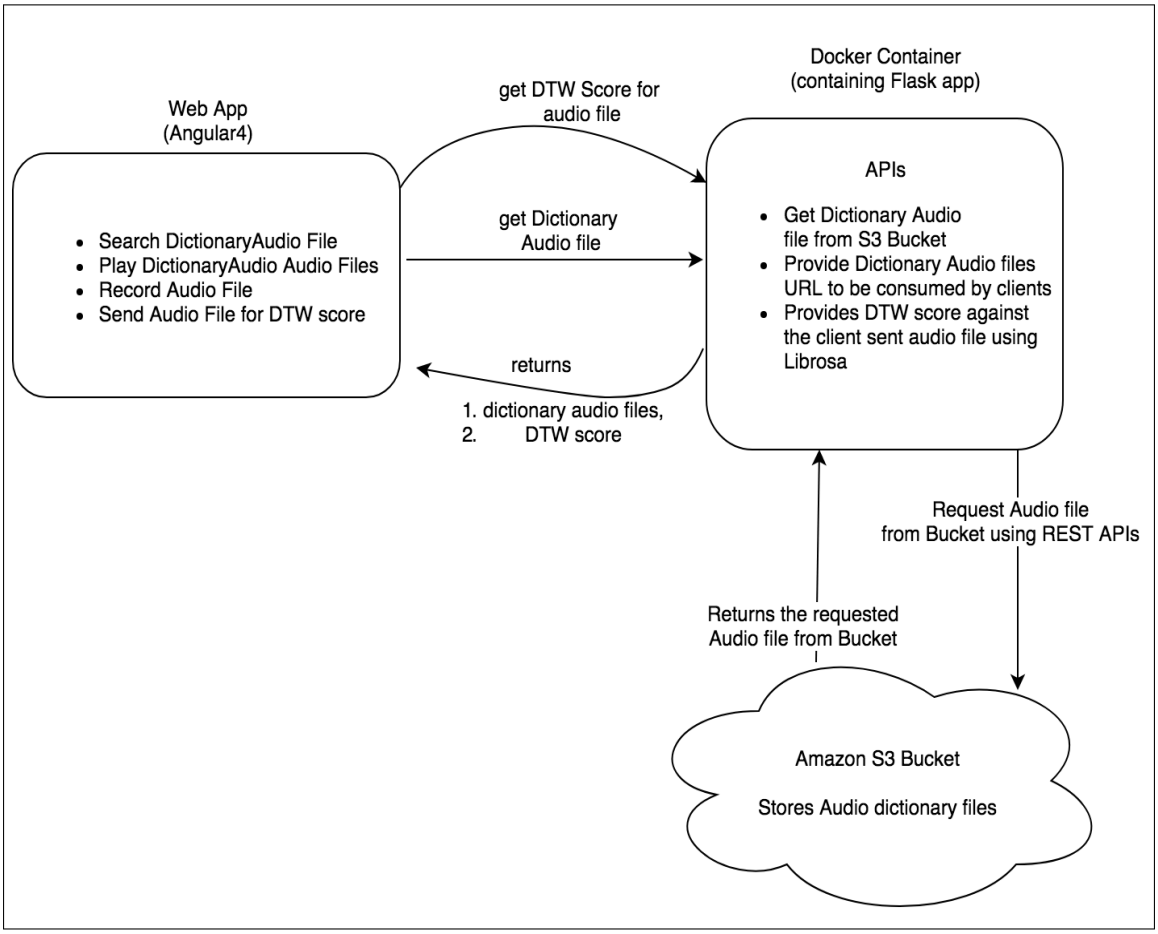

Figure 4.1: Application Architecture

4.2 Implementation Challenges

The project aims to effectively give users feedback for their audio speech in the form of a DTW score. The lower the score means user input is similar to the dictionary standard. The higher the score means there is a greater distinction between the two audio files. There were a couple challenges while implementing the web app:

1. Python's Librosa package was throwing a number of exception on the local machine. When running the librosa package on a Virtual machine the performance of virtual machine was very low in terms of speed, and it was taking more time in development than required. This was solved by using docker. Docker is a computer program that performs operating-system-level virtualization also known as containerization[14].

2. Recording audio files and attaching them to rest API was a challenge. There are many recording modules that could be consumed by the Angular front end, but many of them did not work in every browser. After some research it was found that the RecordRTC[15] library supports a large number of browsers and has huge community support.

3. There are a large number of audio dictionary files. Storing them as BLOB (Binary large object) in the database is not a best practice. For this purpose the AWS S3[8] Bucket was used. All the files were stored on Amazon cloud and consumed by using AWS REST APIs when required.

4.2.1 Algorithm

The process of comparing two music/audio files happens in two steps. First, the two files are transformed to suitable sequence features. The most commonly used feature extraction method in speech recognition is Mel-Frequency Cepstral Coefficients (MFCC) [6][22]. This feature extraction method was first mentioned by Bridle and Brown in 1974 and further developed by Mermelstein in 1976 and is based on experiments of the human misconception of words [23]. To extract a feature vector containing all information about the linguistic message, MFCC mimics some parts of the human speech production[24] and speech perception[25]. MFCC mimics the logarithmic perception of loudness and pitch of human auditory system, and tries to eliminate speaker dependent characteristics by excluding the fundamental frequency and their harmonics[6]. Second, the features are brought into temporal correspondence with a technique known as Dynamic time warping[2] (DTW).

The main objective of DTW is to compare two given sequences of features. In this case, a sequence of MFCC features were used for the comparison of two different audio files. Let the first sequence be denoted by $X:=(x_1,x_2,...,x_N)$ where N is the length of the first sequence and let the second sequence be denoted by $Y:=(y_1,y_2,...,y_M)$ where

M is the length of the second sequence. The two feature sequences do not have to be of the same length. The main goal of DTW is to find the optimal alignment between two sequences of features. As illustrated in the figure below the sequence X has length N=12 and sequence Y has length M=15.

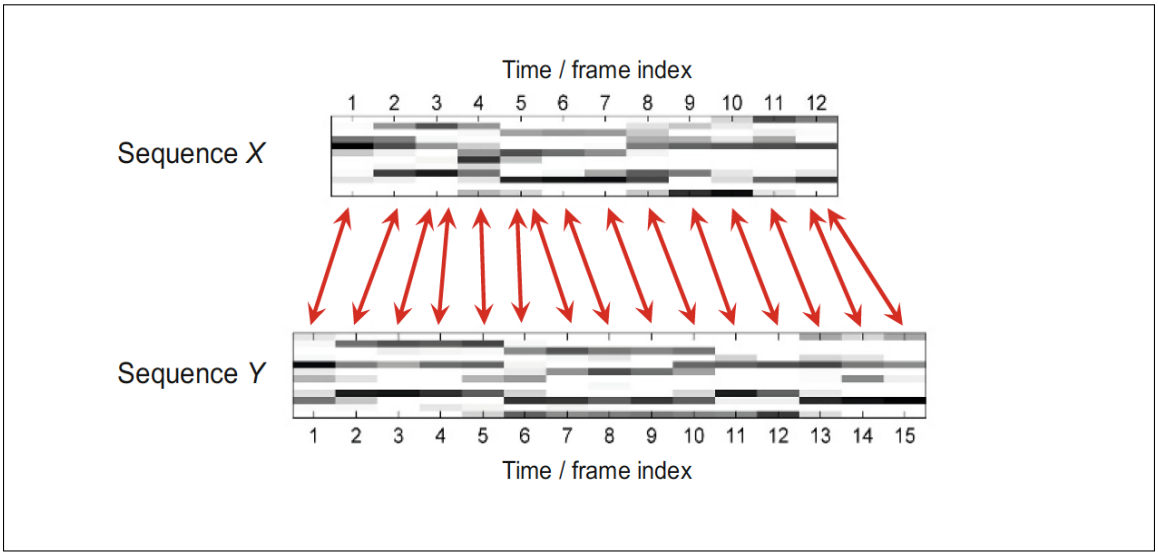

Figure 4.2: Alignment of two sequences of feature vectors. Aligned points are indicated by the arrows.[1]

The purpose of DTW is to compensate for the differences between the two sequences and find an optimal alignment between them. The DTW algorithm is also used in various other fields such as data mining, information retrieval, and bioinformatics.

4.2.2 How DTW works

Cost Matrix

The objective of the DTW is to compare two sequences defined by X:=(x_1,x_2,...,x_N) and Y:=(y_1,y_2,...,y_M). The sequences can be signals, feature sequences or any time series. To compare two different features a local cost measure is needed which is defined as the local distance measure. The cost is small (i.e. low if the features are similar) or large (i.e. high). The local cost measure for each pairing of elements from X with elements from Y is called a cost matrix C. Defined by C(n,m) := c(x_n,y_m) n $\in$ [1:N] and m $\in$ [1:M].

There are many ways to define the distance between two features. Cosine distance is beneficial in the computation of an entire cost matrix efficiently. Another measure is Euclidean distance, one of the most common, defined as $||x - y||$.

Warping Path

An warping path is a sequence P = $(p_1, p_2, ..., p_L)$ With p_l =(n_l, m_l) $\in$ [1:N] X [1:M] for l $\in$ [1:L] satisfying the following three conditions:

- **Boundary condition**: p_1 = (1,1) and p_l =(N,M)
- **Monotonicity condition**: $n_1 \leq n_2 \leq ... \leq n_L$ and $m_1 \leq m_2 \leq ...m_L$
- **Step size condition**: $p_{l+1} - p_l \in (1,0), (0,1), (1,1)$ for $l \in [1 : L-1]$

An (N,M) - warping path P = $(p_1, p_2, ..., p_L)$ defines an alignment between two sequences X and Y by assigning the element x_{n_l} of X to the element y_{n_l} of Y

Optimal Warping path and DTW Distance

The total cost of warping path between two sequences X and Y is defined as

$\boldsymbol{C_p(X,Y) := \sum_{l=1}^{L} C(x_{n_l}, y_{m_l} = \sum_{l=1}^{L} C(n_l, m_l)}$

where p is the warping path[1].

A warping path is considered good if its total cost is low and bad if its total cost is high[1]. The DTW algorithm finds the minimal total cost warping path P* among all the warping paths. Which leads to the definition of DTW distance denoted as DTW(X,Y) between the two sequences X of length N and Y of length M, defined as total cost of an optimal (N,M) warping path P*

$DTW(X,Y) := c_{P^*} = min\{c_P(X,Y)\}$ where p is the warping path [1].

DTW(X,Y) is referred as "DYW distance" between two sequences X and Y. For the optimal warping path P* the cost of all possible warping paths is computed and the one with minimal cost is chosen. However the approach in unfeasible for large N and M. The following algorithm computes DTW in O(NM) time:

Algorithm: DTW

Input: Cost matrix $\mathbf{C}$ of size $N \times M$
Output: Accumulated cost matrix $\mathbf{D}$
Optimal warping path P^*

Procedure: Initialize $(N \times M)$ matrix $\mathbf{D}$ by $\mathbf{D}(n,1) = \sum_{k=1}^{n} \mathbf{C}(k,1)$ for $n \in [1:N]$ and $\mathbf{D}(1,m) = \sum_{k=1}^{m} \mathbf{C}(1,k)$ for $m \in [1:M]$. Then compute in a nested loop for $n = 2,\ldots,N$ and $m = 2,\ldots,M$:

$$\mathbf{D}(n,m) = \mathbf{C}(n,m) + \min\{\mathbf{D}(n-1,m-1), \mathbf{D}(n-1,m), \mathbf{D}(n,m-1)\}.$$

Set $\ell = 1$ and $q_\ell = (N,M)$. Then repeat the following steps until $q_\ell = (1,1)$:

Increase ℓ by one and let $(n,m) = q_{\ell-1}$.
If $n = 1$, then $q_\ell = (1,m-1)$,
else if $m = 1$, then $q_\ell = (n-1,m)$,
else $q_\ell = \text{argmin}\{\mathbf{D}(n-1,m-1), \mathbf{D}(n-1,m), \mathbf{D}(n,m-1)\}$.
(If 'argmin' is not unique, take lexicographically smallest cell.)

Set $L = \ell$ and return $P^* = (q_L, q_{L-1}, \ldots, q_1)$ as well as $\mathbf{D}$.

Figure 4.3: DTW algorithm based on dynamic programming.[1]

Chapter 5

Evaluation, Limitations and Future Work

5.1 Evaluation

Five words were chosen from the dictionary of audio words in order to test the application. Benefit, Estimate, Factor, Specific and Theory were the words chosen randomly from Dr. Averil Coxhead's list of 2000 most frequently occurring words in the English language [26]. Three speakers were chosen to conduct tests of the application by recording the five words chosen. These recordings were compared to the dictionary audio recordings, and each other, for validation using the DTW algorithm. It was predicted that once the speakers listened to the dictionary recordings and re-recorded the word the DTW score would fall.

The speakers chosen to conduct the tests came from various native languages and different genders. This is specifically done to have a wide scope of accents and pronunciations. Figure 5.1 includes specifications for each of the speakers.

Speaker	Gender	Langauage
Spearker 1	Female	Native English Speaker
Spearker 2	Male	Non-Native English Speaker
Speaker 3	Male	Non-Native English Speaker

Figure 5.1: Speaker's details

First, a baseline test was conducted comparing dictionary audio to itself which resulted in a DTW score of zero. The first test conducted was to have each speaker

record the five words as they spoke naturally. The second test was to have each speaker listen to the dictionary audio before recording again. The second test was done to confirm the hypothesis. The speaker's audio was compared to the dictionary and the DTW score was recorded for test one and test two. Figure 5.2 through 5.6 shows the result of each of these tests.

Finally, validity tests were conducted to confirm the validity of the DTW algorithm. These included comparisons where one speaker's audio is compared to a completely different word spoken by the same person, and where one non-native English speaker's audio is compared to a native English speaker's audio of the same word. This was done to compare differences in accent and pronunciation from the dictionary audio.

It was found that when a speaker listened to the dictionary audio prior to recording, their accent and pronunciation greatly improved. In fact, the DTW score decreased in 86 percent of the cases. It is assumed that this is because the speaker was attempting to recite the dictionary audio.

As the validity tests found, when comparing a speaker's word to an entirely different word the DTW decreased. This could be attributed to the comparison being done between the same speaker with different words. As expected, when comparing a non-native English speaker to a native English speaker the DTW score was higher than when the non-native English speaker was compared to the dictionary audio. The reason may be because the native English speaker has a different accent or pronunciation than the dictionary audio.

Overall, the DTW score comparing the speaker's second recording to the dictionary audio was improved. This proves that, when hearing the word in the standard accent and pronunciation, a user can, in fact, improve their speech.

5.1.1 Limitations

Some of the limitations of the project are as follows:

1. There are more than 4000 words that can help users to practice those words. Even though 4000 is huge, some of the basics words are missing in the database.

2. The audio files stored on the server are in a female voice. No experiments are done regarding what will happen to results if the user audio file is compared against the male audio dictionary words.

Benefit Test 1		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Benefit - Dictionary	Benefit - Dictionary	0
Benefit - Dictionary	Benefit - Speaker 1	11651.879
Benefit - Dictionary	Benefit - Speaker 2	9990.8214
Benefit - Dictionary	Benefit - Speaker 3	5864.8432
Benefit Test 2		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Benefit - Dictionary	Benefit - Speaker 1	8582.0823
Benefit - Dictionary	Benefit - Speaker 2	9446.0804
Benefit - Dictionary	Benefit - Speaker 3	6185.5266

Figure 5.2: Results of word Benefit

Estimate Test 1		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Estimate - Dictionary	Estimate - Dictionary	0
Estimate - Dictionary	Estimate - Speaker 1	9959.2044
Estimate - Dictionary	Estimate - Speaker 2	10924.938
Estimate - Dictionary	Estimate - Speaker 3	5846.5519
Estimate Test 2		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Estimate - Dictionary	Estimate - Speaker 1	8380.9076
Estimate - Dictionary	Estimate - Speaker 2	9790.3322
Estimate - Dictionary	Estimate - Speaker 3	8638.7617

Figure 5.3: Results of word Estimate

Factor Test 1		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Factor - Dictionary	Factor - Dictionary	0
Factor - Dictionary	Factor - Speaker 1	11909.582
Factor - Dictionary	Factor - Speaker 2	7441.6264
Factor - Dictionary	Factor - Speaker 3	6187.427
Factor Test 2		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Factor - Dictionary	Factor - Speaker 1	6003.4525
Factor - Dictionary	Factor - Speaker 2	7169.6279
Factor - Dictionary	Factor - Speaker 3	4790.5106

Figure 5.4: Results of word Factor

Specific Test 1		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Specific - Dictionary	Specific - Dictionary	0
Specific - Dictionary	Specific - Speaker 1	9438.6985
Specific - Dictionary	Specific - Speaker 2	10562.806
Specific - Dictionary	Specific - Speaker 3	8087.3492
Specific Test 2		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Specific - Dictionary	Specific - Speaker 1	6980.2575
Specific - Dictionary	Specific - Speaker 2	10104.105
Specific - Dictionary	Specific - Speaker 3	7647.8445

Figure 5.5: Results of word Specific

Theory Test 1		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Theory - Dictionary	Theory - Dictionary	0
Theory - Dictionary	Theory - Speaker 1	9417.7441
Theory - Dictionary	Theory - Speaker 2	7947.4697
Theory - Dictionary	Theory - Speaker 3	5768.6094
Theory Test 2		
Audio 1	Audio 2	
Word - Source	Word - Source	DTW Score
Theory - Dictionary	Theory - Speaker 1	5505.9102
Theory - Dictionary	Theory - Speaker 2	7515.3149
Theory - Dictionary	Theory - Speaker 3	5144.2969

Figure 5.6: Results of word Theory

Validity Tests			
	Audio 1	Audio 2	
	Word - Source	Word - Source	DTW Score
Benefit	Benefit - Speaker 1	Estimate - Speaker 1	4907.01
	Benefit - Speaker 2	Estimate - Speaker 2	4469.76
	Benefit - Speaker 3	Estimate - Speaker 3	6670.05
	Benefit - Speaker 1	Benefit - Speaker 3	8139.84
Estimate	Estimate - Speaker 1	Factor - Speaker 1	6455.05
	Estimate - Speaker 2	Factor - Speaker 2	4045.69
	Estimate - Speaker 3	Factor - Speaker 3	8699.76
	Estimate - Speaker 1	Estimate - Speaker 3	7043.92
Factor	Factor - Speaker 1	Specific - Speaker 1	6467.49
	Factor - Speaker 2	Specific - Speaker 2	4734.62
	Factor - Speaker 3	Specific - Speaker 3	7409.2
	Factor - Speaker 1	Factor - Speaker 3	10039.3
Specific	Specific - Speaker 1	Theory - Speaker 1	6050.2
	Specific - Speaker 2	Theory - Speaker 2	4662.43
	Specific - Speaker 3	Theory - Speaker 3	6581.04
	Specific - Speaker 1	Specific - Speaker 3	9160.31
Theory	Theory - Speaker 1	Specific - Speaker 1	5072.15
	Theory - Speaker 2	Specific - Speaker 2	4969.02
	Theory - Speaker 3	Specific - Speaker 3	5605.38
	Theory - Speaker 1	Theory - Speaker 3	6507.94

Figure 5.7: Results of validity tests

3. The test needs to be done on a large number of people with different native languages.

4. The web application does not store the previous results. These could help the user to monitor their progress over the time, or to classify a speaker's recordings as proficient or not proficient in a certain accent or pronunciation.

5. More efficient implementations of the DTW could be employed for faster computation of the DTW score. Fast techniques for computing DTW include PrunedDTW [27], SparseDTW[28], FastDTW[29], and the MultiscaleDTW[30][31].

6. Dictionary recordings are quite clear, but there is no option to remove the noise from the user's recordings on web application.

5.1.2 Future work

Some of these limitations discussed can be easily overcome in future. More dictionary audio data can be collected and used in the app. This way a wide range of words can be covered.

Since the backend code is contained inside a docker image, it can be hosted on paid services (like AWS) which would allow it to be available to a large audience. More efficient implementations of DTW will help in a faster computation of the results.

In the future the application can be extended to mobile devices. It can also be used easily for other languages in order to help people learn new languages.

Chapter 6

Conclusion

In this project, the goal was to build a web application to effectively give the user feedback on how close their accent is from other accents. The feedback is in the form of a DTW score, a lesser score means the two accents are closer to each other and a higher score means the two accents are more separable.

So far the application only gives a DTW score. More work can be done in order to classify a speakers score as proficient, or less proficient, in the accent or pronunciation they are learning. Also, tests need to be done on the significant number of people with words ranging from simple to complex words for better and more useful insights.

Bibliography

[1] Meinard Müller and SpringerLink (Online service). *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer International Publishing, Cham, 2015 edition, 2015.

[2] Meinard Müller. *Information retrieval for music and motion*, pages 69–84. Springer, New York, 2007.

[3] AC Cargile. Evaluations of employment suitability: Does accent always matter? *Journal OF Employment Counselling*, 37(3):165–177, 2000.

[4] Zainab Thamer Ahmed, Ain Nadzimah Abdullah, and Chan Swee Heng. The role of accent and ethnicity in the professional and academic context. *International Journal of Applied Linguistics English Literature*, 2(5):249–258, 2013.

[5] Ellen B. Ryan and Cynthia M. Bulik. Evaluations of middle class and lower class speakers of standard american and german-accented english. *Journal of Language and Social Psychology*, 1(1):51–61, 1982.

[6] Michael Lutter. Mel-frequency cepstral coefficients, 2014. URL `http://recognize-speech.com/feature-extraction/mfcc`.

[7] PJ Hyett Tom Preston-Werner, Chris Wanstrath. Gitub - git repository hosting service, 2008. URL `https://en.wikipedia.org/wiki/GitHub`.

[8] Amazon. Amazon s3 - cloud storage, 2006. URL `https://en.wikipedia.org/wiki/Amazon_S3`.

[9] Roy Thomas Fielding. Architectural styles and the design of network-based software architectures. 2000.

[10] Solomon Hykes. Docker (software), 1999. URL https://en.wikipedia.org/wiki/Docker_(software).

[11] Google. Angular (application platform), 2016. URL https://en.wikipedia.org/wiki/Angular_(application_platform).

[12] Armin Ronacher. Welcome to flask, 2010. URL http://flask.pocoo.org/docs/0.12/.

[13] Librosa development team. Librosa, 2013. URL http://librosa.github.io/librosa/#librosa.

[14] Guy who sold gluster to red hat now running dotcloud, 2013.

[15] Mauz Khan. Recordrtc, 2014. URL http://recordrtc.org/.

[16] Wikipedia. Mel-frequency cepstrum, 2003. URL https://en.wikipedia.org/wiki/Mel-frequency_cepstrum.

[17] Wikipedia. Power spectrum, 2003. URL https://en.wikipedia.org/wiki/Spectral_density#Power_spectral_density.

[18] S. S. Stevens, J. Volkmann, and E. B. Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937. doi: 10.1121/1.1915893. URL https://doi.org/10.1121/1.1915893.

[19] Wikipedia. Cepstrum, 2011. URL https://en.wikipedia.org/wiki/Cepstrum.

[20] Jasmeet Singh. Ms project frontend. https://github.com/jasmeet17/ms_proj_frontend, 2018.

[21] Jasmeet Singh. Ms project backend. https://github.com/jasmeet17/ms_proj_server, 2018.

[22] Douglas O'Shaughnessy. Invited paper: Automatic speech recognition: History, methods and challenges. *Pattern Recognition*, 41(10):2965–2979, 2008.

[23] Paul Mermelstein. Distance measures for speech recognition – psychological and instrumental. *Haskins Laboratories Status Report on Speech Research*, 47(Jul): 91–103, 1976.

[24] Michael Lutter. Speech production, 2015. URL `http://recognize-speech.com/speech/speech-production`.

[25] Michael Lutter. Sense of hearing, 2014. URL `http://recognize-speech.com/speech/sense-of-hearing`.

[26] Averil Coxhead. A new academic word list. *TESOL Quarterly*, 34(2):213–238. URL `https://onlinelibrary.wiley.com/doi/abs/10.2307/3587951`.

[27] Diego F. Silva; Gustavo E. A. P. A. Batista;. *Speeding Up All-Pairwise Dynamic Time Warping Matrix Calculation*. Society for Industrial and Applied Mathematics, 2016.

[28] Ghazi Al-Naymat, Sanjay Chawla, and Javid Taheri. Sparsedtw: A novel approach to speed up dynamic time warping. *CoRR*, abs/1201.2969, 2012. URL `http://arxiv.org/abs/1201.2969`.

[29] Stan Salvador and Philip Chan. Fastdtw: Toward accurate dynamic time warping in linear time and space. In *KDD workshop on mining temporal and sequential data*. Citeseer, 2004.

[30] Meinard Müller, Henning Mattes, and Frank Kurth. An efficient multiscale approach to audio synchronization. In *International Society for Music Information Retrieval (ISMIR)*, 2006.

[31] Thomas Pratzlich, Jonathan Driedger, and Meinard Muller. Memory-restricted multiscale dynamic time warping. pages 569–573. IEEE, 2016.