PARLA: mobile application for English pronunciation
A supervised machine learning approach

Davide Berdin
{davide.berdin.0110}@student.uu.se

Master Thesis
Department of Information Technology

July 9, 2016

To my family that never stopped believing in me

Abstract

Learning and improving a second language is fundamental in the globalised world we live in. In particular, English is the common tongue used everyday by billions of people and the necessity of having good pronunciation in order to avoid misunderstanding is higher then ever. Smartphones and other mobile devices have rapidly become an everyday technology with endless potential given the large size of screens as well as the high portability. Old-fashioned language courses are very useful and important, however using the technology for picking up a new language in an automatic way with less time to dedicated to this process is still a challenge and an open research field. In this thesis, we describe a new method to improve the English language pronunciation of non-native speakers through the usage of a smartphone, using a machine learning approach. The aim is to provide the right tools for those users that want to quickly improve their English pronunciation without attending an actual course. The test has been conducted on users using the application for two weeks. The results show that the proposed approach is not particularly effective on people due the difficulty in understanding the feedback we delivered.

Keywords microlearning, second language pronunciation, mobile phone, visual feedback, supervised machine learning, non-native speakers

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Philip J. Guo for the continuous support during my time doing research, for his motivation and enthusiasm. He gave me the opportunity to experience research almost at Ph.D level as well as the opportunity to study in the U.S. I cannot thank enough for that.

Besides my supervisor, I would like to thank my reviewer, Olle Gällmo for his insightful comments and suggestions when writing this thesis.

My sincere thanks also go to the Department of Computer Science at University of Rochester for guesting me during my research time there. I had the opportunity to study in one of the top notch university in the world and to experience the American life-style in all its essence.

I want to thank two incredible students I had the opportunity to work with at U of R: Jeremy Warner and Leonard Brown. Their infinite patience in listening to all the issues I had during the development and their great support in providing suggestions and workarounds helped me to finish the project (not to mention all the stimulating discussions about the world!).

Also I thank my friends at Uppsala University: Francesca Martin, John Paton, Laurence Wainwright and all the people from the Computer Science department - I would like to list all of you guys but you are so many!

Last but not the least, I would like to thank my family: my parents Lorella and Dino and my brother Elia, for helping me to realize my dream of studying in America, encouraging me through all the hard times and their unwavering faith in my capabilities. I could never imagine to achieve what I did without knowing that their were always there for me. I will never forget it!

Contents

List of Figures

Chapter 1

Introduction

Pronunciation is the hardest part of learning a language among all the other components, such as grammar rules and vocabulary. To achieve a good level of pronunciation, non native speakers have to study and constantly practice the target language for an incredible number of hours. In most cases, when students are learning a new language, the teacher is not a native speaker, which implies that the pronunciation may be influenced by the country where he or she comes from, since it is a normal consequence of second learning language [6]. In fact, Medgyes and Peter (2001) state that the advantages of having a native speaker as a teacher lies in the superior linguistic competences, especially the usage of the language more spontaneously in different communication situations. Pronunciation falls into those competences underlying a base problem in teaching pronunciation at school.
The basic questions asked in this work are:

1) Why is pronunciation so important?
2) What are the most effective methods for improving the pronunciation?
3) What is the research state-of-art and how can it be improved?

The first question is fairly easy to answer. There are two reasons to claim why pronunciation is important: *(i)* it helps to acquire the target language faster and *(ii)* being understood. Regarding the first point, the earlier a learner masters the basics of pronunciation, the faster the learner will become fluent. The reason is because *critical listening* with a particular focus on hearing the sounds will lead to improved fluency in speaking the language. The second point is **crucial** when working with other people, especially as these days both in school and business the environment is often multicultural. Pronunciation mistakes may lead the person to being misunderstood affecting the results of a project for example.

With these statements in mind, Gilakjani et al. (2011) gives suggestions on how a learner can effectively improve the pronunciation. Four important ways are depicted: *Conversation* is the most relevant approach to improve pronunciation, although a supervision of an *expert guidance* that corrects the mistakes is fundamental during the process of learning. At the same time, learners have to be pro-active to have conversation with other native speakers in such a way to constantly practice. *Repetition* of pronunciation exercises is another important factor that will help the learner to be better in speaking. Lastly, *Critical listening*, which was mentioned earlier, amplifies the opportunity to learn how native speakers pronounce words. In particular, for a learner, it is important to understand the difference between how he or she is pronouncing a certain sentence and how it is pronounced by the native speaker. This method is very effective and is important for understanding the different sounds of the language and how a native speaker is able to reproduce them [7].

An important factor while learning a second language is to get feedback about improvements. Teachers are usually responsible for judging the learners' progress. In fact, when teaching pronunciation, one often draws the intonation and the stress of the words in such a way that the learner is able to see how the utterances should be pronounced. The *British Council* shows this practice [8]. The usage of visual feedbacks is the key to learning pronunciation and it is the main feature of this research.

In the computer science field, some work has previously been done regarding pronunciation. For instance, Edge et al. (2012) helps learners to acquire the tonal sound system of Mandarin Chinese through a mobile game. Another example is given by Head et al. (2014), in which the application provides a platform where learners of Chinese language can interact with native speakers and challenging them to a competition of pronunciations of Chinese tones.

The idea behind this project is based on the fact that people need to keep practicing their pronunciation to have a significant improvement, as well as needing immediate feedbacks to understand if they are going in the right direction or not. The approach we used is based on these two factors and we designed the system to be as useful and portable as possible. The mobile application is where the user will test the pronunciation; a server using a machine learning technique will compute the similarity between the user's pronunciation and the native speaker's one and the results will be displayed on the phone.

Data was collected from *American Native Speakers* by asking them to pronounce a set of most used idioms and slang. Each candidate had to repeat the same sentence several times trying to be as consistent as possible. After the data was gathered, a preprocessing step was needed since we are seeking specific features such as voice-stress, accent, intonation and formants. This part has been done using an external tool called **FAVE-Extract**[9] which uses **PRAAT**[10] to analyse the sound. At this point, the next step is processed differently when treating native speaker files because we manually define the correct *phonemes* for each sentence. This step is called **force alignment**, in which an estimate is made for the beginning and the end of when a phoneme is pronounced by the speaker. For non-native speakers we used the phonemes extracted using the speech recognition system.

The machine learning part is divided in two. The first consists of using the library called **CMU Sphinx 4**[11] with an acoustic model trained with all the data collected from the native speakers. This library is a **Hidden Markov Model-based**[12](HMM) system with multiple searching systems written in Java. To estimate the overall error between the native pronunciation and the user and, the performance of the speech recognition system, a method called **Word Error Rate** (WER) has been used. The second part consists of using a **Gaussian Mixture Model**[13] (GMM) that we used to predict the vowels pronounced by the user. The result should help the user to better understand *how close* his/her vowel pronunciation is compared with the native ones.

After the server has computed the speech recognition extracting the phonemes and predicted the similarity of vowels, the system creates graphs that are used in the mobile application as feedback. In this way the user has a clear understanding of how he/she should **adjust** the way the words should be pronounced.

Chapter 2

Sounds of General American English

In *General American English* there are 41 different sounds that can be structured by the way they are produced[14]. In Table 2.1 the kind of sounds with the respective number of possible productions is shown. Each type will be described in a dedicated section of this thesis. An important factor is the way *constriction* of the flow of air is made. In fact, to distinguish between *consonants*, *semivowels* and *vowels*, the *degree* of constriction is checked. Instead, for *sonorant* consonants the air flow is continuous with no pressure. *Nasal* consonants have an occlusive consonant made with a lowered velum, thus allowing the airflow in the nasal cavity. The *continuant* consonants are produced without blocking the airflow in the oral cavity.

Type	**Number**
Vowels	18
Fricatives	8
Stops	6
Nasals	3
Semivowels	4
Affricates	2
Aspirant	1

Table 2.1: Type of English sounds

2.1 Vowel production

Generally speaking, when a vowel is pronounced, there is no air-constriction in the flow. This means that the articulators, like the tongue, lips and uvula do not touch, allowing the flow of air from the lungs. The consonants instead have another pattern when producing them. Moreover, to produce each vowel, the mouth has to make a different shape in such a way that the resonance is different. Figure 2.1 shows the way the mouth, the jaw and the lips are combined in a such a way to produce the acoustic sound of a vowel.

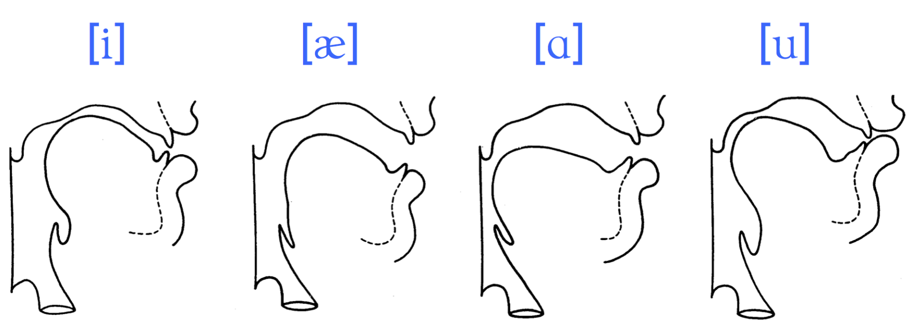


Figure 2.1: Vowels production [1]

2.1.1 Vowel of American English

There are 18 different vowels in American English that can be grouped by three different sets: the **monopthongs**, the **diphthongs**, and the **schwa's**, or reduced vowels.

/iʸ/	iy	beat	/ɔ/	ao	bought	/ɑʸ/	ay	bite
/ɪ/	ih	bit	/ʌ/	ah	but	/ɔʸ/	oy	Boyd
/eʸ/	ey	bait	/oʷ/	ow	boat	/ɑʷ/	aw	bout
/ɛ/	eh	bet	/ʊ/	uh	book	[ə]	ax	about
/æ/	ae	bat	/u/	uw	boot	[ɨ]	ix	roses
/ɑ/	aa	Bob	/ɝ/	er	Bert	[ɚ]	axr	butter

Figure 2.2: Example of words depending on the group [1]

The first column shows some examples of monopthongs. A *monopthong* is a clear vowel sound in which the utterance is fixed at both the beginning and at the end. The central part of the picture represents the diphthongs. A *diphthong* is the sound produced by two vowels when they occur within the same syllable. In the last column are depicted some examples of reduced vowels. *Schwa's* refers to the vowel sound that stays in the mid-central of the word. In general, in English, the schwa is found in an unstressed position.

2.1.2 Formants

A *formant* is the resonant frequency of a vocal track that resonate the loudest. In a spectrum graph, formants are represented by the peaks. In Figure 2.3 it is possible to see how the three first formants are defined by the peaks. The picture is of the *envelope*, a spectrogram of the vowel **[i]**. Frequencies are the most relevant information to determine which vowel has been pronounced. In general, within a spectrum graph there may be a different number of formants, although the most relevant are the first three and they are named **F1**, **F2** and **F3**.

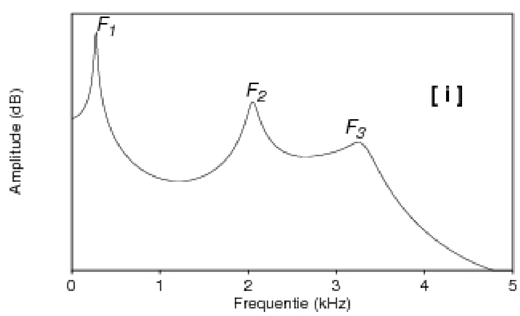


Figure 2.3: Spectral envelope of the [i] vowel pronunciation. F1, F2 and F3 are the first 3 formants [2]

The frequencies produced by the formants are highly dependent on the tongue position. In fact, formant *F1*'s frequencies are produced when the tongue is either in a *high* or *low* position, whereas formant *F2* when the tongue is in either *front* or *back* position and formant *F3* when the tongue is doing *Retroflexion*. **Retroflection** is more present when pronouncing the consonant *R*

2.1.3 Vowel duration

The duration of a vowel is the time that is taken when pronouncing it. Duration is measured in *centiseconds* and in English the different lengths are defined by certain rules. In general, the length of *lax vowels* such as /ɪ e æʌ ɒ u ə/ are short whereas *tense vowels* like /iː ɑː ɔː uː ɜː/ including diphthongs /eɪ aɪ ɔɪ əʊ aʊ ɪə ea ʊə/ have a variable length but longer than lax vowels [3]. Figure 2.4 is an example of time-length of some vowels. In General American English, the length of vowels are not as distinctive as in the *RP*[1] pronunciation. In some American accents, to express an emphasis the length of vowels can be extended.

[1]More commonly referred as the Standard English in the UK

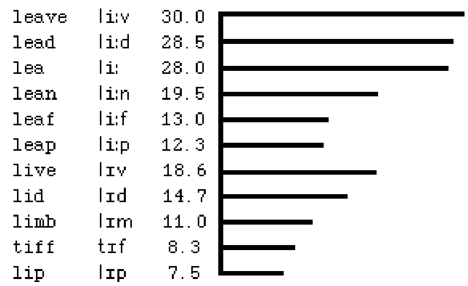


Figure 2.4: RP vowel length [3]

2.2 Fricative Production

A **fricative** is a consonant sound that is produced by narrowing the cavity causing a friction as the air goes through it [15]. There are eight fricatives in American English divided into two categories: *Unvoiced* and *Voiced*. These two categories are often called *Non-Strident* and *Strident* which means that there is a constriction behind the alveolar ridge.

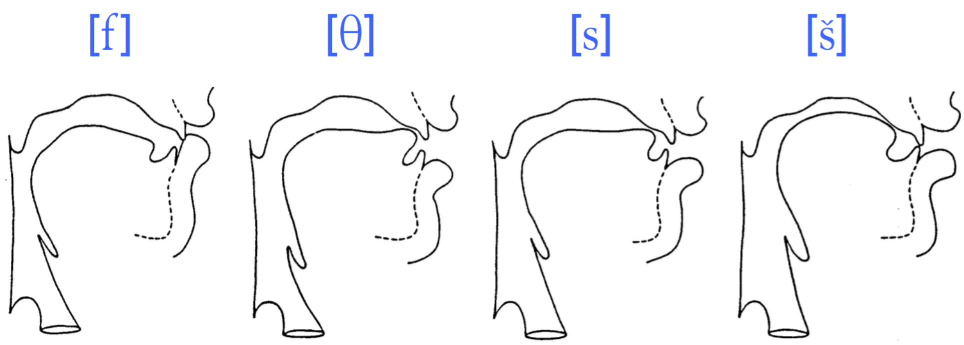


Figure 2.5: Fricative production [1]

In Figure 2.6 it is possible to see some examples of these two categories. Each consonant also belongs to a specific articulation position. In fact, each figure in 2.5 represents a specific articulation position. From left to right there is: *Labio-Dental* (Labial), *Interdental* (Dental), *Alveolar* and *Palato-Alveolar* (Palatal).

Type	Unvoiced			Voiced		
Labial	/f/	f	fee	/v/	v	v
Dental	/θ/	th	thief	/ð/	dh	thee
Alveolar	/s/	s	see	/z/	z	z
Palatal	/š/	sh	she	/ž/	zh	Gigi

Figure 2.6: Fricative examples of productions [1]

2.3 Affricate Production

An **affricate** consonant is produced by stopping the airflow first and then release it similarly to a fricative. The result is also considered a *turbulence noise* since the produced sound has a sudden release of the constriction. In English there only two affricate phonemes, as depicted in 2.7.

Voiced			Unvoiced		
/ǰ/	jh	judge	/č/	ch	church

Figure 2.7: Affricative production [1]

2.4 Aspirant Production

An **aspirant** consonant is a strong outbreak of breath produced by generating a turbulent airflow at glottis level. In American English there exists only one aspirant consonant and it is the /h/, for instance in the word *hat*.

2.5 Stop Production

A **stop** is a consonant sound formed by stopping the airflow in the oral cavity. The stop consonant is also known as *plosive*, which means that when the air is released it creates a small *explosive* sound [16]. The occlusion can come up in three different variance as shown in Figure 2.8: from left to right there is a *Labial* occlusion, the *Alveolar* occlusion and the *Velar* occlusion. The pressure built up in the vocal tract, determine the produced sound depending on which occlusion is performed.

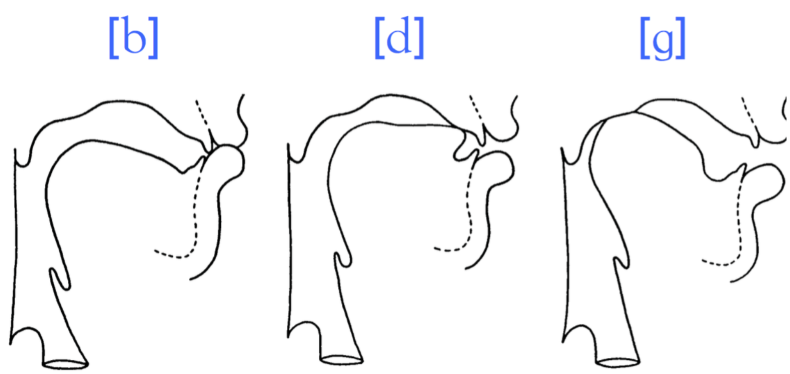


Figure 2.8: Stop production [1]

In American English there are six stop consonants, as represented in 2.9. As for the fricative consonants, the two main categories are the *Voiced* and *Unvoiced* sounds. Although, a particularity of the Unvoiced stops is that they are typically *aspirated* whereas in the Voiced ones there is a *voice-bar* during the closure movement. These two particularities are very useful where analyzing the formants because the frequencies are very well distinguished allowing a classification system to better understand the difference between stop phonemes.

Type	Voiced			Unvoiced		
Labial	/b/	b	bought	/p/	p	pot
Alveolar	/d/	d	dot	/t/	t	tot
Velar	/g/	g	got	/k/	k	cot

Figure 2.9: Stop examples of production [1]

2.6 Nasal Production

A **nasal** is an occlusive consonant sound that is produced by the lowering of the soft palate *(lowered velum)* at the back of the mouth, allowing the airflow to go out through the nostrils [17]. Because the airflow escapes through the nose, the consonants are produced with a closure in the vocal tract. Figure 2.11 shows the three different positions to produce a nasal consonant. From left to right the positions are *Labial*, *Alveolar* and *Velar*.

Due to this particularity, the frequencies of nasal *murmurs* are quite similar. Examining the spectrogram in Figure 2.10, it is possible to notice that nasal consonants have a high similarity. In a classification system, this can be a problem.

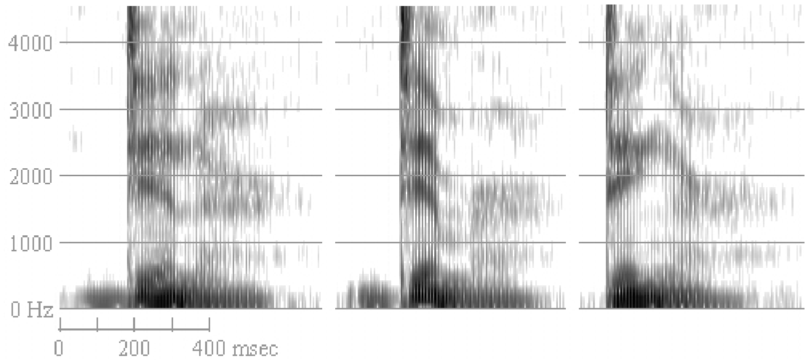


Figure 2.10: Nasal Spectrograms[2] of **dinner**, **dimmer**, **dinger**

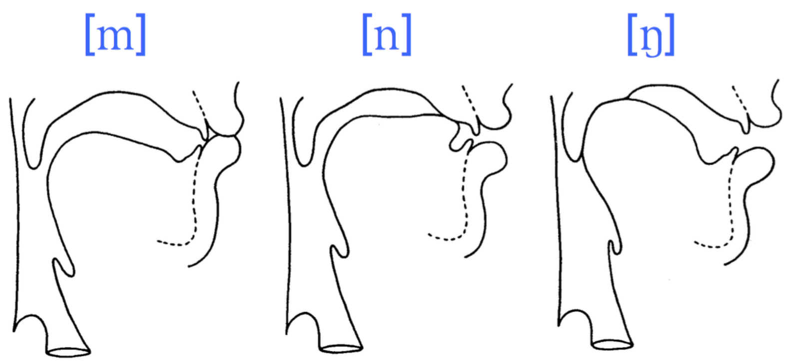


Figure 2.11: Nasal production [1]

Since the sound produced by a nasal is produced with an occlusive vocal tract, each consonant is **always attached** to a vowel and it can form an entire syllable. Although, in English, the consonant /**ŋ**/ always occur immediately after a vowel. In Figure 2.12 are shown some examples of nasal consonants divided by articulation position.

Type	Nasal		
Labial	/m/	m	me
Alveolar	/n/	n	knee
Velar	/ŋ/	ng	sing

Figure 2.12: Nasal examples of production [1]

2.7 Semivowel Production

A **semivowel** is a sound that is very close to a vowel sound but it works more likely as a syllable boundary rather than a core of a syllable [18]. A typical example of semivowels in English are the **y** and **w** in words *yes* and *west*. In the *IPA* alphabet they are written /j/ and /w/ and they correspond to the vowels /iː/ and /uː/ in the words *seen* and *moon*. In Figure 2.14 there are some examples of semivowels production.
The sound is produced by making a constriction in the oral cavity without having any sort of air turbulence. To achieve that, the articulation motion is slower than other consonants because the laterals[3] form a complete closer combined with a tongue tip. In this way the airflow has to pour out using the sides of the constriction.

[3]They are a pair of upper teeth that are located laterally from the central incisors [19]

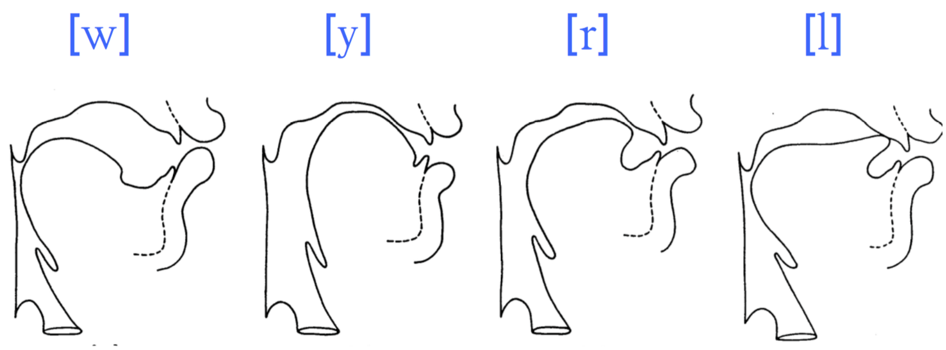


Figure 2.13: Semivowel production [1]

In American English there are four semivowels and they are depicted in Figure 2.13. An important fact about semivowels is that they are always close to a vowel. Although, the /l/ can form an entire syllable by itself when there is no stress in a word.

Type	Semivowel			Nearest Vowel
Glides	/w/	w	wet	/u/
	/y/	y	yet	/i/
Liquids	/r/	r	red	/ɝ/
	/l/	l	let	/o/

Figure 2.14: Semivowel examples of production [1]

Acoustic Properties of Semivowels

Semivowels have some properties that are taken into account when doing any sort of analysis. In fact, /w/ and /l/ are the semivowels that are more confusable because both are characterized by a *low* range of frequencies for both formants *F1* and *F2*. Although, the /w/ can be distinguished by the *rapid falloff* in the F2 spectrogram whereas /l/ has more often a *high frequency energy* compared to /w/. The **energy** is the relationship between the *wavelength* and the *frequency*. So, having a high energy means that there is a high frequency value and a small wavelength [20]. The semivowel /y/ is characterized by having a very low frequency value in formant F1 and a very high in formant F2. The /r/ instead is presented with a very low frequency value of formant F3.

2.8 The Syllable

The definition of the **syllable** can be divided in two sub-definitions: one from the phonetic point of view and one from the phonological point of view.
In phonetic analysis, syllables are basic units of speech which *"are usually described as consisting of a centre which has little or no obstruction to airflow and which sounds comparatively loud; before and after that centre (...) there will be greater obstruction to airflow and/or less loud sound"* [21]. Taking the word *cat* (/kæt/) as example, the **centre** is defined by the vowel /æ/ in which takes place only a little obstruction. The surrounding *plosive* consonants (/k/ and /t/) the airflow is completely blocked [22].
A phonological definition of the syllable establishes that it is *"a complex unit made up of nuclear and marginal elements"*[23]. In this context, the vowels are considered the **Nuclear** elements, or syllabic segments, whereas the **Marginal** ones are the consonants, or non-syllabic segments [22]. Considering the word *paint* (/peɪnt/) for example, the nuclear element is defined by the diphthong /eɪ/ whereas /p/ and /nt/ are the marginal elements.

2.8.1 Syllable Structure

In the phonological theory, the syllable can be decomposed in a hierarchical structure instead of a linear one. The structure starts with the σ letter which represents not only the root, but the syllable itself. Immediately after, there are two *branches* called **constituents** that represent the *Onset* and the *Rhyme*. The left branch includes any

consonants that precede the vowel (or Nuclear element), whereas the right branch includes both the nuclear element and any consonants (or Marginal elements) that potentially could follow it.
Usually, the rhyme branch is further split into two other branches represented by the **Nucleus** and the **Coda**. The first one represents the nuclear element in the syllable. The second one instead, subsumes all the consonants that follow the Nucleus in the syllable [22]. In Figure 2.15 there is a representation of the syllable structure based on the word *plant*.

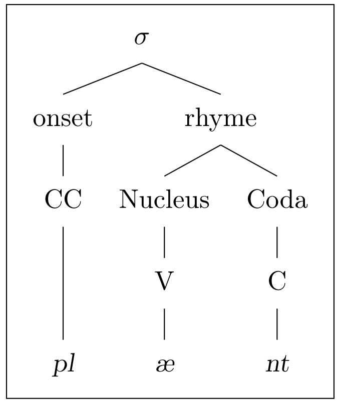


Figure 2.15: Tree structure of the word **plant**[4]

2.8.2 Stress

In the areas of linguistic studies and speech recognition, the stress is the emphasis that a person puts in a specific part of a word or sentence. Typically, the stress part of a word/sentence is detected by paying attention to the sudden change of pitch or increased loudness.
Figure 2.16 is an example in which more emphases is given when pronouncing that particular sentence. The big black dots represent such emphasis.

Figure 2.16: Example of stress representation[5]

[4]**C** means *Consonant* whereas **V** means *Vowel*
[5]http://linguistics.stackexchange.com/questions/2420/what-is-the-difference-between-syllable-timing-and-stress-timing

Chapter 3

Acoustics and Digital Signal Processing

In the past decade, digital computers have significantly helped *signal processing* to quantify a finite number of bits. The flexibility inherited from digital elements allows the usage of a vast number of techniques in which had not been possible to implement in the past. Nowadays, digital signal processors are used to perform multiple operations, such as *filtering*, *spectrum estimation* and many other algorithms [24].

3.1 Speech signals

The **speech** is the human way of communication. The protocol used in communication is based on a syntactic combination of different words taken from a very large vocabulary. Each word in the vocabulary is composed of a small set of vowels and consonants that combined with a phonetic unit forms a spoken word.
When a word is pronounced[1], a sound is produced causing the air particles to be excited at a certain vibration rate. The source of our voice is due to the vibration of the vocal cords. The resultant signal is *non-stationary* but it can be divided in segments since each phoneme has a common acoustic properties. In Figure 3.1 it is possible to notice how the pronounced words have a different shape as well as when the intensity of the voice is higher/lower during the pronunciation.

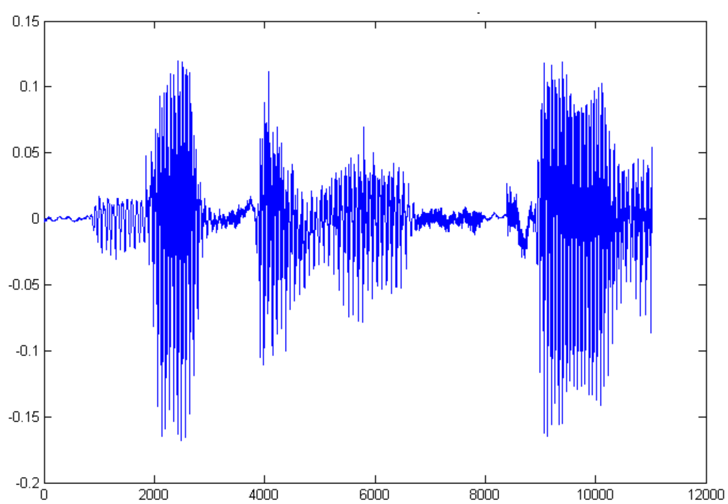


Figure 3.1: Example of a speech sound. In this case, the sentence **This is a story** has been pronounced [4]

The simplest form of sound is the *sinusoid* and it is the easiest waveform to describe because it corresponds to a **pure tone**. A pure tone consist in a waveform that consists only on one frequency.

3.1.1 Properties of Sinusoids

A sinusoid is a simple waveform represented by an up and down movement. There are three important measures that must be taken into consideration when defining the shape of the sinusoid: *amplitude*, *frequency* and *phase*.

[1]Chapter 2 explains in details how phonemes are pronounced

Amplitude

The amplitude, from a sound point of view, corresponds to the *loudness* whereas in the soundwave it corresponds to the amount of **energy**. In general the amplitude is measured in units called **deciBels** (dB), which are a logarithmic scale relative to a standard sound[2].

Frequency

Frequency is the number of cycles per unit of time[3]. To define a cycle, one can think of an oscillation that starts from the middle line, goes to the maximum point, down to the minimum and get back to the middle point. The unit of measure of the frequency is calculated in **Hertz** (Hz). Also, by calculating the time taken for one cycle, one estimates the so called **period**.
Frequency plays a fundamental role with the *pitch*. In fact, changing the number of oscillations but keeping the same waveform, causes an increase or decrease the level of the pitch.

Phase

The **phase** measures the starting point position of the waveform. If the sinusoids start at the very minimum of the wave, the value of the phase is π radians whereas starting from the top of the wave it will have a phase of *zero*. When two sounds do not have the same phase, it is possible to perceive the difference in the time scale since one of the two is delayed compared to the other. When comparing two signals, there is the need to obtain a *"phase-neutral"*, that means the comparison is made taking only Amplitude and Frequency into account. This method is called **autocorrelation** of the signals.

3.1.2 Spectrograms

A **spectrogram** is the visual representation of an acoustic signal[4]. Basically, a Fourier Transformation (Section 3.2) is applied to the sound, in such a way to obtain the set of waveforms extracted form the original signal and separate their frequencies and amplitudes. The result is typically depicted in a graph with degrees of amplitude with a *light-dark* representation. Since amplitude represents the *energy*, having a darker shade means that the energy is more intense in a certain range of frequencies - lighter when there is low energy. In Figure 2.10 there is an example of the spectrogram.
The visual feedback of the spectrogram is highly dependent from the **window size** of the Fourier Analysis. In fact, different sizes affect the levels of frequencies and time resolution.
If the window size is *short*, the adjacent **harmonics** are distorted but the time resolution is better [**?**]. An harmonic is an integer multiple of the fundamental frequency or component frequencies. This is helpful when looking for the *formant structure* because the striations created by the spectrogram highlights the individual pitch periods.
On the other hand, a *wider* window size, helps to locate the harmonics because the band of the spectrogram are narrower.

3.2 Fourier Analysis

Fourier Analysis is the process of decomposing a periodic waveform into a set of sinusoids having different amplitudes, phases and frequencies. Adding those waveforms again will yield the original signal. The analysis has been involved in many scientific applications and the reason is due to the following transform properties:

- Linear transformation - the relationship between two modules is kept
- Exponential function are eigenfunctions of differentiation [25]
- Invertible - derived from the linear relationship

In signal processing, Fourier analysis is used to isolate singular components of a complex waveform. A set of techniques consist of using the **Fourier Transformation** on a signal in such a way as to be able to manipulate the data in the easiest way possible, but at the same time maintaining invertibility of the transformation [26]. The next subsections describe the fundamental steps for manipulating a signal.

[2] http://web.science.mq.edu.au/~cassidy/comp449/html/ch03s02.html
[3] In general, a unit of time is considered a single second
[4] https://home.cc.umanitoba.ca/~robh/howto.html

3.2.1 Sampling

Sampling is the process in which a continuous signal is periodically measured every **T** seconds [24].
Consider a sound signal that varies in time as a continuous function $s(t)$. Every T seconds, the value of the function is measured. This frame of time is called the *sampling interval* [27]. To calculate the sequence a sampled function is given as follow: $s(nT), \forall$ integer values of n. Thus, the *sampling rate* is the average number of samples obtained in a range of $T = 1sec$. An example of sampling is shown in 3.2.

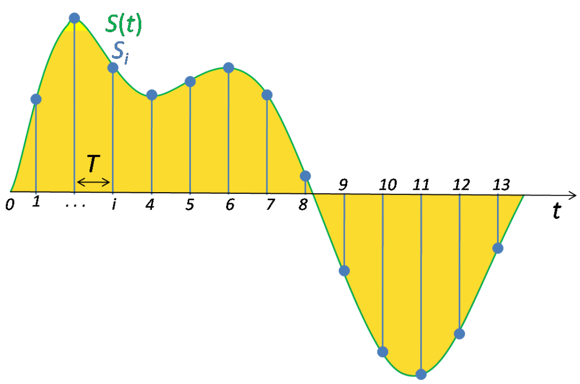


Figure 3.2: Example of signal sampling. The green line represents the continuous signal whereas the samples are represented by the blue lines[5]

As previously mentioned, using Fourier Analysis it is desirable to be able to reconstruct the original signal from the transformed one. To allow this, the **Nyquist-Shannon** theorem states that the sampling rate has to be larger than twice the maximum frequency of the signal, in order to rebuild the original signal [28].
The *Nyquist sampling rate* is defined by the following equation:

$$f_s > f_{Nyquist} = 2f_{max} \tag{3.1}$$

3.2.2 Quantization

To finalize the transformation from a continuous signal to a discrete one, the signal must be *quantized* in such a way as to obtain a finite set of values. Unlike sampling, in which permits to reconstruct the original signal, quantization is an irreversible operation that introduces a loss of information.
Consider x be the sampled signal and x_q the quantized one where x_q can be expressed as the signal x plus the error e_q. Then:

$$x_q = x + e_q \Leftrightarrow e_q = x - x_q \tag{3.2}$$

Given the equation above, the range of error can be restricted to $-q/2... + q/2$ because no error will be larger than the half of the quantization step. From a mathematical point of view, the error-signal is a random signal with an uniform probability distribution between the range of $q/2$ and $+q/2$, giving the following [29]:

$$p(e) = \begin{cases} \frac{1}{q} & \text{for } \frac{-q}{2} \leq e < \frac{q}{2} \\ 0 & \text{otherwise} \end{cases} \Bigg\} \tag{3.3}$$

This is why the quantization error also called quantization noise.

3.2.3 Windowing Signals

Speech sound is a **non-stationary** signal where its properties (amplitude, frequency and pitch) rapidly change over time [30]. Due to the quick changes of those properties, it makes it hard to use *autocorrelation* or the *Discrete Fourier Transformation*. Chapter 2 highlighted the fact that phonemes have some invariant properties for a small period of time. Having said that, it is possible to apply methods that will take *short windows* (pieces of signal) and process them. This window is also called a **frame**. Typically, the shape of this window is *rectangular* because one of the most used methods are the *Hanning* and *Hamming* in which the window covers the whole amplitude spectrum between a range. In Figure 3.3 there is an example on how the Hamming window is taken from a signal. The rectangle called *Time Record*, is the frame that is extracted and processed by the windowing function.

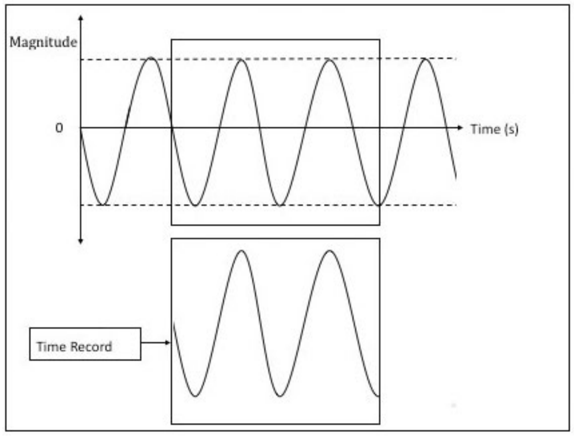


Figure 3.3: Hamming window example on a sinusoid signal

3.2.4 Hann Function

This is one of the most used windowing method in signal processing. The function is discrete and it is defined by equation:

$$w(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{3.4}$$

The method is a linear combination of the *rectangular function* defined by the following:

$$w_r = \mathbf{1}_{[0,N-1]} \tag{3.5}$$

Starting from *Euler's formula*, it is possible to inject the rectangular equation as shown below:

$$w(n) = \frac{1}{2}\,w_r(n) - \frac{1}{4}e^{\mathrm{i}2\pi\frac{n}{N-1}}w_r(n) - \frac{1}{4}e^{-\mathrm{i}2\pi\frac{n}{N-1}}w_r(n) \tag{3.6}$$

From here, given the properties of the *Fourier Transformation*, the spectrum of the window function is defined as follows:

$$\hat{w}(\omega) = \frac{1}{2}\hat{w}_r(\omega) - \frac{1}{4}\hat{w}_r\left(\omega + \frac{2\pi}{N-1}\right) - \frac{1}{4}\hat{w}_r\left(\omega - \frac{2\pi}{N-1}\right) \tag{3.7}$$

Combining the spectrum with equation 3.5 yields the below equation in which the signal modulation factor *disappears* when the windows are moved around time 0:

$$\hat{w}_r(\omega) = e^{-\mathrm{i}\omega\frac{N-1}{2}}\frac{\sin(N\omega/2)}{\sin(\omega/2)} \tag{3.8}$$

The reason why this windowing method is one of the most diffuse is due to the *low aliasing*

3.2.5 Zero Crossing Rate

Zero crossing is the point of the function where the sign changes from a positive value to a negative one or vice versa. The method of counting the zero crossings is widely used in speech recognition for estimating the *fundamental* frequency of the signal. The zero-crossing rate is the rate of this positive-negative changes. Formally, it is defined as follows:

$$ZCR = \frac{1}{T-1}\sum_{t=1}^{T-1}\left\{\begin{matrix}1 & s_t s_{t-1} < 0\\ 0 & \text{otherwise}\end{matrix}\right\} \tag{3.9}$$

where s is the signal of length T.

3.2.6 The Discrete Fourier Transform

Before jumping into the definition of the Discrete Fourier Transformation (DFT),the Fourier Transformation (FT) must first be introduced from the mathematical point of view. The FT of a continuous-signal $x(t)$ is defined by the following equation:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt, \qquad \omega \in (-\infty, \infty) \tag{3.10}$$

The discrete operation allows us to transform the equation above from an infinite space in a finite sum as follows:

$$X(\omega_k) = \sum_{n=0}^{N-1} x(t_n)e^{-j\omega_k t_n}, \qquad k = 0, 1, 2, \ldots, N-1 \tag{3.11}$$

where $x(t_n)$ is the *amplitude* of the signal at time t_n (sampling time). T is the sampling period in which the transformation is applied. $X(\omega_k)$ is the *spectrum* of the complex value x at frequency ω_k. Ω is the sampling interval defined by the *Nyquist-Shannon* theorem whereas N is the number of samples.

The motivation behind the DFT is to move the signal from the *Time or space domain* to the *Frequency domain*. This allows us to analyse the spectrum in a simpler way. 3.4 shows the transformation.

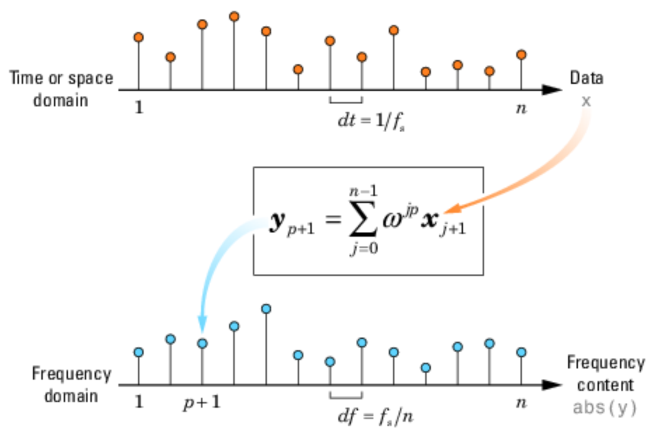


Figure 3.4: DFT transformation[6]

Chapter 4

Speech Recognition

Speech recognition is an application of machine learning which allows a computer program to extract and recognize words or sentences from a human's language and converting them back to a machine language. Google Voice Search[1] and Siri[2] are two examples of speech recognition software with the capability of understanding natural language.

4.1 The Problem of Speech Recognition

Human languages are very complex and different among each other. Despite the fact that they might have a well-structured grammar, automatic recognition is still a very difficult problem, since people have many ways to say the same thing. In fact, spoken language is different from the written one because the articulation of verbal utterance is less strict and complicated.
The environment in which the sound is taken has a big influence on the speech recognition software because it introduces an *unwanted* amount of information in the signal. For this reason, it is important that the system is capable of *identifying* and *filtering out* this surplus of information [31].

Another interesting set of problems are related to the speaker itself. Each person has a different body which means there are a variety of components that the recognition system has to take care of in such a way to be able to understand correctly. Gender, vocal tracts, speaking style, speed of the speech, regional provenience are fundamental parts that have to be taken into consideration when building the *acoustic model* for the system. Despite these features being unique for each person, there some common aspects that will be used to construct the model. The acoustic model represents the relationship between the acoustic signal of the speech and the phonemes related to it.

Ambiguity presents the major concern since natural languages have inherited it. In fact, it may so happen that in a sentence, we are not able to discriminate which words are actually intended [31]. In speech recognition there are two types of ambiguity: *homophones* and *word boundary ambiguity*.
Homophones are those words that are spelled in a different way but they **sound** the same. Generally speaking, these words are not correlated to each other but it happens that the sound is equivalent. On the other hand, word boundary ambiguity occurs when there are multiple ways of grouping phones into words[31]. For example, *peace* and *piece*, *idle* and *idol*, are two examples of homophones.

4.2 Architecture

Generally speaking, a speech recognition system is divided in three main components: the **Feature Extraction** (or Front End), the **Decoder** and the **Knowledge Base** (KB). In Figure 4.1 the KB part is represented by the three sub-blocks called *Acoustic Model*, *Pronunciation Dictionary* and *Language Model*. The *Front End* takes as input the voice signal where it is analysed and converted in the so called *Features Vectors*. This last is the set of common properties that we discussed in chapter 2. From here we can say that $\mathbf{Y}1 : N = y_1, ..., y_N$ where Y is the set of features vectors.
The second step consists in feeding the *Decoder* with vectors we obtained from the previous step, attempting to find

[1] https://www.google.com/search/about/
[2] http://www.apple.com/ios/siri/

the sequence of words $\mathbf{w}1 : L = w_1, ..., w_L$ that have most likely generated the set Y[5]. The decoder tries to find the likelihood estimation as follows:

$$\widehat{w} = \underset{w}{arg\,max}\; P(\mathbf{w}|\mathbf{Y}) \tag{4.1}$$

The $P(w|Y)$ is difficult to find directly[3], but using Bayes' Rules we can transform the equation above in

$$\widehat{w} = \underset{w}{arg\,max}\; P(\mathbf{Y}|\mathbf{w})P(\mathbf{w}) \tag{4.2}$$

in which the probability $P(Y|w)$ and $P(w)$ are estimated by the *Knowledge Base* block. In particular, the *Acoustic Model* is responsible to estimate the first one whereas, the *Language Model* estimates the second one.
Each word $\mathbf{w}$ is decomposed in smaller components called *phones*, representing the collection of phonemes $\mathbf{K}_w$ (see chapter 2). The *pronunciation* can be described as $\mathbf{q}^{(w)}_{1:K_w} = q_1, ..., q_{K_w}$. The likelihood estimation of the sequence of phonemes is calculated by a **Hidden Markov Model** (HMM). In the section, a general overview of HMM is given. A particular model will not be discussed here because every speech recognition system uses a variation of the general HMM chain.

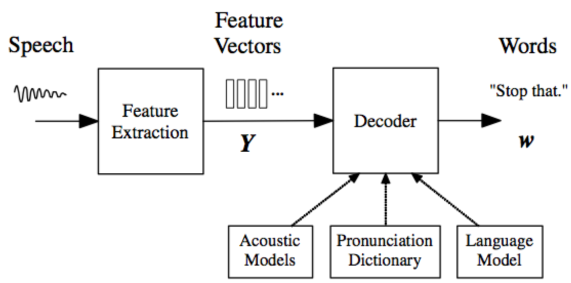


Figure 4.1: HMM-Based speech recognition system [5]

4.3 Hidden Markov Model

"An Hidden Markov Model is a finite model that describes the probability distribution over an infinite number of possible sequences"[12]. Each sequence is determined by a set of *transition probabilities* which describes the transitions among states. The **observation** (or outcome) of each state is generated based on the associated probability distribution. From an *outside* perspective, the *observer* is only able to see the outcome and not the state itself. Hence, the states are considered **hidden** which leads to the name Hidden Markov Model[4][33].

An HMM is composed of the following elements:

- The number of states (N)
- The number of observations (M), that becomes infinite if the set of observations is continuous
- The set of transition probabilities, $\Lambda = \{a_{ij}\}$

  The set of probabilities is defined as follows:

$$a_{ij} = p\{\, q_{t+1} = j \,|\, q_t = i \,\},\;\; 1 \leq i, j \leq N, \tag{4.3}$$

where q_t is the state we are currently in and a_{ij} represent the transition from state i to j. Each transition should satisfy the following rules:

[3]There is discriminate way of finding the estimation directly as described in [32]
[4]http://jedlik.phy.bme.hu/~gerjanos/HMM/node4.html

$$a_{ij} \leq 1, \;\; 1 \leq i, j \leq N, \tag{4.4a}$$

$$\sum_{j=1}^{N} a_{ij} = 1, \;\; 1 \leq j \leq N \tag{4.4b}$$

For each state S we can define the probability distribution $S = \{s_j(k)\}$ as follows:

$$s_j(k) = p\,\{\, o_t = v_k \,|\, q_t = j \,\}, \;\; 1 \leq j \leq N, \; 1 \leq k \leq M \tag{4.5}$$

where v_k is the k^{th} observation whereas o_t is the outcome. Furthermore, $b_j(k)$ must satisfy the same stochastic rules described in equation 4.4.

A different approach is made when the number of observations is infinite. In fact, we are not going to use a set of discrete probabilities but instead a continuous probability density function. Given that, we can define the parameters of the density function by approximating it by a weighted sum of M Gaussian distributions φ [34]. We can describe the function as follows:

$$s_j(o) = \sum_{m=1}^{M} c_{jm} \varphi(\mu_{jm}, \Sigma_{jm}, o_t) \tag{4.6}$$

where c_{jm} is the weighted coefficients, μ_{jm} is the mean vector and Σ_{jm} is the covariance matrix. The coefficients should satisfy the stochastic rules in equation 4.4.
We can then define the initial state distribution as $\pi = \{\pi_i\}$ where

$$\pi_i = p\,\{q_I = i\,\}, \;\;\; 1 \leq i \leq N \tag{4.7}$$

Hence, to describe the HMM with the discrete probability function we can use the following compact form

$$\lambda = (\Lambda, S, \pi) \tag{4.8}$$

whereas to denote the model with a continuous density function, we use the one described in equation 4.9

$$\lambda = (\Lambda, c_{jm}, \mu_{jm}, \Sigma_{jm}, \pi) \tag{4.9}$$

4.3.1 Assumptions

The theory behind HMM requires three important assumptions: the **Markov assumption**, the **stationarity assumption** and the **output independence assumption**.

The Markov Assumption

The Markov assumption assumes that the following state depends only from the state we are currently in, as given in equation 4.3. The result model is also referred as *first order* HMM. Generally speaking though, the decision of the next coming state might depend on **n** previous states, leading to a n^{th} HMM order model. In this case, the transition probabilities is defined as follows:

$$a_{i_1 i_2 \ldots i_n j} = p\,\{\, q_{t+1} = j \,|\, q_t = i_1, \, q_{t-1} = i_2, ..., \, q_{t-k+1} = i_k \,\}, \;\;\; 1 \leq i_1, i_2, ..., i_k, j \leq N \tag{4.10}$$

The Stationary Assumption

The second assumption states that the transition probabilities are *time-independent* when the transitions occur. This is defined by the following equation for any t_1 and t_2:

$$p\,\{\, q_{t+1} = j \,|\, q_{t_1} = i \,\} \; = \; p\,\{\, q_{t_2+1} = j \,|\, q_{t_2} = i \,\} \tag{4.11}$$

The Output Assumption

The last assumption says that the current observation is statistically independent from the previous observations. Let's consider the following observations:

$$O = o_1, o_2, ..., o_T \tag{4.12}$$

Now, recalling equation 4.8, it is possible to formulate the assumption as follows:

$$p\{\,O\,|\,q_1, q_2, ..., q_T, \lambda\,\} \;=\; \prod_{t=1}^{T} p\{\,o_t\,|\,q_t,\,\lambda\,\} \tag{4.13}$$

4.4 Evaluation

The next step in the HMM algorithm is the *evaluation*. This phase consists in estimating the likelihood probability of a model when it produces that output sequence. Generally speaking, there are two famous algorithms that have been extensively used: **forward** and **backward** probability algorithms. In the next two subsections, we describe these two algorithms, either one of which may be used.

4.4.1 Forward probability algorithm

Let us consider the equation 4.13 where the probabilistic output estimation is given. The major drawback of this equation is that the computational cost is exponential in T because the probability of O is calculated directly. It is possible to improve the previous approach by *caching* the calculations. The cache is made using a *lattice* (or trellis) of states where at each time step, the α value is calculated by summing all the states at the previous time step [35].

The α value (or forward probability) can be calculated as follows:

$$\alpha_t(i) \;=\; P(o_1, o_2, ..., o_t, q_t = s_i|\lambda) \tag{4.14}$$

where s_i is the state at time t.
Given that, we can define the forward algorithm in three steps as follows:

1. Initialization:

$$\alpha_1(i) = \pi_i b_i(o_1),\; 1 \leq i \leq N \tag{4.15}$$

2. Induction step:

$$\left(\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right) b_j(o_{t+1}) \text{ where } 1 \leq t \leq T-1,\; 1 \leq j\; N \tag{4.16}$$

3. Termination:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{4.17}$$

The key of this algorithm is equation 4.16, where for each state s_j the α value contains the probability of the observed sequence from the beginning to time t. The direct algorithm has a complexity of $2TN^T$ whereas the new one is N^2T.

4.4.2 Backward probability algorithm

This algorithm is very similar to the previous one with the only difference when calculating the probability. Instead of estimating the probability as in equation 4.14, the backward algorithm estimates the likelihood of *"the partial observation sequence from $t+1$ to T, starting from state s_i"*[5].

The probability is calculated with the following equation:

[5] http://digital.cs.usu.edu/~cyan/CS7960/hmm-tutorial.pdf

$$\beta_t(i) \;=\; P(o_{t+1}, o_{t+2}, ..., o_T \,|\, q_t = s_i,\, \lambda) \tag{4.18}$$

The usage of either one depends on the type of problem we need to face.

4.5 Viterbi algorithm

The main goal of this algorithm is to discover the sequence of hidden states that are more likely to be produced given a sequence of observations. This block is called **decoder** (see Figure 4.1 for reference). The *Viterbi algorithm* is one of the most used solution for finding a *single best sequence* for a given set of observations. What makes this algorithm suitable for this problem, is the similarity between the forwarding algorithm with the only difference that, instead of summing the transition probabilities at each step, it calculates the **maximum**. In Figure 4.2 it is shown how the maximization estimation is calculated during the recursion step.

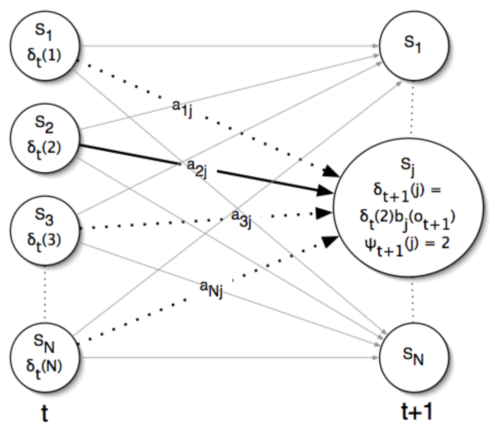


Figure 4.2: The recursion step

Let's define the probability of the most likely sequence for a given partial observation:

$$\delta_t(i) = \max_{q_1,q_2,...,q_{t-1}} P(q_1, q_2, ..., q_t = s_i, o_1, o_2, ..., o_t \,|\, \lambda) \tag{4.19}$$

Using this, the steps of the are algorithm as follows:

1. Initialization:

$$\delta_1(i) = \pi_i b_i(o_1),\; 1 \leq i \leq N, \phi_1(i) = 0 \tag{4.20}$$

2. Recursion:

$$\delta_t(j) = \max_{1\leq i\leq N} [\delta_{t-1}(i)a_{ij}]\, b_j(o_t),\; 2 \leq t \leq T,\; 1 \leq j \leq N, \tag{4.21a}$$

$$\psi_t(j) = \arg\max_{1\leq i\leq N} [\delta_{t-1}(i)a_{ij}],\; 2 \leq t \leq T,\; 1 \leq j \leq N, \tag{4.21b}$$

3. Termination:

$$P^* = \max_{1\leq i\leq N} [\delta_T(i)] \tag{4.22a}$$

$$q_t^* = \arg\max_{1\leq i\leq N} [\delta_T(i)] \tag{4.22b}$$

4. Backtracking:

$$q_t^* = \psi_{t+1}(q_{t+1}^*),\; t = T-1, T-2, ..., 1 \tag{4.23}$$

As previously stated, the Viterbi algorithm maximizes the probability during the recursion step. After that, the resulting state is used as a *back-pointer* in which during the backtracking step, the best sequence will be found. In Figure 4.3 is depicted how the backtracking step works.

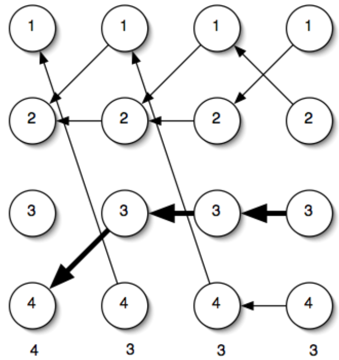


Figure 4.3: The backtracking step

4.6 Maximum likelihood estimation

The last part of the model is represented by the *Learning* phase, in which the system is able to decide what the final word pronounced by a user. With the usage of HMM models, it is possible to extract one or more sequences of states. The last piece of the puzzle is to estimate the sequence of words. To do so, a typical speech recognition system uses the *Maximum Likelihood estimation* (MLE).

Given a sequence of n *independent* and *identical* observations $x_1, x_2, ..., x_n$, assuming that the set of samples comes from a probability distribution with an *unknown density function* called $f_0(x_1, ..., x_n)$. The function belongs to a family of a certain kind of distributions in which θ is the *parameters vector* for that specific family.

Before using MLE, a *joint density function* must be specified first for all observations. Given the previous set of observation, the joint density function can be denoted as follows:

$$f(x_1, x_2, ..., x_n|\theta) = f(x_1|\theta) \times f(x_2|\theta) \times ... \times f(x_n|\theta) \tag{4.24}$$

Now, consider the same set of observations as a *fixed* parameters whereas θ is allowed to change without any constraint. From now on, this function will be called **likelihood** and denoted as follows:

$$L(\theta \sim x_1, x_2, ..., x_n) = f(x_1, x_2, ..., x_n|\theta) = \prod_{i=1}^{n} f(x_i|\theta) \tag{4.25}$$

In this case, $\sim$ indicates a simple separation between the parameters function and the set of observations.
Often, there is a need to use the *log* function; that is transform the likelihood as follows:

$$ln\, L(\theta \sim x_1, x_2, ..., x_n) = \sum_{i=1}^{n} ln\, f(x_i|\theta) \tag{4.26}$$

To estimate the log-likelihood of a single observation, it is necessary to calculate the average of equation 4.26 as follows:

$$\hat{l} = \frac{1}{n} ln\, L \tag{4.27}$$

The *hat* in equation 4.27 indicates that the function is an estimator. From here we can define the actual MLE. This method estimates the θ_0 by finding the value of θ that returns the maximum value of $\hat{l}(\theta \sim x)$. The estimation is defined as follows if the maximum exists:

$$\hat{\theta}_{mle} \subseteq \{\underset{\theta}{arg\,max}\ \hat{l}\ (\theta \sim x_1, x_2, ..., x_n)\} \tag{4.28}$$

The MLE corresponds to the so called *maximum a posteriori estimation* (MPE) of *Bayes rule* when a uniformed prior distribution is given. In fact, θ is the MPE that maximize the probability. Given the Bayes' theorem we have:

$$P(\theta|x_1, x_2, ..., x_n) = \frac{f(x_1, x_2, ..., x_n|\theta)P(\theta)}{P(x_1, x_2, ..., x_n)} \tag{4.29}$$

where $P(\theta)$ is the prior distribution whereas $P(x_1, x_2, ..., x_n)$ is the averaged probability of all parameters. Due to the fact that the denominator of the Bayes' theorem is independent from θ, the estimation is obtained by maximizing $f(x_1, x_2, ..., x_n|\theta)P(\theta)$ with respect of θ.

4.7 Gaussian Mixture Model

A Gaussian mixture model is a probabilistic model where it is assumed that the set of points comes from a *mixture model*, in particular, from a fixed number of *Gaussian distributions* where the parameters are *unknown*. This approach can be thought of a generalization of the clustering algorithm called *k-means* where we are looking for the **covariance** and the center of the Gaussian distribution and not only the centroids[6]. There are different ways of fitting the mixture model, but we are going to focus in particular to the one where the expectation-maximization is involved (see section 4.6).

Let the following equation defining a weighted sum of N Gaussian densities component:

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{N} w_i \, g(\mathbf{x}|\mu_i, \Sigma_i) \tag{4.30}$$

where $\mathbf{x}$ defines the set of features (data-vector) of continuous values. The sequence $w_i = 1, ..., N$ represents the set of mixture weights whereas the function $g(\mathbf{x}|\mu_i, \Sigma_i)$, $i = 1, ..., N$ defines the Gaussian densities component. The following equation specifies each Gaussian component's form:

$$g(\mathbf{x}|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2}\,|\Sigma_i|^{1/2}} exp\left\{-\frac{1}{2}(\mathbf{x} - \mu_\mathbf{i})'\ \Sigma_i^{-1}\ (x - \mu_\mathbf{i})\right\} \tag{4.31}$$

where $\mu_\mathbf{i}$ is the mean vector and Σ_i is the covariance matrix. Given that, we can assume that the mixture satisfy the constraint that $\Sigma_{i=1}^{N}\ w_i = 1$.

With the notation in equation 4.32, we can now define the complete GMM since all the component densities are parameterize by the covariance matrices, the mean vectors and the mixture weights [36].

$$\lambda = \{w_i, \mu_\mathbf{i}, \Sigma_i\}\ \ i = 1, ..., N \tag{4.32}$$

The choice of model configuration highly depends on the available dataset. In fact, to estimate the GMM parameters we have to determine the covariance matrix Σ_i. This can be either full rank or constrained to be diagonal. In the first case, all rows and columns are linearly independent and all the values are taken into account, whereas in the second case, we consider only the values in the diagonal. The covariance matrix is not the only parameter that needs to be carefully chosen. In fact, the *number of components* in general, refers to the amount of possible *"clusters"* in the dataset.

It is important to note that in recognition, it is allowed to assume the size of the acoustic space of the spectral. The spectral is referred to the phonetic events as we described in chapter 2. In fact, these acoustic classes have well defined features that allows the model to distinguish one phoneme from another. For the same reason, GMM is also used in *speaker recognition* in which the vocal tracts spectral is taken into account to distinguish a speaker from another [13].

Continuing with the speaker recognition example, the spectral shape i can be thought of as an acoustic class which can be represented by the mean μ_i of the $i - th$ component density. The variation in the spectrum can be defined as the covariance matrix Σ_i. Also, a GMM can be viewed as a Hidden Markov Model with a single state assuming that the feature vectors are independent as well as the observation density from the acoustic classes is a Gaussian mixture [36] [37].

[6] http://scikit-learn.org/stable/modules/mixture.html.

Chapter 5

Implementation

In this chapter we explain the infrastructure that performs all the necessary steps to produce efficient feedback. A general overview is given and for each section, we describe in particular the tools as well as the way we manipulated the data in order to obtain the information useful for the user. The chapter is divided in two parts: the first part focuses on the back-end and the services we used to extract the features we described in chapter 4. The second part describe the front-end, that is, the *Android*[1] application (called **PARLA**[2]) with a particular focus on the feedback page and the general usage.

5.1 General architecture

In 5.1 the general architecture of the infrastructure is shown. The flow displays only the *pronunciation testing* phase:

1) User says the sentence using the internal microphone of the smartphone (or through the headset)
2) The application sends the audio file to the *Speech Recognition service*
3) The result of step 2 is sent to the *Gaussian Mixture Model service* (or *Audio Analysis Service*)
4) The result of step 3 is sent back to the application where a *Feedback page* is displayed
5) A short explanation for each chart is given to the user
6) Back to step 1

The flow described above is the main feature of the whole project, although, the application also supplies two other important functionalities that are described more in detail in section 5.4. The first one is related to **critical listening** where the user is able to listen to the *native pronunciation* as well as to their own. This feature has a big impact on improving the pronunciation because it pushes the user to understand the differences as well as to emulate the way native speakers pronounce a specific sequence of words. The second feature regards the **history** (or progress). This page shows the trend of the user based on all the pronunciation he/she made during the usage of PARLA. The purpose of the history page is to help the user to see their progress and to get an idea of how to improve the pronunciation.

Implementation procedure

Several steps were made before reaching the architecture depicted in Figure 5.1. Generally speaking, the implementation was divided into two main categories: the first is composed of the *data collection and training* phase whereas the second is formed by the *mobile application* and *server communication*.

The very first step was to collect the data from native speakers and apply some pre-processing techniques in such a way that we were able to obtain only the information we needed to train the two services we had on the server. After the data collection, we trained both the models with the information we extracted in the previous step. The detailed procedures are described in sections 5.3.1 and 5.3.2.

[1] https://www.android.com

[2] https://github.com/davideberdin/PARLA

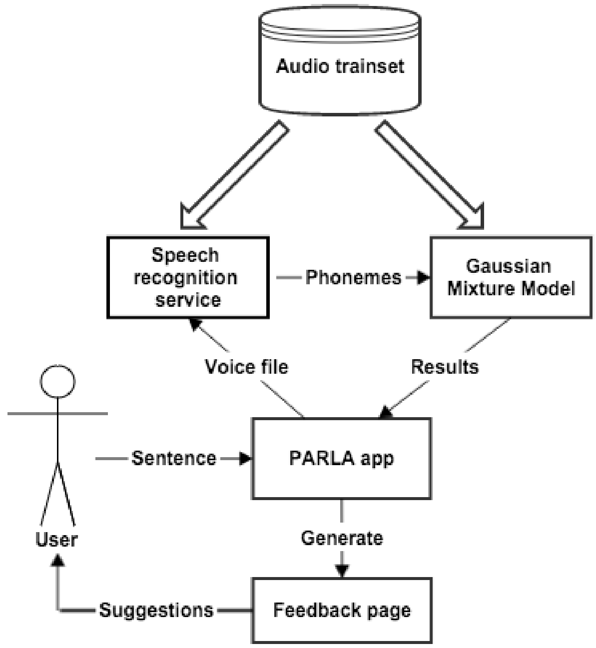


Figure 5.1: General architecture of the infrastructure

5.2 Data collection

The data collection step is a crucial phase of the entire project. The reason is that the audio record has to be clear, clean and as natural as possible. In fact, the people who participated in this phase were asked to pronounce the sentences as they would say them in a day-by-day conversation.

We recorded 8 people, 4 males and 4 females, at the University of Rochester using *Audacity*[3]. Each person had to pronounce 10 sentences (see Table 5.1) and each sentence was pronounced 10 times.
The sentences were chosen in order to cover the most used English sounds and based on the frequencies of everyday usage[4].

Sentences	
A piece of cake	Fair and square
Blow a fuse	Get cold feet
Catch some zs	Mellow out
Down to the wire	Pulling your leg
Eager beaver	Thinking out loud

Table 5.1: Idioms used for testing the pronunciation

The number of files gathered is 800 and the average length of each file is **1s**. In total, 14 minutes of recorded audio was gathered. This amount of time was sufficient for training the speech recognition model and the GMM. In reality, for the speech recognition service, the model was initially trained with a bigger dataset and then the sentences were added later (details in Figure 5.4). The reason is that the tool used for the speech recognition requires a much larger dataset[5].

5.2.1 Data pre-processing

The data pre-processing step is one of the most important procedures of the whole project. In fact, extracting the right information is crucial for both training the models and those voice-features that should be shown to the user.

[3] http://audacityteam.org
[4] http://www.learn-english-today.com/idioms/idioms_proverbs.html
[5] http://cmusphinx.sourceforge.net/wiki/tutorialam

The process starts by using the tool called **PRAAT**[6]. This tool is used for analysis of speech in phonetics as well as for speech synthesis and articulatory synthesis[38]. PRAAT was used to analyse the audio files we collected in the very beginning of the project and extracting formants and stress, which were described in sections 2.1.2 and 2.8.2. From here, a set of CSV files is generated where we saved the values of the formants and the stress for each audio file. These files are then used as input for a tool called **FAVE-Align**[39].

FAVE-Align is a tool used for *force alignment*[7]. This process is used to determine where a particular word occurs in an audio frame. In other words, FAVE-Align takes a text transcription and produces a PRAAT TextGrid file where it shows when those words start and end in the related audio file [27]. The tool performs different phases in order to align audio and text.

The first step is to sample the audio file and apply the Fourier Transformation because there is the need to move from the *time domain* to *frequencies domain*. From here, the tool extracts the *spectrum* and applies the Inverse Fourier Transformation onto it to obtain the so called **Cepstrum**. The *cepstrum* is the representation in a small-window frame of the spectrum. Although, the amount of information extracted from the cepstrum is too high, and so the tool uses *Perceptual Linear Prediction coefficients* to retrieve the necessary data to perform the alignment decision. These coefficients are used for feature extraction. The detailed process can be found at [40].
The last part of this process is the decision making part and this is done by a *Hidden Markov Model*.

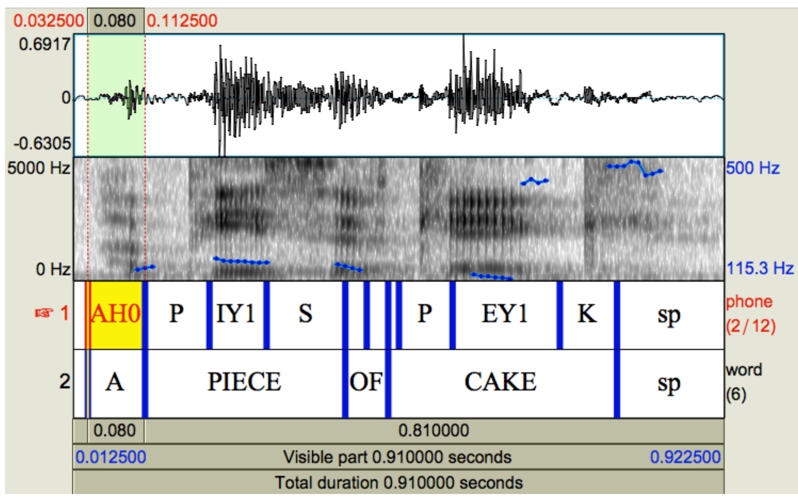


Figure 5.2: Result from FAVE-Align tool opened in PRAAT

The outcome of the previous step is used as input for the tool called **FAVE-Extract**. This tool helps to automate the vowel formant analysis. The process is dived in two main steps: the first is finding the *Measurement Points* and the second is the *Remeasurement*.

For most vowels it is possible to find the measurement point by listening 1/3 of the total duration[9]. This point is necessary for determining the identity of the vowel, that is, the name of the vowel itself. For more complex vowels, a different approach is done; that is, the point is halfway between the F1 (main formant) maximum value and the beginning of the segment. In addition, the LPC analysis is performed on both beginning and end of the vowel in order to pad the vowel's window. This is to *ensure a formant track through the full vowel's duration*[41]. The result of this step is a set of candidates. This set is composed of the potential formants estimated from the likelihood of the **ANAE** distribution. The *Atlas of North American English* (ANAE) is the set of phonology formants values depending on the English regional area. The winner formant is determined by the Posterior probability. This step does not take into consideration the provenience of the speaker.

The second part of the formants extraction tool is to remeasure the parameters by adjusting the ANAE distribution based on the regional area of the speaker. In this way, the formant value will be more accurate. An example of result

[6]http://www.fon.hum.uva.nl/praat/
[7]http://www.voxforge.org/home/docs/faq/faq/what-is-forced-alignment?func=add;class=WebGUI::Asset::Post;withQuote=0.

from FAVE-Extract is shown in Figure 5.3.

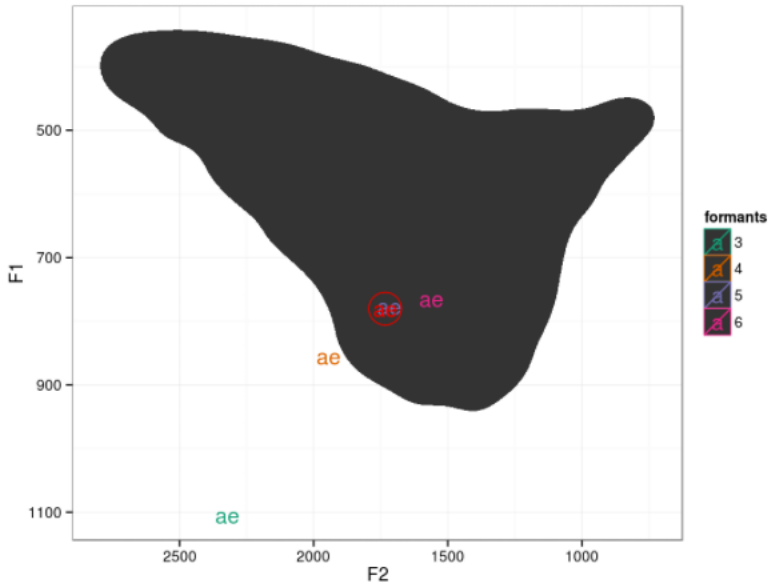


Figure 5.3: Result from FAVE-Extract

The result of the data pre-processing is a set of information composed of the average value of F1, F2 and F3 formants with their respectively vowels text representation. The formants values will be then used to train both the speech recognition model and the Gaussian Mixture Model.

5.3 Server

The back-end system is divided in two different services: the first one handles the speech recognition converting the user's voice into a set of phonemes, whereas the second service is in charged of all the other operations a user can do, such as login/logout, history data, vowels prediction system, usage collection, etc.. This section explains more in detail how the information is extracted from the audio files and manipulated before giving the feedback to the user.

5.3.1 Speech Recognition service

The first service in order of usage within the whole system is the speech recognition one. This has been made possible by using the well-known **CMU-Sphinx** software by Carnegie Mellon University [42]. The framework is written in Java and it is completely open-source. The system has been deployed on a *Tomcat*[8] service as Java Servlet to serve the requests from the Android application.

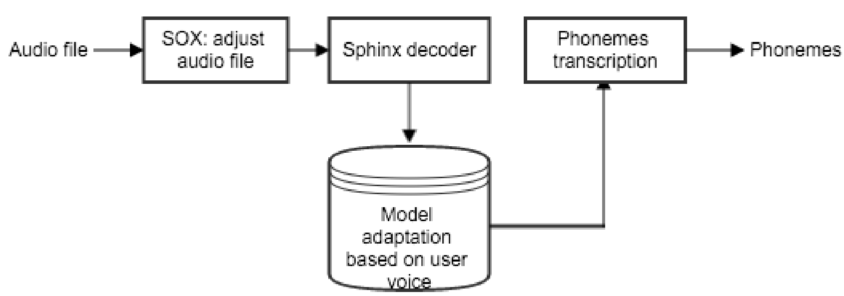


Figure 5.4: Architecture of the Speech recognition service

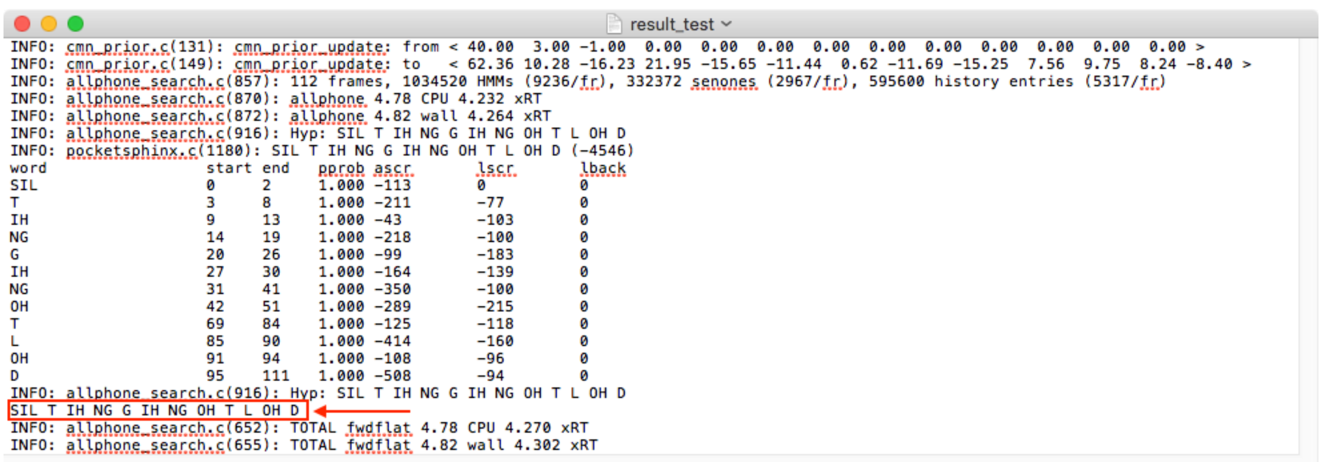
[8] https://tomcat.apache.org

The first phase consisted in training the audio model with two different language models. The first (and largest) is the *Generic U.S. English model* whereas the second is composed of the data audio-files collected from the native speakers. The first dataset is directly provided by the tool and already embedded in the decoder. This means that the system has been already trained with a generic model so that new developers do not have to collect data to train the model. This project is a special case because it focuses attention on only 10 specific sentences; in order to specialize the language model, specific files had to be added. This phase took several hours of work because the amount of data used was very large.

Once the model has been trained, the parameters can be adjusted based on the voice of the user. For this task, CMU-Sphinx provides a particular method that permits the model to be adapted based on pitch and speed-of-speech of the user. To do so, the system had to be built in such a way that for each user, a specific file with the voice's parameters was created. In this way, CMU-Sphix would improve the recognition every time a user feeds the system with audio files.

At this point the system is trained and ready to recognize. When the service receives an audio file, the first step before proceeding to CMU-Sphinx is to change some properties of the audio file itself. The Sphinx decoder has the best performance only when the audio files are in *mono-channel* and have a sampling frequency of *16Khz*[9]. The library we used to record the user in PARLA, is sampled in *stereo-channels* and *11Khz*. For this reason, a special tool called **SOX**[10] was used to change the properties of the audio file according to the required ones.

Once the file has been manipulated, the voice's parameters file of the user is retrieved and used to start the recognition part. CMU-Sphinx goes through several internal procedures (general details in chapter 4) and during this process it adapts the model based on the user's voice. At the end of the whole process, a string containing the phonemes of the pronounced sentence is given back as result. An example is given in Figure 5.5. The red box indicates the result taken into consideration.




```
INFO: cmn_prior.c(131): cmn_prior_update: from < 40.00  3.00 -1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 >
INFO: cmn_prior.c(149): cmn_prior_update: to   < 62.36 10.28 -16.23 21.95 -15.65 -11.44  0.62 -11.69 -15.25  7.56  9.75  8.24 -8.40 >
INFO: allphone_search.c(857): 112 frames, 1034520 HMMs (9236/fr), 332372 senones (2967/fr), 595600 history entries (5317/fr)
INFO: allphone_search.c(870): allphone 4.78 CPU 4.232 xRT
INFO: allphone_search.c(872): allphone 4.82 wall 4.264 xRT
INFO: allphone_search.c(916): Hyp: SIL T IH NG G IH NG OH T L OH D
INFO: pocketsphinx.c(1180): SIL T IH NG G IH NG OH T L OH D (-4546)
word                    start end   pprob ascr      lscr      lback
SIL                     0     2     1.000 -113      0         0
T                       3     8     1.000 -211      -77       0
IH                      9     13    1.000 -43       -103      0
NG                      14    19    1.000 -218      -100      0
G                       20    26    1.000 -99       -183      0
IH                      27    30    1.000 -164      -139      0
NG                      31    41    1.000 -350      -100      0
OH                      42    51    1.000 -289      -215      0
T                       69    84    1.000 -125      -118      0
L                       85    90    1.000 -414      -160      0
OH                      91    94    1.000 -108      -96       0
D                       95    111   1.000 -508      -94       0
INFO: allphone_search.c(916): Hyp: SIL T IH NG G IH NG OH T L OH D
SIL T IH NG G IH NG OH T L OH D
INFO: allphone_search.c(652): TOTAL fwdflat 4.78 CPU 4.270 xRT
INFO: allphone_search.c(655): TOTAL fwdflat 4.82 wall 4.302 xRT
```

Figure 5.5: Example of phonemes recognition using CMU-Sphinx for the sentence *Thinking out loud*. The phoneme **SIL** stands for *Silence*

5.3.2 Voice analysis system

The second service handles the analysis of the audio file in order to give feedback to the user. This process is long because it involves several steps and sometimes the user had to wait up to 40 seconds before receiving the results. Figure 5.6 depicts the macro-view of the service's architecture.

The system was written in Python using Django[11] as web-framework. The choice was made based on the availability of machine learning libraries and the language tools (FAVE-extract and FAVE-align). In fact, we used *scikit-learn*[12],

[9]http://cmusphinx.sourceforge.net/wiki/faq

[10]http://sox.sourceforge.net

[11]https://www.djangoproject.com

[12]http://scikit-learn.org/stable/

a well-known python library for data-analysis, data mining and machine learning.

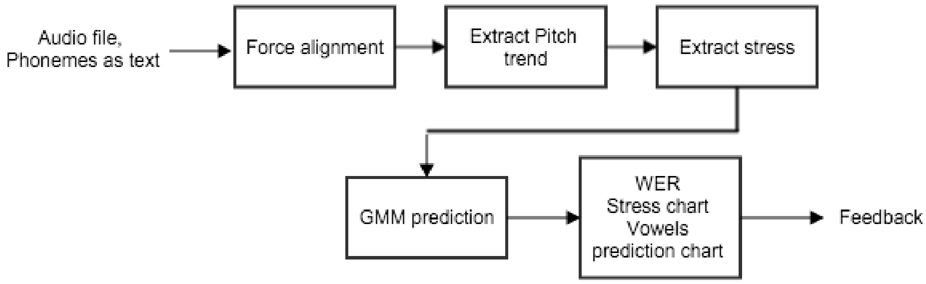


Figure 5.6: Architecture of the Voice analysis service

5.3.3 Training GMM

As for the speech recognition service, the Gaussian Mixture Model had to be trained to incorporate the audio features of the native speakers. As explained in section 5.2.1, formants F1, F2 and F3 formed the training dataset for this model. The first three formants are sufficient for recognizing the phoneme that has been pronounced because the first two formants are not enough for discriminating the *"value"* of the phoneme due to a big overlapping in their spectrum. Using the third formants, those frequencies can be caught to act as decision makers[43].

Scikit-learn provides a GMM out of the box. Although, the number of parameters available make it hard to properly set the model. For this reason, we used a method called **Bayesian information criterion** (BIC) to find the optimal solution for our purpose.

BIC is a model selection method that gives a score on an estimated model performance based on a testing dataset. The lower the score, the better the model is.
Equation 5.1 is the formula used for calculating the score, where T is the size of the training set and $\ln \hat{L}$ is the maximum likelihood value of the given model (details in section 4.6), whereas k is the number of *free* parameters that can be estimated.

When the BIC method is attempted, it tries to avoid the risk of *overfitting* the model by injecting a *penalty term* of $k \cdot \ln(T)$ that augment proportionally with the number of parameters[13]. This term also helps to avoid unnecessary parameters and keep the model as simple as possible. In Figures 1 and 2 of the Appendix, the BIC evaluations are shown.

$$BIC = -2 \cdot \ln \hat{L} + k \cdot \ln(T) \tag{5.1}$$

Given the results of the evaluation, the model parameters with the lowest BIC score were selected. Listing 5.1 displays the code used to create the classifier after having run the BIC evaluation.

Listing 5.1: Parameters of GMM classifier

```
gmm_classifier = mixture.GMM(n_components=12, covariance_type='full',
                             init_params='wmc', min_covar=0.001, n_init=1,
                             n_iter=100, params='wmc', random_state=None,
                             thresh=None, tol=0.001)
```

[13] http://stanfordphd.com/BIC.html

The next list of parameters are those that have been automatically selected by the evaluation whereas the others are set by default:

- *Number of components* decided based on the total amount of possible phonemes (in our case, 12)
- *Covariance type* set to **full** as indicated by BIC
- *Initial parameters* updated by **weight**(w), **means**(m) and **covariance**(c), as indicated by BIC
- *Tol* is the convergence threshold. The Expected Maximization breaks when the average gain log-likelihood is below **0.001**

After the training part, we tested the classifier with a testing set composed by the first 3 Formants that we extracted using PRAAT from 5 audio files provided by the same person. Both *Training accuracy* and *Testing accuracy* were calculated using the function **numpy.mean()** where the average is computed along the axes that has been specified.

Listing 5.2: Code for accuracy estimation of training and testing set

```
train_accuracy = numpy.mean(y_train_predicted == y_train) * 100
test_accuracy = numpy.mean(y_test_predicted == y_test) * 100
```

Table 5.2 shows the results after the training of Gaussian Mixture Model. The accuracy values can be improved by increasing the amount of training data.

Table 5.2: Testing results after the training

Sentence	Training Accuracy	Testing Accuracy
A piece of cake	**82.5%**	**90.7%**
Blow a fuse		
Thinking out loud		
Mellow out		
Eager beaver		

5.3.4 Pitch, stress and Word Error Rate

After the training phase, we built three other components were build to deal directly with the feedback to the user. PRAAT was used to extract the *pitch contour* in order to show the user the way his/her voice changes compared to a native speaker. Figure 5.7 shows an example of contour that was used as feedback for the user. It is possible to notice that both the natives have a similar way of saying the same sentence. This is a key point because the non-native will compare the way he/she will pronounce the sentence and understand the eventual differences.

Stress is extracted in a different way. Instead of using PRAAT, *FAVE-extract* was chosen because it provides a feature that retrieves the stress position(s) in the sentence. Moreover, it offers the opportunity to know on which phoneme the stress occurs. Given that, to the user it will be presented the phoneme representation of the pronounced sentence provided by the speech recognition service as well as in which phonemes the stress is emphasized.

The last piece of the system is to calculate the difference between the pronunciation of the native and the user, from the phoneme point of view. For this purpose, a well-known evaluation metric system called **Word Error Rate** (WER) was used.

WER is an evaluation metric system for checking the accuracy of a speech recognition's system. The main idea is to calculate the distance between the **hypothesis** and **reference**. The first is the result produced by the system whereas the second is the expected text. The distance is measured by calculating the minimum number of edits that are needed to change one single character to another. Likewise, WER calculates the minimum amount of operations that have to be done for moving from the reference to the hypothesis.

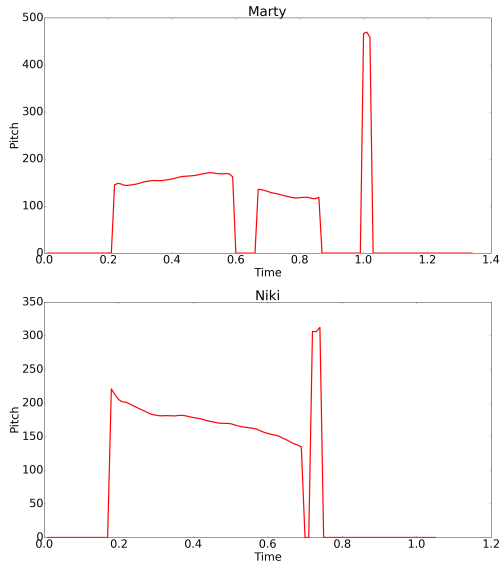


Figure 5.7: Example of pitch contour provided by two native speakers for the sentence *Mellow out*

The possible edits are:

- *Insertion*: a word was added to the hypothesis
- *Substitution*: an aligned word from the hypothesis has been substituted in the reference
- *Deletion*: a word has been deleted in the reference

The calculations are done by putting each edit on a Levenshtein distances table and then *backtracing* in it through the shortest path to the origin $(0, 0)$ [44]. Each step during the backtrace is counted. After this, WER uses the formula in equation 5.2 to calculate the *error rate*.

$$WER = \frac{S + D + I}{N} \tag{5.2}$$

where S is the substitutions, D the deletions, I the insertions and N are the words in the reference text.

5.4 Android application

The choice of using the Android OS for developing the mobile application was in part forced by the fact that the other mobile OS do not allow installing applications outside their respective stores. using Android allowed for the unrestricted distribution of the application and for more flexibility regarding the implementation.
We used *Android Studio*[14] as IDE and the *API level 21* where the minimum Android version required is 5.0.

5.4.1 Layouts

The application is composed of four main layouts: *pronunciation page*, *feedback page*, *history page* and *critical listening page*. Among these, the **feedback page** is the most important one since it provides the differences with the native pronunciation.

The **pronunciation page** is depicted in Figure 5.8 and it is possible to notice that the user has access to a multitude of options, such as: listening to the native speaker, change word, see the *IPA* phonetics and, of course, test his/her pronunciation.

[14]http://developer.android.com/tools/studio/index.html

The **critical listening** section has been created to help the user to understand the differences between what he/she said and the native, based only on the audio. The page is split in two parts: on the left side there are all the native pronunciations whereas on the right there are all the non-native ones. For each sentence it is then possible to either *test the pronunciation* or *see the history*. The first choice redirects the user to the Main page whereas the second will show the eventual progress for that particular sentence.

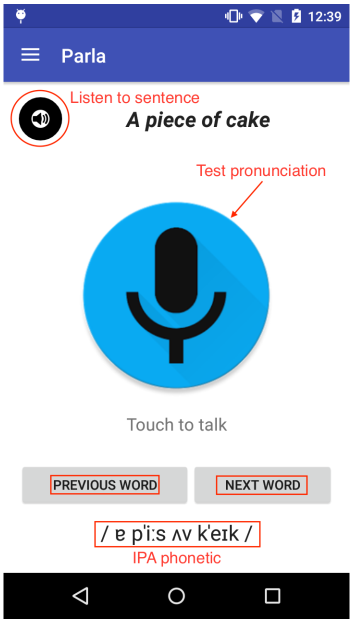


Figure 5.8: Pronunciation (or Main) page of PARLA

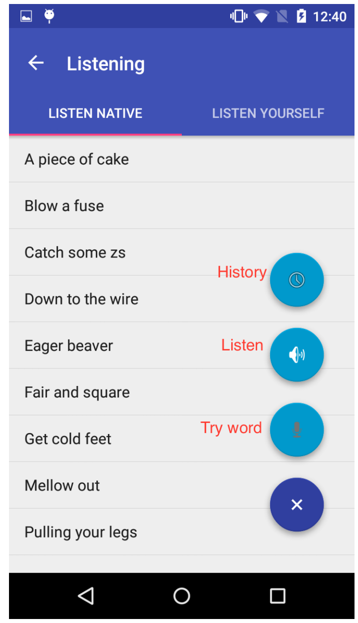


Figure 5.9: Listening page

The **history page** provides a simple visualization of the user's progress. Figure 5.11 shows that the page is split in two parts: the top part shows the vowels pronunciation of each time the user tested the pronunciation whereas the bottom part shows how *close* the articulation of that specific vowel was to the native.

These layouts have been designed to provide the necessary information prior testing the pronunciation with the only exception of the history page. As discussed in chapter 1, *listening* and *phonetics* help the student to improve the quality of the pronunciation as well as the correctness. Keeping these statements in mind, we designed the pages in order to achieve the maximum effect.

The initial development of the User Interface included a simple study wherein a group of 4 people were asked to interact with the sketches of the initial UI. The process was straightforward because the purpose and goals of this application were explained at the beginning of the study. The study was based on a sequence of questions aimed to improve the usability. The investigated topics were:

- Navigation among pages
- Modifications in the main page
- Modifications in the critical listening page
- Modifications in the history page
- Modifications in the feedback page
- Add/Remove features

Based on the answers the layouts were changed accordingly.

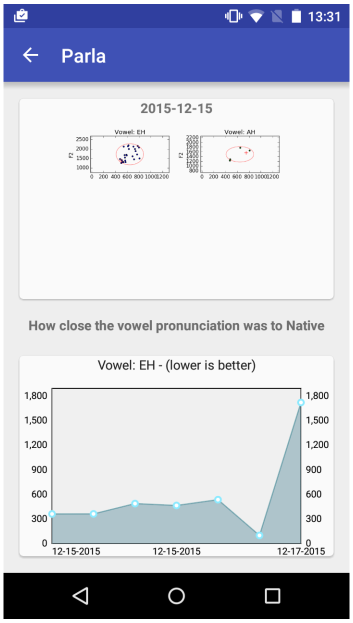


Figure 5.10: Example of History page

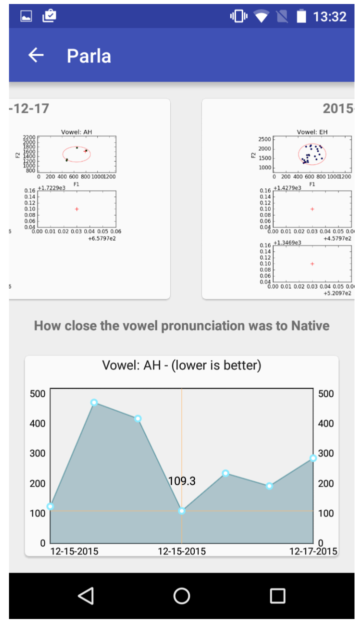


Figure 5.11: History page with interaction

5.4.2 Feedback layout

This *feedback page* has been designed to provide as much information as possible giving the minimum explanation regarding the differences. Basically, attention was focused on creating charts and the phonetic representations in order to give feedback. This page combines all the information from both the speech recognition system and the voice analysis service.

The page is divided in three main parts. The first part is represented by the phoneme representation and the WER located at the very top of the page. The button on top left provides the meaning of the sentence as well as a typical usage in a context.
For each sentence we show the comparison between the native and the user. Figures 5.12 and 5.13, show the user immediately can understand those differences, if any. The syllables highlighted in red represent the **stress** of that particular sentence. In addition, the *word error rate* shows how different the user's pronunciation was from the native speaker's. The second picture shows that the user mispronounced the word **to**, that is why *WER* is 10%. The stress is correct in both cases, otherwise it would be highlighted.

The second part is represented by the chart in which the *stress trend* (or *pitch contour*) is depicted. This chart provides a graphical representation of how the sentence should be emphasized. Similarly to the first part, the difference between the native and the user is displayed. However, the two trends will **never** be the same because the process of extracting this information involves the *voice pitch* of a person. Basically, what the user should pay attention to, is the *shape* of those lines. If the trends are similar, then it means that the stress is in the right position during the pronunciation. Figure 5.14, for example, depicts a very bad pronunciation. In fact, WER is 75% and the stress trend does not look like the native's one. The picture clearly shows the impact on a user when making mistakes in the pronunciation.

The last part of this page is represented by the *vowels prediction* chart. Here we show the various pronunciation formants values of the vowels involved in a particular sentence. These values are extracted from the GMM in the voice analysis service. Figure 5.15 shows how the vowels formants are well defined and clustered together. The circles represent the range of formants values in which that determined vowel should be pronounced. The *red crosses* are the user's formants prediction. The feedback information here is that the user should understand that if the red cross is not within the circle and close to the group of green dots, then he/she should change the pronunciation. To improve this there are two methods: using the critical listening or looking at the numbers in the chart. The first method is simpler and more effective whereas the second option is for those that have a prior knowledge in linguistics.

It is important to mention that the user can interact with all the charts. In fact it is possible to zoom in/out and

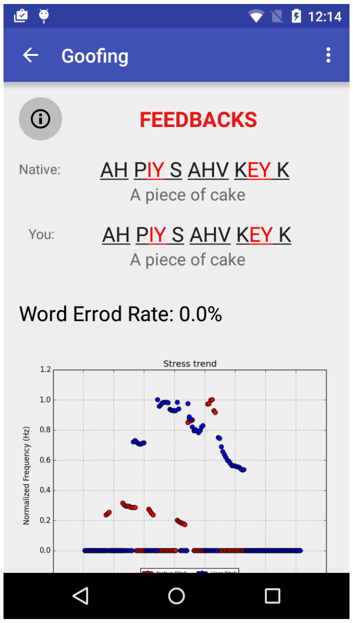


Figure 5.12: Correct pronunciation

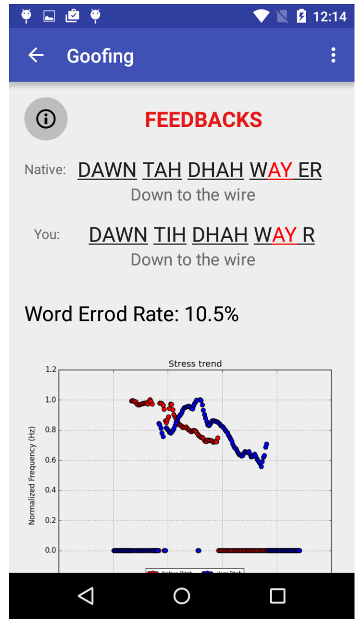


Figure 5.13: Small error in pronunciation

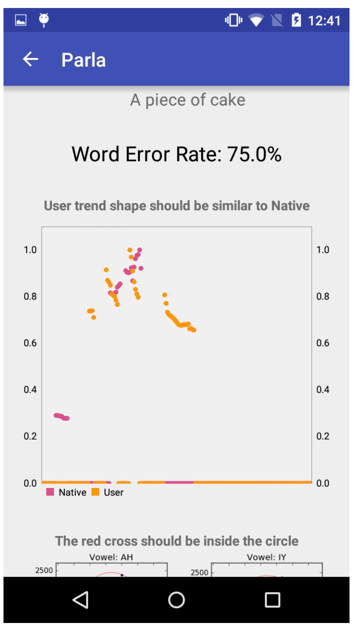


Figure 5.14: Stress contour chart

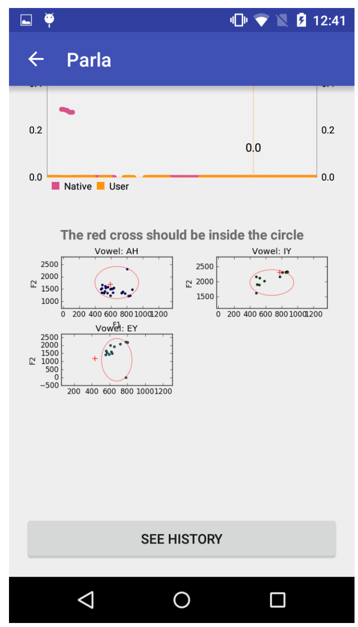


Figure 5.15: Vowels prediction representation

retrieve the value of each single point by tapping on top of the line. This interaction allows users with a linguistic background to have a better numeric-understanding on how the pronunciation was done.
On the very end of this page, the user can navigate directly to the history and see the progress he/she made for that specific sentence.

Chapter 6

User studies and Results

The results of this study were determined by the answers to a survey completed by the users that participated in the testing phase.
We recruited 6 people from Uppsala University and asked them to use the application for a period of 2 weeks and fill up a survey with 26 questions (see Appendix). The survey is anonymous and divided in 3 sections: the first part was designed to gather the information related to the audience. The second part aimed to rate the interest in learning a new language using a mobile device, and the third part was dedicated to the application itself.

6.1 Audience

This section presents the answers related to the users personal information to get a better understanding of the audience. From Figures 6.1 and 6.2 we can say that the majority of our users are male and between the age of 24-29 years old. Table 6.1 describes the native language of the users.

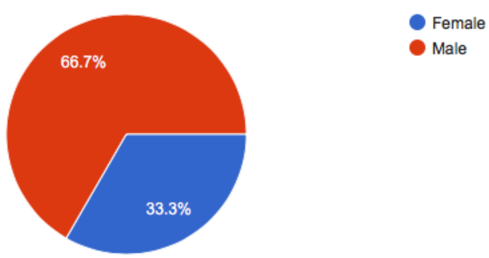


Figure 6.1: Gender chart

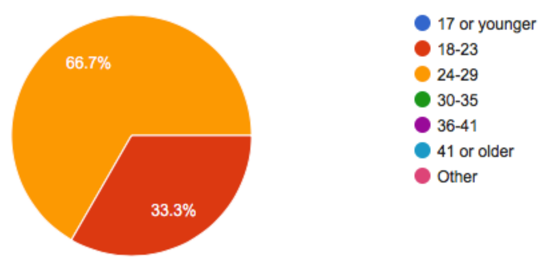


Figure 6.2: Age chart

Native language	Amount
Italian	2
Greek	2
Swedish	1
Arabic	1

Table 6.1: Users native languages

6.2 Interest

This section describes the interest of our testers in learning and improving a new language using a mobile application instead of the traditional student-teacher class. Results are positive and confirm that the interest is high. In particular, avoiding the interaction with a physical teacher is very welcomed. In fact, the interest in not having this sort of supervision, the *standard deviation* is 0.8367, the *mean* is 4.5 and the *variance* is 0.7 (Figure 6.7).

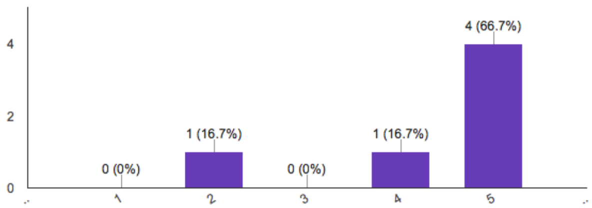


Figure 6.3: Interest in learning a new language

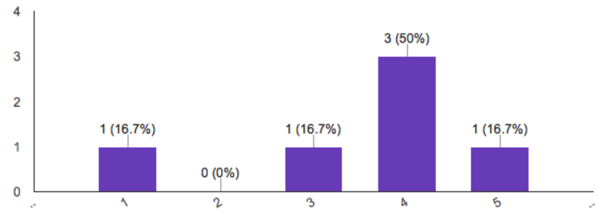


Figure 6.4: Interest in improving English language

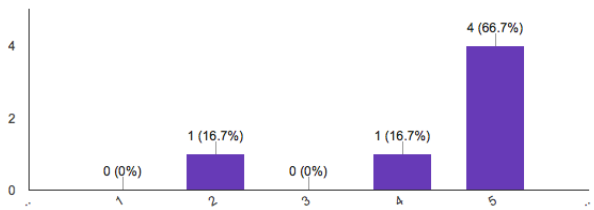


Figure 6.5: Interest in using a smartphone

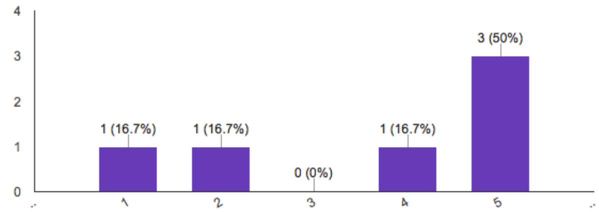


Figure 6.6: Interest in having visual feedback

Learning a new language has received a positive interest because the *mean* is 4.3 with a *standard deviation* of 1.2 and a *variance* of 1.4 (Figure 6.3). This indicates that users are eager to acquire new linguistic competencies. The same positive interest was given to the usage of a smartphone as a way of learning. In fact the *mean* is 4.3 with a *standard deviation* of 1.2 and a *variance* of 1.4 (Figure 6.5). These two results go along with the fact that people want to learn new languages and avoid the direct supervision with a teacher. The usage of a smartphone is an effective way for delivering linguistic knowledge.

A slight difference was observed concerning the English pronunciation and the visual feedback. In fact, according to our results, people are more interest in acquiring new languages rather then improving the one that they have already a good knowledge of. Looking at the results, we observed that the interest of improving English has a *mean* of 3.5 with a *standard deviation* of 1.3 with a *variance* of 1.9 (Figure 6.4), whereas, the interest of using visual feedback as approach of learning has a *mean* of 3.6 with a *standard deviation* of 1.7 with a *variance* of 3 (Figure 6.6).

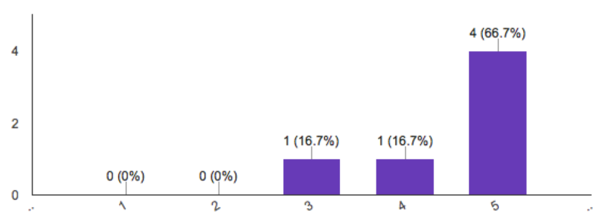


Figure 6.7: Interest in not having a teacher's supervision

6.3 Application

The following charts are the results of the survey's questions related to the application itself. The questions were designed in order to understand the feelings about how the users understood the different features provided by the application. These questions are divided into three sub-categories: the first one is related to a broad view of the product, the second category aims to define the *understanding* of the users about the features, whereas the third one, how useful these features are in order to improve the pronunciation.

The *general appreciation* received a positive feedback. In fact, the *mean* is 3.3 with a *standard deviation* of 0.8 and a *variance* of 0.6 (Figure 6.9). Also, the users have expressed a positive interest in continuing using the application if there would be a real product on the market in the future. The *mean* is 3.1 with a *standard deviation* of 0.7 and a *variance* of 0.5 (Figure 6.12). As last question of this first sub-category, we asked *"how difficult was the usage"* of the entire system. Users responded with a *mean* of 4 with a *standard deviation* of 1 and a *variance* of 1.2 (Figure 6.10).

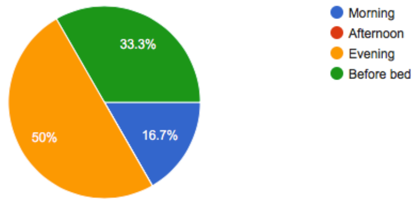


Figure 6.8: Moment of the day

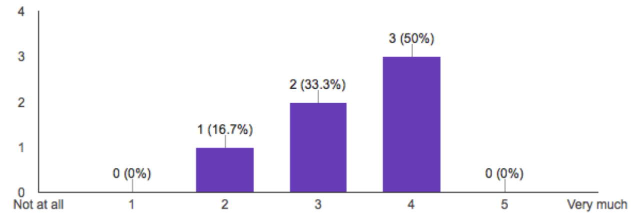


Figure 6.9: General appreciation

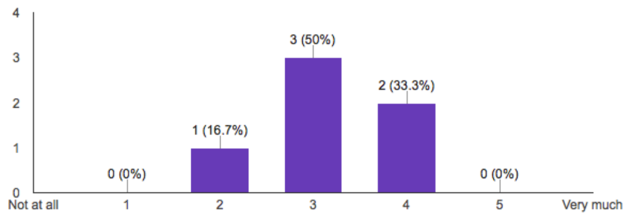


Figure 6.10: Interest in continuing using the application

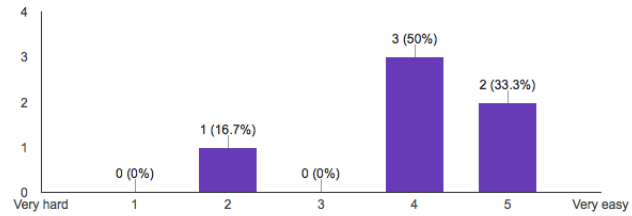


Figure 6.11: Usage difficulty

The *"Understanding"* sub-category received a slightly different appreciation level. In fact, according to the results of the survey, the users had some difficulties in understanding the usage of the charts in the feedback page. However, regarding the main page and the critical listening, the results were still positive. The main page received a *mean* value of 3.8 with a *standard deviation* of 1.1 and a *variance* of 1.3 (Figure 6.12), whereas the critical listening had a *mean* of 4.1 with a *standard deviation* of 0.9 and a *variance* of 0.9 (Figure 6.13). Basically, the users clearly understood the meaning and the usage of all the functionalities regarding the two pages in the application.
The main overview of the feedback page did receive good feedback as well. In fact we had a *mean* of 3.1 with a *standard deviation* of 1.1 and *variance* of 1.3 (Figure 6.14). Despite these results, the inner functionalities of the feedback page, have received low scores. According to the results, the *vowels charts* had the lowest *mean*, with a value of 2.5 and a *standard deviation* of 0.8 and *variance* of 0.7 (Figure 6.17). This a clear indication that the users did not properly understood the way the chart worked. The *pitch trend chart* received a slightly better score with a *mean* of 2.6 and a *standard deviation* of 1 with a *variance* of 1 (Figure 6.16).
The *stress* had a *mean* of 2.6 with a *standard deviation* of 0.8 and *variance* of 0.6 (Figure 6.15), whereas the *history page* had a *mean* of 3.5 with a *standard deviation* of 1 and *variance* of 1.1 (Figure 6.18).

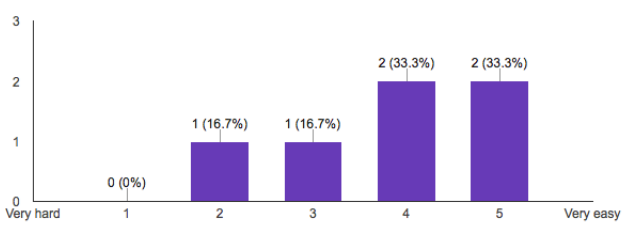


Figure 6.12: Understanding the main page

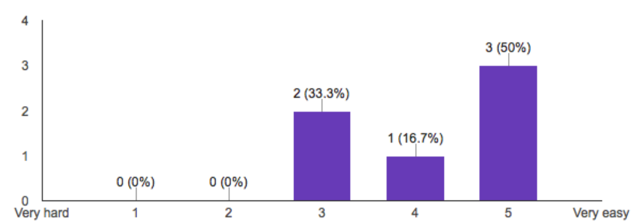


Figure 6.13: Understanding the critical listening page

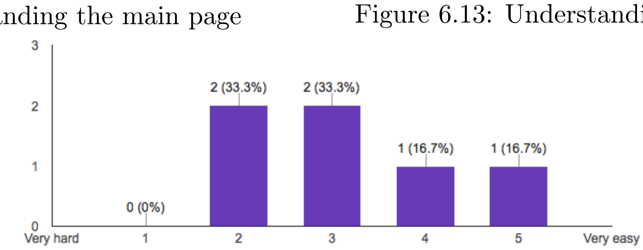


Figure 6.14: Understanding feedback page

The last sub-category is regarding on *"how useful are the features"* in order to improve the pronunciation. Unfortunately, the users did not find the *listening* or the *history* to be useful. In fact, for the first one, we had a *mean* of 2.1 with a *standard deviation* of 1.3 and a *variance* of 1.7 (Figure 6.20), and the second one had a *mean* of 1.6 with a *standard deviation* of 0.8 and *variance* of 0.6 (Figure 6.22). These results clearly show that the feeling of the users

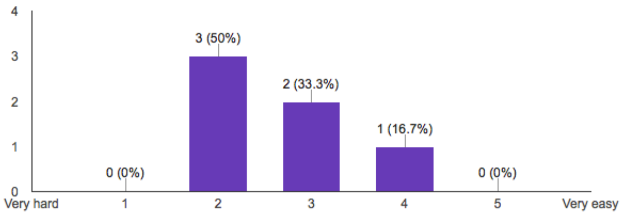


Figure 6.15: Understanding stress on a sentence

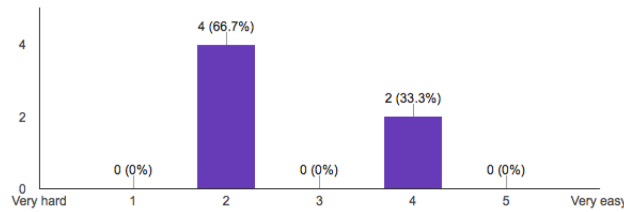


Figure 6.16: Understanding pitch trend

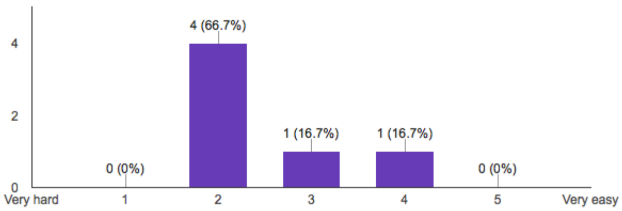


Figure 6.17: Understanding vowels chart

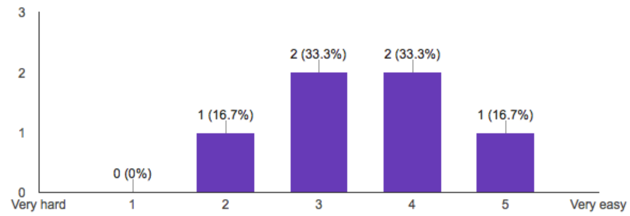


Figure 6.18: Understanding history page

regarding these two functionalities were very similar. Slightly better for the feedback page: we had a *mean* of 2.3 with a *standard deviation* of 1.2 and a *variance* of 1.4 (Figure 6.21).
The very last question aimed to find out whether the users actually improved the pronunciation or not. The results were not particularly encouraging because we had a *mean* of 2.1 with a *standard deviation* of 1.3 and a *variance* of 1.7 (Figure 6.19).

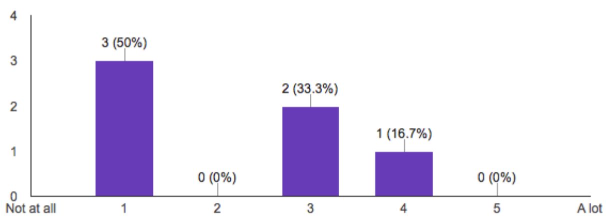


Figure 6.19: Pronunciation improved

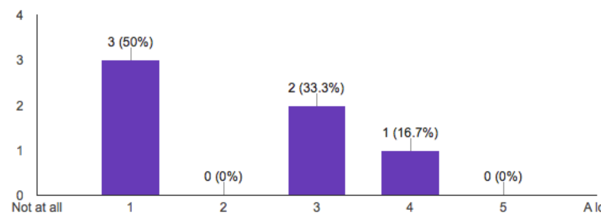


Figure 6.20: Utility of critical/self listening

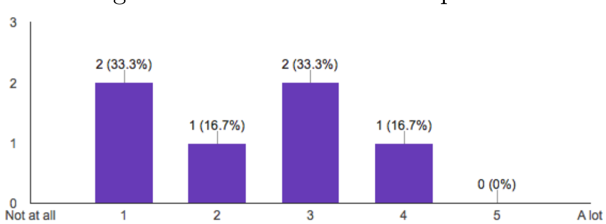


Figure 6.21: Utility of feedback

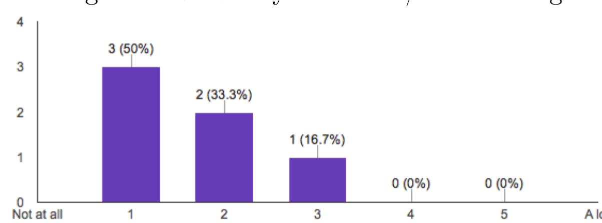


Figure 6.22: Utility of history page

Chapter 7

Conclusions

In conclusion, we can claim that the application has a lot of potential and there is a lot of room for improvement. Looking at the results, users did not significantly improve the pronunciation with the tools we provided. After a careful analysis of the data, we think that users did not use the application enough for seeing an actual improvement. There could be multiple reasons for that: few sentences available, few indications on improving the pronunciation, long waiting time in order to get feedback, etc.
During the design of the tests, we thought that using a small group of testers would be enough for this particular application, but probably the choice was not correct. Also, asking people with already a sufficient knowledge of the English language, was not a good choice as well. We also think that users did not understand why we included features such as the *critical listening* and the *history page*. One reason could be that, despite linguistic research claiming that these two features are very useful during the learning process, for pronunciation purpose, users need something else. Discovering these actual needs, goes beyond the scope of our research. To be sure, we understood that these two methods are not particularly important for this process.
The results related to the feedback page were not particularly positive. We think that a more careful study on how to deliver the feedback to user is necessary in the future, or rather, finding a way to give clearer directions on how to improve the pronunciation. However, the aim of the project was to avoid the usage of words in order to give feedback, but simply to use visual information. We find out that this type of information is very hard to deliver and in the future, a better/different approach is definitely necessary.

Despite some low scores, our testers appreciated the prototype and the way we delivered the idea. People need this kind of application to learn and improve languages, and the indicated interest regarding the usage of mobile devices to become well-versed in other languages is very high.

Chapter 8

Future Works

For future works, the method of delivering the feedback has to be improved with a more careful design of the user interface. Also, providing more sentences to pronounce as well as more indications on how to improve the pronunciation, are the definitely necessary. Moreover, a larger number of testers with less knowledge of the English language could give better data.

Further, other applications can be extracted from this prototype. Smartwatches for example are becoming the next hot-platform for developing new applications. In fact, it is possible to extend this product in such a way that a user can practice day-by-day by simply using the internal microphone of the smartwatch. The procedure and the time taken for the whole process is less then using a common smartphone. Of course, the whole feedback system has to be redesigned and scaled to be able to fit the information onto a smaller screen.

Another interesting way for pushing the limits of this application is to make it more challenging, more like a video game. In fact, providing the opportunity for the user to challenge other users should give a psychological boost for improving the pronunciation and being better than other competitors. Thus, the usage of achievements, objectives, etc. will involve the user in a completely different experience but still with the intent of improving the pronunciation.

Google Glass[1], *Microsoft HoloLens*[2], *Oculus Rift*[3] and other augmented reality devices, could be used for language learning process. The user will then be involved in an experience that would be closer to an actual lecture with a qualified teacher. Using a virtual assistant and a complex AI system, it would be possible to reproduce this old, but still very effective, way of learning. At the same time, interaction with other users that have the same application and device, would be incredibly effective to train not only the pronunciation but also grammar, reading-comprehension and conversation.

The number of possible and future applications is incredibly large. These were simple examples of how up-and-coming technology could be employed in the world of learning languages.

[1]https://www.google.com/glass/start/
[2]https://www.microsoft.com/microsoft-hololens/en-us
[3]https://www.oculus.com/en-us/

Bibliography

[1] J. Glass and V. Zue, "6.345 automatic speech recognition," Spring 2003. http://ocw.mit.edu, (Massachusetts Institute of Technology: MIT OpenCourseWare), (Accessed 23 Sep, 2015). License: Creative Commons BY-NC-SA.

[2] "The spectrum of acousting," 2015. accessed 2015-09-28. Available: http://www.hum.uu.nl/uilots/lab/courseware/phonetics/basics_of_acoustics_2/formants.html.

[3] "Rp vowel length: some details," 2015. accessed 2015-10-08. Available: https://notendur.hi.is/peturk/KENNSLA/02/TOP/VowelLength0.html#lengths.

[4] "Example of autocorrelation," 2015. accessed 2015-10-28. Available: http://www.eng.usf.edu/~lazam2/Project/sht_time_timedom/xmp_acr.htm.

[5] M. Gales and S. Young, "The application of hidden markov models in speech recognition," *Foundations and trends in signal processing*, vol. 1, no. 3, pp. 195–304, 2008.

[6] T. M. Derwing and M. J. Munro, "Second language accent and pronunciation teaching: A research-based approach," *Tesol Quarterly*, pp. 379–397, 2005.

[7] M. Rost and C. Candlin, *Listening in language learning*. Routledge, 2014.

[8] "Word stress - british council," 2015. accessed 2015-09-28. Available: https://www.teachingenglish.org.uk/article/word-stress.

[9] I. Rosenfelder, J. Fruehwald, K. Evanini, and J. Yuan, "Fave (forced alignment and vowel extraction) program suite," *U RL http://fave. ling. upenn. edu*, 2011.

[10] P. Boersma and D. Weenink, "Praat: doing phonetics by computer," 2010.

[11] P. Lamere, P. Kwok, E. Gouvea, B. Raj, R. Singh, W. Walker, M. Warmuth, and P. Wolf, "The cmu sphinx-4 speech recognition system," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2003), Hong Kong*, vol. 1, pp. 2–5, Citeseer, 2003.

[12] S. R. Eddy, "Hidden markov models," *Current opinion in structural biology*, vol. 6, no. 3, pp. 361–365, 1996.

[13] D. A. Reynolds, "A gaussian mixture modeling approach to text-independent speaker identification," 1992.

[14] K. N. Stevens, *Acoustic phonetics*, vol. 30. MIT press, 2000.

[15] "What are fricatives ?," 2015. accessed 2015-10-28. Available: http://www.pronuncian.com/Lessons/default.aspx?Lesson=9.

[16] "Consonants: Stops," 2015. accessed 2015-10-28. Available: http://facweb.furman.edu/~wrogers/phonemes/phono/stop.htm.

[17] "Nasal speech sound," 2015. accessed 2015-10-28. Available: http://www.britannica.com/topic/nasal-speech-sound.

[18] P. Ladefoged and I. Maddieson, "The sounds of the world's languages," *Language*, vol. 74, no. 2, pp. 374–376, 1998.

[19] "Maxillary lateral incisor," 2015. accessed 2015-10-28. Available: https://en.wikipedia.org/wiki/Maxillary_lateral_incisor.

[20] "Syllable, stress & accent," 2015. accessed 2015-10-28. Available: http://hubblesite.org/reference_desk/faq/answer.php.id=73&cat=light.

[21] P. Roach and E. Phonetics, "Phonology: A practical course," *Cambridge UP Cambridge*, 2000.

[22] "What is the relationship between wavelength, frequency and energy?," 2015. accessed 2015-10-28. Available: http://www.personal.rdg.ac.uk/~llsroach/phon2/mitko/syllable.htm.

[23] J. Laver, *Principles of phonetics*. Cambridge University Press, 1994.

[24] S. J. Orfanidis, *Introduction to signal processing*. Prentice-Hall, Inc., 1995.

[25] L. C. Evans, "Partial differential equations and monge-kantorovich mass transfer," *Current developments in mathematics*, pp. 65–126, 1997.

[26] L. R. Rabiner and B. Gold, "Theory and application of digital signal processing," *Englewood Cliffs, NJ, Prentice-Hall, Inc., 1975. 777 p.*, vol. 1, 1975.

[27] M. Weik, *Communications standard dictionary*. Springer Science & Business Media, 2012.

[28] "Sampling and quantization," 2015. accessed 2015-11-08. Available: https://courses.engr.illinois.edu/ece110/content/courseNotes/files/?samplingAndQuantization.

[29] "Digital signals - sampling and quantization," 2015. accessed 2015-11-08. Available: http://rs-met.com/documents/tutorials/DigitalSignals.pdf.

[30] "Windowing signal processing," 2015. accessed 2015-11-08. Available: http://www.cs.tut.fi/kurssit/SGN-4010/ikkunointi_en.pdf.

[31] M. Forsberg, "Why is speech recognition difficult," *Chalmers University of Technology*, 2003.

[32] M. Gales, "Discriminative models for speech recognition," in *Information Theory and Applications Workshop, 2007*, pp. 170–176, IEEE, 2007.

[33] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ieee assp magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[34] "Definition of hidden markov model," 2015. accessed 2015-09-08. Available: http://jedlik.phy.bme.hu/~gerjanos/HMM/node4.html.

[35] H. A. Bourlard and N. Morgan, "Hidden markov models," in *Connectionist Speech Recognition*, pp. 27–58, Springer, 1994.

[36] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1, pp. 19–41, 2000.

[37] D. Reynolds, R. C. Rose, *et al.*, "Robust text-independent speaker identification using gaussian mixture speaker models," *Speech and Audio Processing, IEEE Transactions on*, vol. 3, no. 1, pp. 72–83, 1995.

[38] P. Boersma *et al.*, "Praat, a system for doing phonetics by computer," *Glot international*, vol. 5, no. 9/10, pp. 341–345, 2002.

[39] J. Yuan and M. Liberman, "Speaker identification on the scotus corpus," *Journal of the Acoustical Society of America*, vol. 123, no. 5, p. 3878, 2008.

[40] H. Hermansky, "Perceptual linear predictive (plp) analysis of speech," *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[41] P. Harrison, *Variability of formant measurements*. Department of language and linguistic science, 2004.

[42] W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, and J. Woelfel, "Sphinx-4: A flexible open source framework for speech recognition," 2004.

[43] B. Prica and S. Ilić, "Recognition of vowels in continuous speech by using formants," *Facta universitatis-series: Electronics and Energetics*, vol. 23, no. 3, pp. 379–393, 2010.

[44] "Word error rate (wer) and word recognition rate (wrr) with python," 2016. accessed 2016-01-08. Available: http://progfruits.blogspot.com/2014/02/word-error-rate-wer-and-word.html.

[45] "Bayesian information criterion," 2016. accessed 2016-01-08. Available: http://stanfordphd.com/BIC.html.

[46] S. Chen and P. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the bayesian information criterion," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, vol. 8, pp. 127–132, Virginia, USA, 1998.

[47] R. Das, J. Izak, J. Yuan, and M. Liberman, "Forced alignment under adverse conditions," *Unpublished manuscript*, 2010.

[48] "Discrete fourier transform (dft)," 2015. accessed 2015-11-08. Available: http://www.mathworks.com/help/matlab/math/discrete-fourier-transform-dft.html.

[49] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning.* MIT press, 2012.

[50] Z. Ghahramani, "Unsupervised learning," in *Advanced Lectures on Machine Learning*, pp. 72–112, Springer, 2004.

[51] A. P. Engelbrecht, *Computational intelligence: an introduction.* John Wiley & Sons, 2007.

[52] J. Archibald, S. Roy, S. Harmel, K. Jesney, E. Dewey, S. Moisik, and P. Lessard, *A review of the literature on second language learning.* ERIC, 2006.

[53] A. Gilakjani, S. Ahmadi, and M. Ahmadi, "Why is pronunciation so difficult to learn?," *English Language Teaching*, vol. 4, no. 3, p. p74, 2011.

[54] D. Edge, K.-Y. Cheng, M. Whitney, Y. Qian, Z. Yan, and F. Soong, "Tip tap tones: mobile microtraining of mandarin sounds," in *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, pp. 427–430, ACM, 2012.

[55] P. Medgyes, "When the teacher is a non-native speaker," *Teaching English as a second or foreign language*, vol. 3, pp. 429–442, 2001.

[56] A. Head, Y. Xu, and J. Wang, "Tonewars: Connecting language learners and native speakers through collaborative mobile games," in *Intelligent Tutoring Systems*, pp. 368–377, Springer, 2014.

Appendices

Figure 1: BIC results for GMM selection

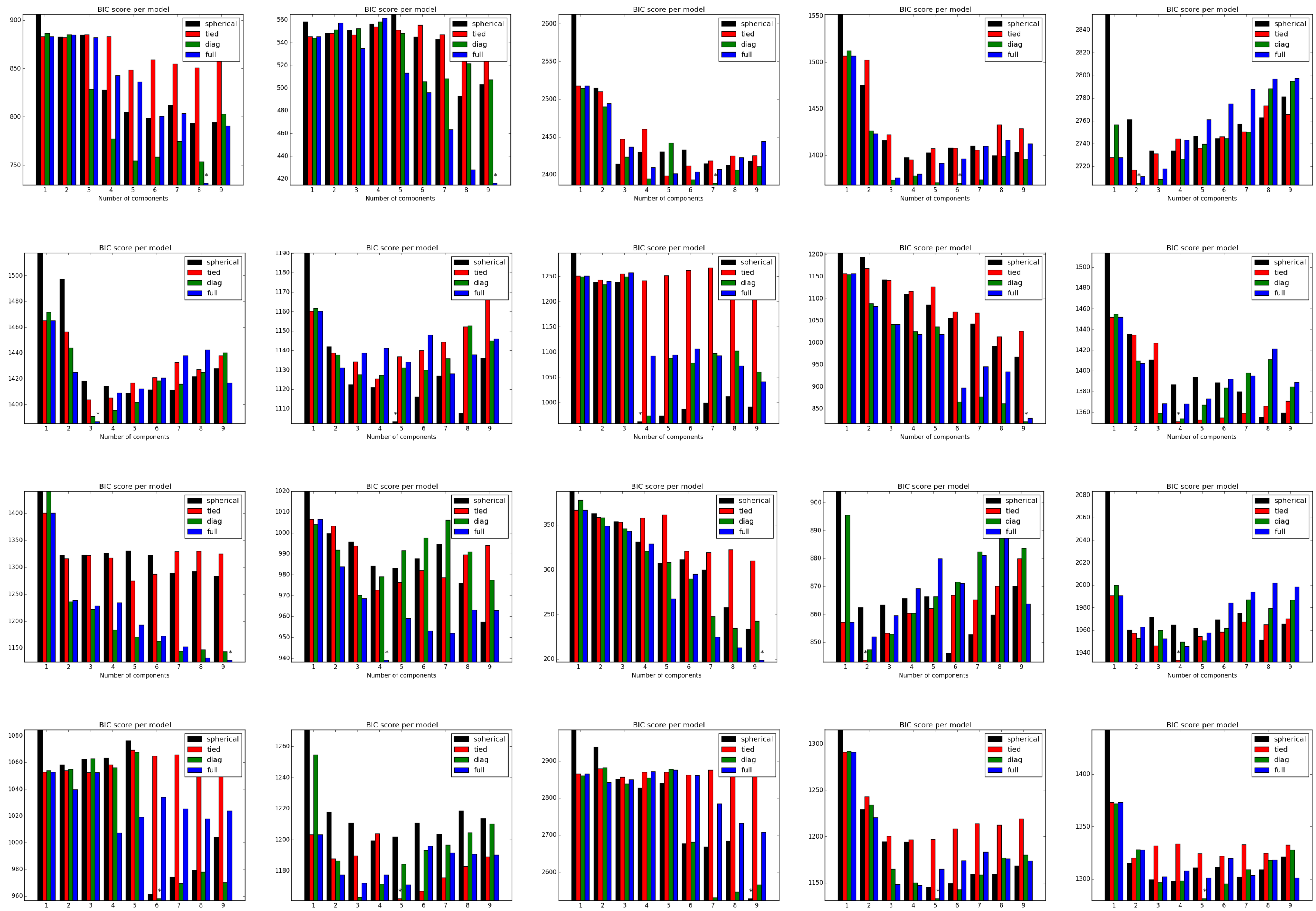
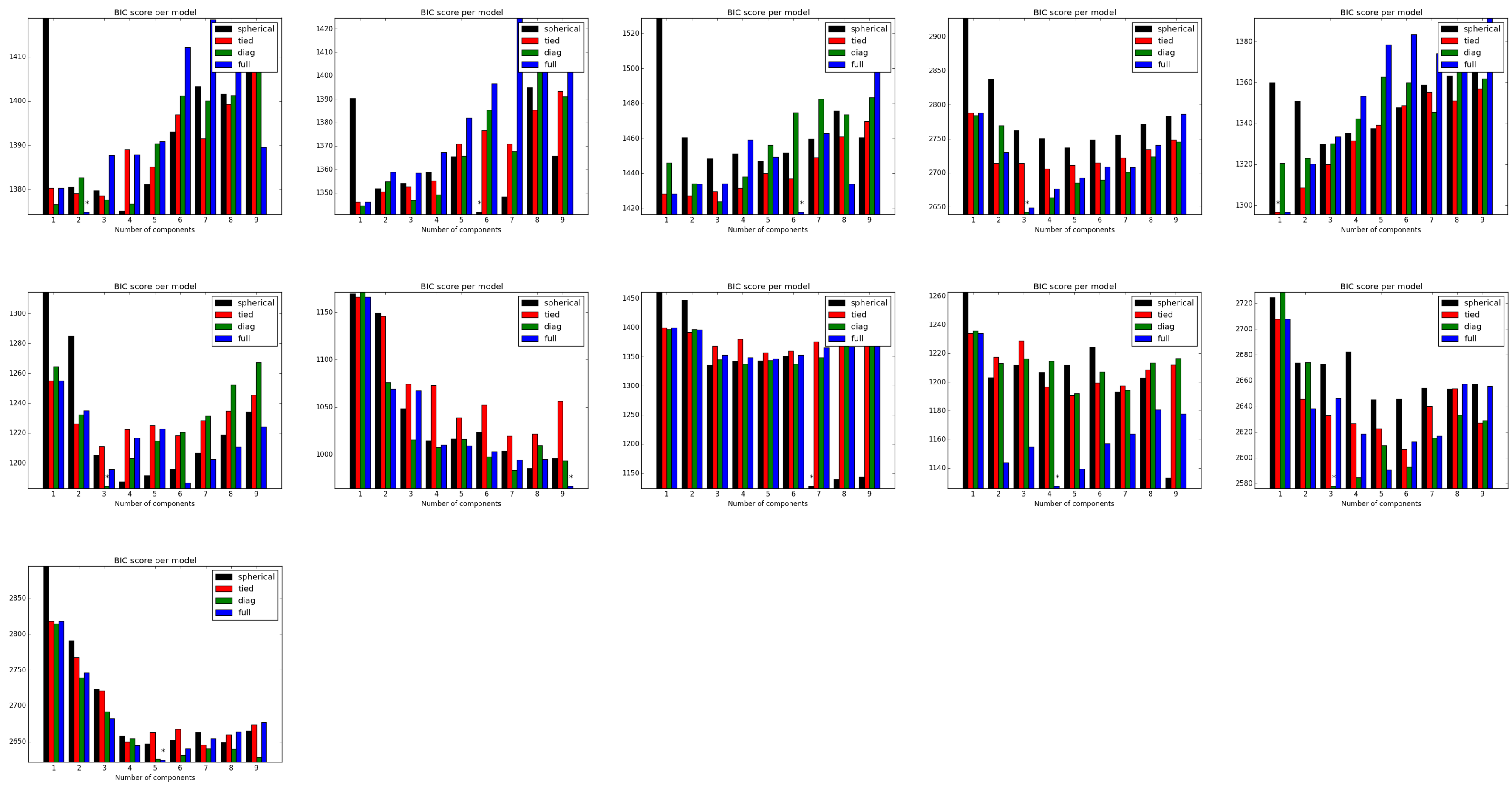


Figure 2: BIC results for GMM selection



PARLA: mobile application for English pronunciation

Survey regarding the application

* Required

1. **Gender** *
   Mark only one oval.
   - Female
   - Male

2. **Age** *
   Mark only one oval.
   - 17 or younger
   - 18-23
   - 24-29
   - 30-35
   - 36-41
   - 41 or older
   - Other: ______________________

3. **Occupation** *
   Mark only one oval.
   - Student
   - Teacher
   - Worker
   - Other: ______________________

4. **Native language** *
   Type the name of your native language

   ____

5. **Improving the pronunciation** *
Rate your interest in improving the English pronunciation
Mark only one oval.

	1	2	3	4	5	
Not interested	◯	◯	◯	◯	◯	Very interested

6. **Languages** *
Rate your interest in having the same application for other languages
Mark only one oval.

	1	2	3	4	5	
Not interested	◯	◯	◯	◯	◯	Very interested

7. **Using a smartphone** *
Rate your interest in using a smartphone for language pronunciation improvement
Mark only one oval.

	1	2	3	4	5	
Not interested	◯	◯	◯	◯	◯	Very interested

8. **Visual feedback** *
Rate your interest in having visual feedback for improving pronunciation
Mark only one oval.

	1	2	3	4	5	
Not interested	◯	◯	◯	◯	◯	Very interested

9. **No teacher** *
Rate your interest in having immediate feedback based only on random native speakers and not a single qualified teacher in which can take sometime before to give you some feedback.
Mark only one oval.

	1	2	3	4	5	
Not interested	◯	◯	◯	◯	◯	Very interested

10. **Period of usage** *
In which part of the day have you used the application the most ?
Mark only one oval.

- Morning
- Afternoon
- Evening
- Before bed

11. **Time of usage** *
For how long have you used the application in total ?
Mark only one oval.

- 10 - 30 minutes
- 3 - 6 hours
- more than 6 hours
- 30m - 1h
- 1 - 3 hours
- less then 10 minutes

12. **Liked application** *
Did you enjoy using the application ?
Mark only one oval.

	1	2	3	4	5	
Not at all						Very much

13. **Usage of application** *
Would you use this application despite this limited amount of time ?
Mark only one oval.

	1	2	3	4	5	
Not at all						Very much

14. **Difficulty of usage** *
How difficult was to use the application ?
Mark only one oval.

	1	2	3	4	5	
Very hard						Very easy

15. **Understanding the main page** *
How difficult was to use the main page (the one with the big blue button) ?
Mark only one oval.

	1	2	3	4	5	
Very hard	◯	◯	◯	◯	◯	Very easy

16. **Understanding the critical listening page** *
How difficult was to understand the critical listening page ?
Mark only one oval.

	1	2	3	4	5	
Very hard	◯	◯	◯	◯	◯	Very easy

17. **Understanding the feedback** *
How difficult was to understand the feedback in general ?
Mark only one oval.

	1	2	3	4	5	
Very hard	◯	◯	◯	◯	◯	Very easy

18. **Understanding the stress** *
How difficult was to understand the phoneme-stress in the feedback page ?
Mark only one oval.

	1	2	3	4	5	
Very hard	◯	◯	◯	◯	◯	Very easy

19. **Understanding the pitch trend** *
How difficult was to understand the pitch chart (the first chart from top to bottom) ?
Mark only one oval.

	1	2	3	4	5	
Very hard	◯	◯	◯	◯	◯	Very easy

20. **Understanding vowels chart** *
How difficult was to understand the vowels chart (the second from top to bottom) ?
Mark only one oval.

	1	2	3	4	5	
Very hard	◯	◯	◯	◯	◯	Very easy

21. **Understanding the history page** *
How difficult was to understand the charts in the history page ?
Mark only one oval.

	1	2	3	4	5	
Very hard	⬭	⬭	⬭	⬭	⬭	Very easy

22. **Pronunciation improved** *
Did the application help you to improve the pronunciation ?
Mark only one oval.

	1	2	3	4	5	
Not at all	⬭	⬭	⬭	⬭	⬭	A lot

23. **Critical listening** *
Did the critical listening page help you to improve the pronunciation ?
Mark only one oval.

	1	2	3	4	5	
Not at all	⬭	⬭	⬭	⬭	⬭	A lot

24. **Self listening** *
Did the self listening page help you to improve your pronunciation ?
Mark only one oval.

	1	2	3	4	5	
Not at all	⬭	⬭	⬭	⬭	⬭	A lot

25. **Feedback page** *
Did the feedback page help you to improve the pronunciation ?
Mark only one oval.

	1	2	3	4	5	
Not at all	⬭	⬭	⬭	⬭	⬭	A lot

26. **History page** *
Did the history page help you to improve the pronunciation ?
Mark only one oval.

	1	2	3	4	5	
Not at all	⬭	⬭	⬭	⬭	⬭	A lot

Viagiar descànta, ma chi parte mona torna mona
Old Venetian aphorism