UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

MODELO ACÚSTICO DE LÍNGUA INGLESA FALADA POR PORTUGUESES

Carla Alexandra Coelho Simões

Mestrado em Engenharia Informática

2007

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

MODELO ACÚSTICO DE LÍNGUA INGLESA FALADA POR PORTUGUESES

Carla Alexandra Coelho Simões

Projecto orientado pelo Prof. Dr Carlos Teixeira
e co-orientado por Prof. Dr Miguel Salles Dias

Mestrado em Engenharia Informática

2007

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

ACOUSTIC MODEL OF ENGLISH LANGUAGE SPOKEN BY PORTUGUESE SPEAKERS

Carla Alexandra Coelho Simões

Project advisers: Prof. Dr Carlos Teixeira
and Prof. Dr Miguel Salles Dias

Master of Science in Computer Science Engineering

2007

FACULDADE · DE · CIÊNCIAS UNIVERSIDADE · DE · LISBOA

Declaração

Carla Alexandra Coelho Simões, aluno nº28131 da Faculdade de Ciências da Universidade de Lisboa, declara ceder os seus direitos de cópia sobre o seu Relatório de Projecto em Engenharia Informática, intitulado "Modelo Acústico de Língua Inglesa Falada por Portugueses", realizado no ano lectivo de 2006/2007 à Faculdade de Ciências da Universidade de Lisboa para o efeito de arquivo e consulta nas suas bibliotecas e publicação do mesmo em formato electrónico na Internet.

FCUL, de de 2007

Carlos Jorge da Conceição Teixeira, supervisor do projecto de *Carla Alexandra Coelho Simões*, aluno da Faculdade de Ciências da Universidade de Lisboa, declara concordar com a divulgação do Relatório do Projecto em Engenharia Informática, intitulado "Modelo Acústico de Língua Inglesa Falada por Portugueses".

Lisboa, de de 2007

Resumo

No contexto do reconhecimento robusto de fala baseado em modelos de Markov não observáveis (do inglês *Hidden Markov Models* - HMMs) este trabalho descreve algumas metodologias e experiências tendo em vista o reconhecimento de oradores estrangeiros.

Quando falamos em Reconhecimento de Fala falamos obrigatoriamente em Modelos Acústicos também. Os modelos acústicos reflectem a maneira como pronunciamos/articulamos uma língua, modelando a sequência de sons emitidos aquando da fala. Essa modelação assenta em segmentos de fala mínimos, os fones, para os quais existe um conjunto de símbolos/alfabetos que representam a sua pronunciação. É no campo da fonética articulatória e acústica que se estuda a representação desses símbolos, sua articulação e pronunciação.

Conseguimos descrever palavras analisando as unidades que as constituem, os fones. Um reconhecedor de fala interpreta o sinal de entrada, a fala, como uma sequência de símbolos codificados. Para isso, o sinal é fragmentado em observações de sensivelmente 10 milissegundos cada, reduzindo assim o factor de análise ao intervalo de tempo onde as características de um segmento de som não variam.

Os modelos acústicos dão-nos uma noção sobre a probabilidade de uma determinada observação corresponder a uma determinada entidade. É, portanto, através de modelos sobre as entidades do vocabulário a reconhecer que é possível voltar a juntar esses fragmentos de som.

Os modelos desenvolvidos neste trabalho são baseados em HMMs. Chamam-se assim por se fundamentarem nas cadeias de Markov (1856 - 1922): sequências de estados onde cada estado é condicionado pelo seu anterior. Localizando esta abordagem no nosso domínio, há que construir um conjunto de modelos - um para cada classe de sons a reconhecer - que serão treinados por dados de treino. Os dados são ficheiros áudio e respectivas transcrições (ao nível da palavra) de modo a que seja possível decompor essa transcrição em fones e alinhá-la a cada som do ficheiro áudio correspondente. Usando um modelo de estados, onde cada estado representa uma observação ou segmento de fala descrita, os dados vão-se reagrupando de maneira a criar modelos estatísticos, cada vez mais fidedignos, que consistam em representações das entidades da fala de uma determinada língua.

O reconhecimento por parte de oradores estrangeiros com pronuncias diferentes da língua para qual o reconhecedor foi concebido, pode ser um grande problema para precisão de um reconhecedor. Esta variação pode ser ainda mais problemática que a variação dialectal de uma determinada língua, isto porque depende do conhecimento que cada orador têm relativamente à língua estrangeira.

Usando para uma pequena quantidade áudio de oradores estrangeiros para o treino de novos modelos acústicos, foram efectuadas diversas experiências usando corpora de Portugueses a falar Inglês, de Português Europeu e de Inglês.

Inicialmente foi explorado o comportamento, separadamente, dos modelos de Ingleses nativos e Portugueses nativos, quando testados com os corpora de teste (teste com nativos e teste com não nativos). De seguida foi treinado um outro modelo usando em simultâneo como corpus de treino, o áudio de Portugueses a falar Inglês e o de Ingleses nativos.

Uma outra experiência levada a cabo teve em conta o uso de técnicas de adaptação, tal como a técnica MLLR, do inglês Maximum Likelihood Linear Regression. Esta última permite a adaptação de uma determinada característica do orador, neste caso o sotaque estrangeiro, a um determinado modelo inicial. Com uma pequena quantidade de dados representando a característica que se quer modelar, esta técnica calcula um conjunto de transformações que serão aplicadas ao modelo que se quer adaptar.

Foi também explorado o campo da modelação fonética onde estudou-se como é que o orador estrangeiro pronuncia a língua estrangeira, neste caso um Português a falar Inglês. Este estudo foi feito com a ajuda de um linguista, o qual definiu um conjunto de fones, resultado do mapeamento do inventário de fones do Inglês para o Português, que representam o Inglês falado por Portugueses de um determinado grupo de prestígio. Dada a grande variabilidade de pronúncias teve de se definir este grupo tendo em conta o nível de literacia dos oradores. Este estudo foi posteriormente usado na criação de um novo modelo treinado com os corpora de Portugueses a falar Inglês e de Portugueses nativos. Desta forma representamos um reconhecedor de Português nativo onde o reconhecimento de termos ingleses é possível.

Tendo em conta a temática do reconhecimento de fala este projecto focou também a recolha de corpora para português europeu e a compilação de um léxico de Português europeu. Na área de aquisição de corpora o autor esteve envolvido na extracção e preparação dos dados de fala telefónica, para posterior treino de novos modelos acústicos de português europeu.

Para compilação do léxico de português europeu usou-se um método incremental semi-automático. Este método consistiu em gerar automaticamente a pronunciação de grupos de 10 mil palavras, sendo cada grupo revisto e corrigido por um linguista. Cada grupo de palavras revistas era posteriormente usado para melhorar as regras de geração automática de pronunciações.

PALAVRAS-CHAVE: reconhecimento automático de fala, sotaque estrangeiro, modelos de Markov escondidos, transcrição fonética.

Abstract

The tremendous growth of technology has increased the need of integration of spoken language technologies into our daily applications, providing an easy and natural access to information. These applications are of different nature with different user's interfaces. Besides voice enabled Internet portals or tourist information systems, automatic speech recognition systems can be used in home user's experiences where TV and other appliances could be voice controlled, discarding keyboards or mouse interfaces, or in mobile phones and palm-sized computers for a hands-free and eyes-free manipulation.

The development of these systems causes several known difficulties. One of them concerns the recognizer accuracy on dealing with non-native speakers with different phonetic pronunciations of a given language. The non-native accent can be more problematic than a dialect variation on the language. This mismatch depends on the individual speaking proficiency and speaker's mother tongue. Consequently, when the speaker's native language is not the same as the one that was used to train the recognizer, there is a considerable loss in recognition performance.

In this thesis, we examine the problem of non-native speech in a speaker-independent and large-vocabulary recognizer in which a small amount of non-native data was used for training. Several experiments were performed using Hidden Markov models, trained with speech corpora containing European Portuguese native speakers, English native speakers and English spoken by European Portuguese native speakers.

Initially it was explored the behaviour of an English native model and non-native English speakers' model. Then using different corpus weights for the English native speakers and English spoken by Portuguese speakers it was trained a model as a pool of accents. Through adaptation techniques it was used the Maximum Likelihood Linear Regression method. It was also explored how European Portuguese speakers pronounce English language studying the correspondences between the phone sets of the foreign and target languages. The result was a new phone set, consequence of the mapping between the English and the Portuguese phone sets. Then a new model was trained with English Spoken by Portuguese speakers' data and Portuguese native data.

Concerning the speech recognition subject this work has other two purposes: collecting Portuguese corpora and supporting the compilation of a Portuguese lexicon, adopting some methods and algorithms to generate automatic phonetic pronunciations. The collected corpora was processed in order to train acoustic models to be used in the Exchange 2007 domain, namely in Outlook Voice Access.

KEYWORDS: automatic speech recognition, foreign accent, hidden Markov models, phonetic transcription.

Contents

Figures List

Tables List

Chapter 1

Introduction

Speaking is the major way of communication among human beings. This gives us the ability of expressing ideas, feelings or thoughts as well as changing different opinions about different ways of seeing and living the world.

In a world we define as a *global village* [1] where people interact and live in a global scale, technology has grown in a sense of supporting a new way of transmitting information allowing users from all over the world to connect with each other. We are attending the creation of new easier ways of interaction where automatic systems supporting spoken language technologies can be very handy for our daily applications, providing an easy and natural access to information. These applications are from different nature with different human-computer interfaces. Besides voice enabled Internet portals or tourist information systems, Automatic Speech Recognition (ASR) systems can be used in home user's experiences where TV and other appliances can be voice controlled, discarding keyboards or mouse interfaces, or in mobile phones and palm-sized computers for a hands-busy and eyes-busy manipulation. An important application area is telephony, where speech recognition is often used for entering digits, recognizing some simple commands for call acceptance, finding out airplane and train information or explores call-routing capabilities. ASR systems can be also applied to dictation use, in some fields such as human-computer interfaces for people with some disability on typing.

When we think of the potential of such systems we must deal with the language-dependency problem. This includes the non-native speaker's speech with different phonetic pronunciations from those of the native speakers' language. The non-native accent can be more problematic than a dialect variation on the language, because there is a larger variation among speakers of the same non-native accent than among speakers

[1] "Global village is a term coined by Wyndham Lewis in his book *America and Cosmic Man* (1948). However, Herbert Marshall McLuhan also wrote about this term in his book *The Gutenberg Galaxy: The Making of Typographic Man* (1962). His book describes how electronic mass media collapse space and time barriers in human communication enabling people to interact and live on a global scale. In this sense, the globe has been turned into a village by the electronic mass media (…) today the global village is mostly used as a metaphor to describe the Internet and World Wide Web." (in Wikipedia)

of the same dialect. This mismatch depends on the individual speaking proficiency and mother's speaker tongue. Consequently, recognition accuracy has been observed to be considerably lower for non-native speakers of the target language than for natives ones [3] [7] [9].

In this work we apply a number of acoustic modelling techniques to compare their performance on non-native speech recognition. All the experiments were based on Hidden Markov Models (HMMs) using cross-word triphone based models for command & control applications. The case of study is focused on English language spoken by European Portuguese (EP) speakers.

1.1 Speech Recognition

In the context of human-computer interfaces tasks are often better solved with visual or pointing interfaces, speech can play a better role than keyboards or other devices. The scientific community has been researching and developing new ways of accurately recognize speech, still spoken language understanding is a difficult task, today the state-of-art systems cannot match human's performance.

Speech recognition is the conversion of an acoustic signal to understandable words. This process is performed by a software component known as the *speech recognition engine*. The primary function of the speech recognition engine is to process spoken input and translate it into text to be understandable for an application. If the application is a *command & control* application it should interpret the result of the recognition as a command. An example is when the caller says "turn off the radio" the application fulfil the order. If the application also supports *dictation* it would not interpret the caller's command, but it will recognize the text simply as a text which means that will return the text "turn off the radio" after the caller's order.

A speech based-application e.g. voice dialler, is responsible for loading the recognition engine to initialize the speech signal processing. The engine interprets the signal as a sequence of encoded symbols (Figure 1.1), and it is important to understand that the audio stream contains not only the speech data but also background noise. Regarding the distortion that this noise may cause to the speech signal, the engine is split into *Front-End* and *Decoder*.

The front-end part analyzes the continual sound waves and converts into a sequence of equally spaced discrete parameter vectors, also called *feature vectors*. This sequence of parameter vectors is an exact representation of the speech waveform, each one with a typically observation of 10 milliseconds. At this point the speech waveform can be regarded as being stationary, where the feature vectors reflect the input sounds as speech rather than noise. The way this part of the front-end works is to listen to certain patterns at certain sound frequencies. Human speech is only emitted at certain frequencies and so the noises which fall outside these frequencies indicate that nothing is being spoken at a particular point.

Once the speech data is in the proper format (feature vectors), the decoder searches for the best match. It does this by taking into consideration the words and phrases it knows about, along with the knowledge provided in the form of an *acoustic model*. The acoustic model gives the likelihood for a given feature vector as produced by a particular sound (Chapter 2). When it identifies the most likely match for what was said, it outputs a sequence of symbols (e.g. words).

During this process the valid words and phrases that the engine knows are specified in a grammar which controls the interaction between the user and the computer (see 1.1.3). Figure 1.1 shows the speech recognition process where a sequence of underlying symbols are recognized by comparing frames of the audio input (feature vectors) to the models stored in an acoustic model.

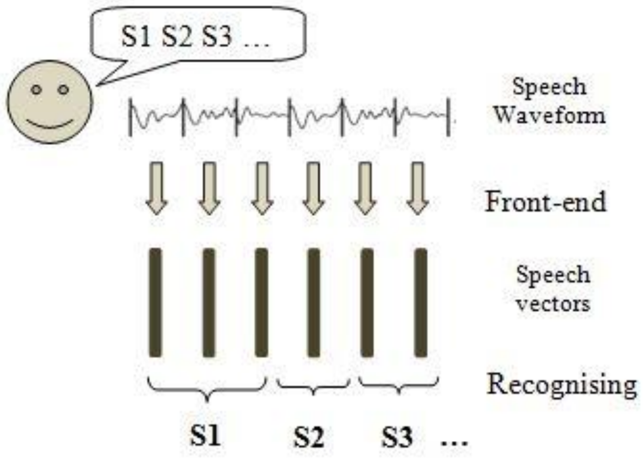


Figure 1.1 Encoding / Decoding process

The performance of a speech recognition system is measurable, normally in terms of its accuracy. This issue is a critical factor in determining the practical value of a speech-recognition application whose tasks are often classified according to its requirements in handling specific or nonspecific speakers, in accepting only isolated or fluent speech as well as the influence of large variations in the speech waveform due to speakers' variability, mood, environment, etc (see 1.1.1). The accuracy is also tied to grammar designs, which means that utterances, which are not contained in the grammar, will not be recognized.

1.1.1 Variability in the Speech Signal

Speech recognition systems can be influenced by several parameters, which determine the accuracy and robustness of speech recognition algorithms. The following sections summarize the major factors involved.

Context Variability

The comprehension between people requires the knowledge of word meanings and communication context. Different words with different meanings when applied in some contexts may have the same phonetic resolution, as we can see in the following example:

You might be right, please write to Mr. Wright explaining the situation…

In addition to the context variability at word level we can find it at phonetic level too. For example the acoustic realization of phoneme */ee/* for words *feet* and *real* depends on its left and right context. This problem can be largely increased in terms of the vocabulary size, this means that speech recognition is easier for recognition of limited words, such as *Yes* or *No* detection or sequences of digits, and harder for tasks with large vocabularies (70 0000 words or more).

Fluency

Spontaneous speech is often diffluent, speakers normally pause in the middle of a sentence, speak in fragments, stumble over the words. The recognizers must deal with it, and some constrains can be imposed when using an *isolated-word speech* recognition. The system requires that speakers pause briefly between words, which provide a correct silence context to each word for an ease decoding of speech. The disadvantage is that systems are unnatural to most people.

Continuous speech error rate is considerably higher than isolated speech [10], especially if speakers reflect their emotional states on whispering, shouting, laughing or crying during a conversation. Continuous speech recognition tasks can be described as *read speech*, that is recognizing speech within a human-to-machine conversation (e.g. dictation, speech dialogue systems), or *conversational speech.* The last one comprehends the human-to-human speech recognition for example for transcribing a telephonic conversation.

Speaker Variability

The speech produced by an individual can be completely different from the one of another person. The differences can be categorize as *acoustic* differences which are related to the size and vocal track, and *pronunciation* differences that generally refers to different dialects and accents (geographical distribution) [16]. We can say that speech reflects the physical characteristics of an individual such as age, gender, height, health, dialect, education, personal style as also emotional changes for example speech production in stress conditions [11]. In this context we can classify recognizers as *speaker-dependent* or *speaker-independent* systems. For speaker-independent speech recognition we must have a large amount of different speakers to build a combined model [8], which in practice is difficult to get full coverage of all required accents.

A speaker-dependent system can perform better than a speaker-independent one because there are no speaker variations within the same model. The disadvantage of these systems is related with the collection of specific speaker data, which may be impractical for applications where the use of speech is getting importance for people daily tasks. The evolution of technology on the use of speech claims for applications with speaker-independent type that are able to recognize speech of people whose speech system has never been trained with.

Environment Variability

The world we live in is full of sounds of varying loudness of different sources. The speech recognition system performance can be affected at different noise levels. It often depends when the interaction between certain devices with embedded speech recognizer takes place. On using these devices in our office we may have people speaking in the background or someone can slam the door. In mobile devices the capture of the speech signal can be deficient because the speaker moves around or is driving and the car engine is too noisy. In addition to the environmental noises the system accuracy may also be influenced by speakers' noises (e.g. noisy pauses, lip smacks) as well as the type and placement of microphone.

Despite the progress in using different methods to solve this problem, the environment variability is still a challenge for nowadays' systems. One of those methods to outline the problem and suppress a noise channel is to use the *spectral suppression* [19] another alternative is to use one or more microphones whenever one is to capture the speech signal and the others to capture the surrounding noise, this technique is called *adaptive noise cancelling* [21].

1.1.2 Speech Recognition Methods

In terms of the current technology the major speech recognition systems are generally based on two main methodologies: the *Dynamic Time Warping* (DTW) and the *Hidden Markov Models.*

The DTW is an algorithm for measuring similarity between two speech sequences which may vary in time [22]. The sequences are warped non-linearly to match each other. Speech recognition is simple to implement and effective for small-vocabulary speech recognition. For a large amount of data the HMM is a much better alternative since it is required a higher training token to characterize the variation among different utterances.

Modern speech recognition systems are generally based on HMMs [2] [24]. This is a statistical model where the speech signal could be viewed as a short-time stationary signal. The sequence of observed speech vectors corresponding to each word is generated by a Markov model. A Markov Model is a finite state machine in which each state is influenced by its previous one. The detailed signal information supplied by the analysis of the speech vectors is useful to outline some factors that spoil the speech recognition systems performance. The analysis is made at certain frequencies and patterns levels (human speech). This method is explained with more detail in Chapter 2.

As a recent approach in acoustic modelling, the use of *Neural-Networks* has been applied with success. They are efficient in solving complicated recognition tasks for short and isolated speech units. When it comes to large vocabularies [41] [42] HMMs reveal a better performance. There are also *hybrid systems* that use part of this methodology with the HMMs [23].

1.1.3 Components for Speech-Based Applications

Speech based applications can be used in different subjects such as applications as command & control, data entry, and document preparation (dictation). After training an acoustic model, the speech recognition engine is ready to be used. For training these models it is necessary a great collection of audio data that fulfils the requirements of the speech-based application in cause and a phonetic dictionary with all the words phonetically transcribed (more details in Chapter 2).

The audio characteristics normally reflect the telephony, desktop, home or mobile environment where the applications are built. One of the most important is the bandwidth of the audio stream. An input speech signal is first digitalized, which requires discrete time *sampling* and *quantization* of the waveform. A signal is sampled by measuring its amplitude in a particular time. Typically sampling rates are 8 kHz for telephonic platform and 16 kHz for desktop. Quantization refers to store real-valued numbers such as the amplitude of the signal into integers, either 8-bit or 16-bit.

The Language Pack, fundamental for this type of applications within Windows Operating System (OS), includes the speech recognition engine and *Text-to-Speech Engine* (TTS). The second is a speech synthesizer and as the name suggests, it converts text into artificial human speech. There are different technologies used to generate artificial speech, relating to the different purposes of the synthesis – the *naturalness* and the *intelligibility* of speech. The *concatenative synthesis* benefits the natural sounding synthesized speech, because it concatenates segments of human recorded speech and consequently the *formant synthesis* does not use any kind of human speech samples - the output is built using acoustic models. The *articulatory synthesis* uses physical models of speech production. These models represent the human vocal tract where the motions of articulators, the distributions of volume velocity and sound pressure in the lungs, larynx, vocal and nasal tracts, are exploited. This may be the best way to synthesize speech but the existing technology in articulatory synthesis does not generate speech quality comparable to formant or concatenative systems.

Even though the formant synthesis avoids the acoustic glitches derived from the variations of segments in the concatenative synthesis, it normally generates unnatural speech, since it has the control of the entire output speech components such as the sentences pronunciation. The contatenative systems relies on high quality voice databases which covers the widest variety of units and phonetic contexts for a certain language – rich and balanced sentences according to the number of words, syllables, diphones, triphones, etc. In order to improve the synthesis process according to its naturalness, the concept of *prosody***,** should be included [6] [39]. Prosody determines how a sentence is spoken in terms of melody, phrasing, rhythm, accent locations and emotions.

The Speech Application Programming Interface (SAPI) is a Microsoft API that provides a communication between the application and the Speech Recognition and Synthesis engines. It is also intended for the easy development of Speech enabled applications (e.g. Voice Command or Exchange Voice Access). Although the example focuses the Microsoft API, there are other solutions in the market such as the Java Speech API, from Sun Microsystems.

A speech-based application is responsible for loading the engine and for requesting actions/information from it. The application communicates with the engine via the SAPI interface and together with an activated grammar the engine will begin processing the audio input. The grammars contain the list of everything a user can say. It can be seen as the model of all the allowed utterances of the engine. The grammar can be any size and represents a list of valid words/sentences, which improves the recognition accuracy by restricting and indicating to the engine what should be expected. The valid sentences need to be carefully chosen, considering the application nature. For example, command and control applications make use of *Context-Free Grammars* (CFG), in order to establish rules that are able to generate a set of words and combinations to build all type of allowed sentences. In 2.6.2 there are more details about grammars formats and which was useful to the project.

Figure 1.2 represents the different components and respective interactions for constructing based-speech applications.

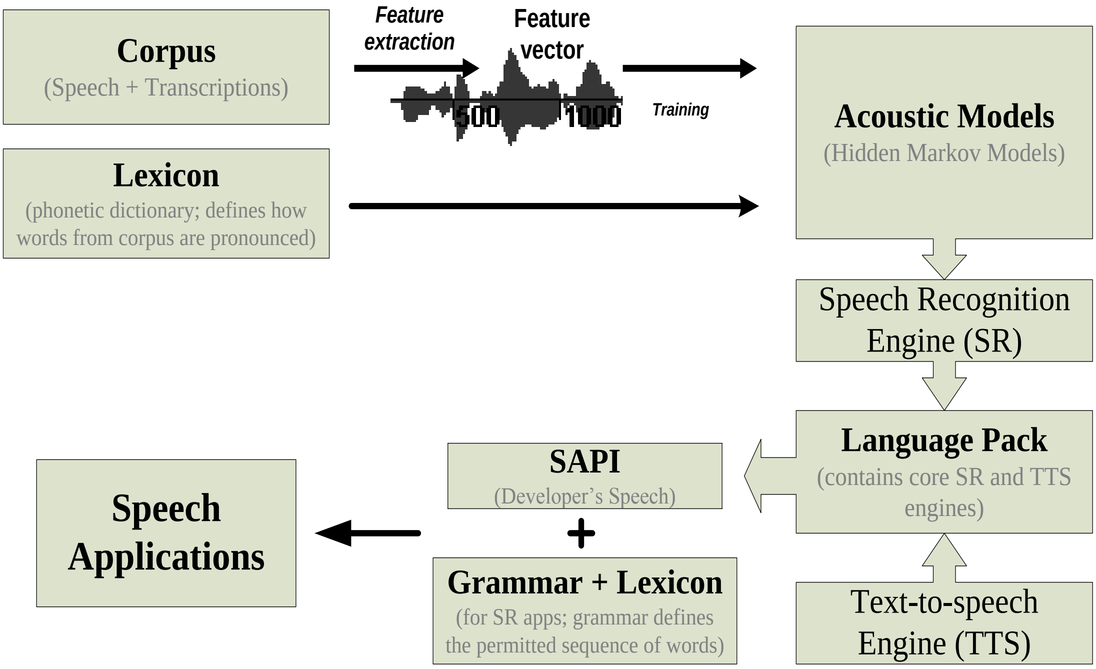

Corpus
(Speech + Transcriptions)
Lexicon
(phonetic dictionary; defines how words from corpus are pronounced)
Feature extraction
Feature vector
500
1000
Training
Acoustic Models
(Hidden Markov Models)
Speech Recognition Engine (SR)
Language Pack
(contains core SR and TTS engines)
Text-to-speech Engine (TTS)
SAPI
(Developer's Speech)
+
Grammar + Lexicon
(for SR apps; grammar defines the permitted sequence of words)
Speech Applications

Figure 1.2 Components of speech-based applications

1.2 Related Work

It is clear that the presence of pronunciation variation within speakers' variability may cause errors in ASR. Modelling pronunciation variation is seen as one of the main research areas related to accent issues and it is a possible way of improving the performance of current systems.

Normally modelling pronunciation methods are categorized according to the source from which information on pronunciation variation will be retrieved and how this information is used for representing it in a more abstract and compact *formalization* or just for *enumerating* it [43]. Regarding this a distinction can be made between *data-driven* vs. *knowledge-based* methods. In data-driven methods the information is mainly obtained from the acoustic signals and derived transcriptions (data), one example of it are the statistical models known as HMMS. The formalization in this method uses phonetic aligned information as a result of the alignment of transcriptions with the respective acoustic signals. An alternative is to enumerate all the pronunciations variants within a transcription and then to add them to the language lexicon. Nevertheless, knowledge-based approach information on pronunciation variation can be a formalized representation in terms of rules, obtained from linguistic studies, or enumerated information in terms of pronunciations forms, as in pronunciations dictionaries.

Pronunciation variations such as non-native speakers' accent can be modelled at the level of the acoustic models in order to optimize them. A considerable number of methods and experiments for the treatment of non-native speech recognition have already been proposed by other authors.

Perhaps the simplest idea of addressing the problem is the use of non-native speakers' speech from a target language and training accent-specific acoustic models. This method is not reasonable because it can be very expensive to collect data that comprehends all the speech variability involved. An alternative is to pool non-native training data with the native training set. Research on related accent issues shows better performance when acoustics and pronunciation of a new accent, are taken into account. In Humphries et al. [12] where the addiction of accent-specific pronunciations reduces the error rate by almost 20%, and in Teixeira et al. [3] it is shown an improvement in isolated-word recognition over baseline British-trained models, using several accent-specific or a single model for both non-native and native accents.

Another approach is the use of multiple models [26] [3]. The target is to facilitate the development of speech recognizers for languages that only little training data is available. Generally the phonetic models used in current recognition systems are predominantly language-dependent. This approach aims at creating language-independent acoustic models that can decode speech from a variety of languages at one and at the same time. This method applies standard acoustic models of phonemes where the similarities of sounds between languages are explored [14] [28] [30]. In Kunzmann et al. [28] it was developed a common phonetic alphabet for fifteen languages, handling the different sounds of each language separately while on the other hand, the common phones are shared through languages as much as possible. It can be also applied to the recognition of non-native speech [27], where each model is optimized for a particular accent or class of accents.

An alternative way to minimize the disparity between foreign accents and native accents is to use adaptation techniques applied to acoustic models concerning speakers' accent variability. Although we typically do not have enough data to train on a specific accent or speaker, these techniques work quite well with a small amount of observable data.

The most commonly used model adaptation techniques are the transformation-based adaptation *Maximum Likelihood Linear Regression* (MLLR) [29] and the Bayesian technique *Maximum A Posteriori* (MAP) [32] [33].

As shown in Chapter 3, both MAP and MLLR techniques begin with an appropriate initial model for adaptive modelling of a single speaker or specific speaker's characteristics (e.g. gender, accent). MLLR computes a set of transformations, where one single transformation is applied to all models in a transformation class. More specifically it estimates a set of linear transformations for the context and variance parameters of a Gaussian mixture HMM system. The effect of these transformations is to shift the component means and to alter the variances in the initial system so that each state in the HMM system can be more likely to generate the adaptation data. In MAP adaptation we need a prior knowledge of the model parameter distribution. The model parameters are re-estimated individually requiring more adaptation data to be effective. When larger amounts of adaptation training data become available, MAP begins to perform better than MLLR, due to this detailed update of each component. It is also possible to serialize these two techniques, which means that MLLR method can be combined with MAP. Consequently, we can take advantages of the different properties of both techniques and instead of only a set of compact MLLR transformations for fast adaptation, we can modify model parameters according to the prior information of the models.

The adaptation techniques can be classified into two main classes: *supervised* and *unsupervised* [31]. Supervised techniques are based on the knowledge provided by the adaptation data transcriptions, to supply adapted models which accurately match user's speaking characteristics. On the other hand, unsupervised techniques use only the outcome of the recognizer to guide the model adaptation. They have to deal with the inaccuracy of automatic transcriptions and the selection of information to perform adaptation.

Another possibility is the lexical modelling where several attempts have been made concerning non-native pronunciation. Liu and Fung [25] have obtained an improvement in recognition accuracy when expanding the native lexicon using phonological rules based on the knowledge of the non-native speakers' speech. It can also be included pronunciation variants to the lexicon of the recognizer using acoustic model interpolation [34]. Each model of a native-speech recognizer is interpolated with the same model of a second recognizer which depends on the speaker’s accent. Stefan Steidl et al. [35] consider that acoustic models of native speech are sufficient to adapt the speech recognizer to the way how non-native speakers pronounce the sounds of the target language. The data-driven models of the native acoustic models are interpolated with each other in order to approximate the non-native pronunciation. Teixeira et. al [3] uses a data-driven approach where pronunciation weights are estimated from training data.

Another approach is the training of selective data [44], where training samples of different sources are selected concerning a desired target task and acoustic conditions. The data is weighted by a confidence measure in order to control the influence of outliers. An appliance of such method is selecting utterances of a data pool which are acoustically close to the development data.

1.3 Goals and Overview

After years of research and development, accuracy of ASR systems remains a great challenge for researchers. It is widely known that speaker’s variability affects speech recognition performance (see 1.1.1), particularly the accent variability [16].

Though the recognition of native speech often reaches acceptable levels, when pronunciation diverges from a standard dialect the recognition accuracy is lowered. This includes speakers whose native language is not the same as the recognizer built for - *foreign accent* - and speakers with regional accents also called *dialects*.

Both regional and foreign accent vary in terms of the linguistic proficiency of each person and the way each word is phonetically pronounced. Regional accent can be considered as more homogenous than foreign accent and therefore, such a difference of the standard pronunciation is easier to collect enough data to model it. On the other hand the foreign accent can be more problematic because there is larger number of foreign accents for any given language and the variation among speakers of the same foreign accent is potentially much greater than among speakers of the same regional accent. The main purpose of this study is to explore the non-native English accent using an experimental corpus of English language spoken by European Portuguese speakers [4].

The native language of a non-native speaker also has influence in the pronunciation of a certain language and consequently in the accuracy of a recognizer. This is related with the capacity of reproducing the target language and the way they slightly alter some phoneme features (e.g. aspirated stops can become non aspirated), and adapt unfamiliar sounds to similar/closer ones of their native phoneme inventory [13] [14] [17].

As it was said before variation due to accents decreases the recognition accuracy quite a bit, generally because acoustic models are trained only on speech with standard pronunciation. Hence, Teixeira et al. [3] [4] have identified a drop of 15% in the recognition accuracy on non-native English accents and Tomokiyo [7] reported that recognition performance is 3 to 4 times lower on an experiment with English spoken by Japanese and Spanish. In order to outline this issue a number of acoustic modelling techniques are applied to the studied corpus [4] and compare their performance on non-native speech recognition.

Firstly we explore the behaviour of an English native model when tested with non-native speakers as well as the performance of a model only trained with non-native speakers. HMMs can be improved by retraining on suitable additional data. Regarding this a recognizer has been trained with a pool of accents, using utterances of English native speakers and English spoken by Portuguese speakers.

Furthermore, adaptation techniques such as MLLR, were used. These reduce the variance between an English native model and the adaptation data, which in this case refers to the European Portuguese accent on speaking English language. To fulfil that task a native English speech recognizer is adapted using the non-native training data.

Afterwards the pronunciation adaptation was explored through adequate correspondences between phone sets of the foreign and target languages. Bartkova et al. [14] and Leeuwen and Orr [15] assume that non-native speakers will use dominantly their native phones. As a consequence of this a common phone set was created for mapping the English and the Portuguese phone sets in order to support English words in a Portuguese dialogue system. Thus, the author tried to use bilingual acoustic models that share training data of English and European Portuguese native speakers so that they can do the decoding on non-native speech.

A second purpose of the project is to collect speech corpora within the Auto-attendant project. This project collects telephonic corpora of European Portuguese to be used in the *Exchange* context. In order to achieve this goal some tools have been developed for fetching and validating the collected speech corpora. There was also a participation in another project, named SIP, for collecting speech corpora. This participation involved annotation and validation tasks.

The third purpose was to coordinate a Portuguese lexicon compilation, adopting some methods and algorithms to generate automatic phonetic pronunciations. This compilation was supported by a linguist expert.

With the increase of speech technologies, the need of adjusting existing Microsoft products to the Portuguese language has emerged. The mission of Microsoft Language Development Center (MLDC) [2] proposes the development of speech technology for the Portuguese language in all the variants. This work obeys to that mission where the training of new acoustic models and the learning of its methodology is the central point for the development of new speech-based applications.

The work carried out will be used in Microsoft products that support synthesis and speech recognition such as the *Exchange 2007* Mail server, which introduces a new speech based interaction method called *Outlook Voice Access* (OVA). *Voice Command* for Windows mobile or other client applications for natural speech interaction are examples of alternative usages for the English spoken by Portuguese speakers' model.

1.4 Dissemination

The work in this thesis has originated the following presentations, which reveals the continuing interest of the scientific community on this subject:

Carla Simões; I Microsoft Workshop on Speech Technology; In Microsoft Portuguese Subsidiary, May 2007, Portugal.

C. Simões, C. Teixeira, D. Braga, A. Calado, M. Dias; European Portuguese Accent in Acoustic Models for Non-native English Speakers; In Proc. CIARP, LNCS 4756, pp.734–742, November 2007, Chile.

[2] "This Microsoft Development Center, the first worldwide outside of Redmond dedicated to key Speech and Natural Language developments, is a clear demonstration of Microsoft efforts of stimulating a strong software industry in the EMEA region. To be successful, MLDC must have close relationships with academia, R&D laboratories, companies, government and European institutions. I will continue fostering and building these relationships in order to create more opportunities for language research and development here in Portugal." (Miguel Sales Dias, in www.microsoft.com/portugal/mldc)

The scientific committees of the *XII International Conference Speech and Computer* (SPECOM'2007) and the *International Conference on Native and Non-native Accents of English* (ACCENTS'2007) have also accepted this work as a relevant scientific contribution. However, we have decided to present and publish this work only in the *12th Iberoamerican Congress on Pattern Recognition* (CIARP'07).

1.5 Document Structure

The next chapters are structured as follows:

Chapter 2 HMM-based Acoustic Models

This chapter explains the subjects approached in this project. The methodology of HMMs is explained as well as the used technology for building them describing the several stages of whole training process.

Chapter 3 Comparison of Native and Non-native Models: Acoustic Modelling Experiments

This chapter presents several methods applied in experiments achieved to improve recognition of non-native speakers' speech. The study was based on an experimental corpus of English spoken by European Portuguese speakers.

Chapter 4 Collection of Portuguese Speech Corpora

This chapter talks about performed tasks concerning speech corpora acquisition. It is also given a description to the developed applications, methodologies and studies accomplished within this purpose.

Chapter 5 Conclusion

This chapter exposes to the final comments and conclusions. The future work lines of research are also approached.

1.6 Conclusions

The goal of this chapter was to present some work motivations and scopes. The major problems that speech recognition systems have to face were printed according to the reality of non-native speakers as the focus problem of this work. Some of the methods and how a speech-based application can be developed were also presented. The structure and evolution of this report has been mentioned.

Chapter 2

HMM-based Acoustic Models

In this chapter we introduce the process for *Acoustic Model* training using the HMMs methodology. To accomplish this task it was used a based HTK Toolkit [2] called *Autotrain* [1]. The Autotrain uses HMMs for the Yakima speech decoder [45], the engine that was used during this project.

The HMMs are one of the most important methodologies of statistical models for processing text and speech. The methodology was firstly published by Baum in 1966 [36], but it was only in 1969 that a HMM based speech recognition application was proposed, by Jelinek [46]. However, in the early eighties the publications of Levinson [47], Juang [48] and Rabiner [24] became this methodology so popular and known.

Each HMM in a speech recognition system models the acoustic information of specific speech segments. These speech segments can be any size, e.g. words, syllables, phonetic units, etc. The acoustic models training requires great amounts of *training data*, that normally comes in a set of waveform files and orthographic transcriptions of the language and acoustic environment in question.

Along this chapter the fundamentals of this methodology are explained. As a result the Autotrain toolkit is introduced as the used technology for building HMMs, which are essential components for acoustic model training.

2.1 The Markov Chain

The HMM is one of the most important machine learning models in speech and language processing. To define it properly the *Markov chain*[3] must be introduced firstly. These are considered as extensions of finite automaton which are defined by a set of states and set of transitions based on the input observations. A Markov chain is a special

[3] "The Russian mathematician Andrei Andreyevich Markov (1856–1922) is known for his work in number theory, analysis, and probability theory. He extended the weak law of large numbers and the central limit theorem to certain sequences of dependent random variables forming special classes of what are now known as Markov chains. For illustrative purposes Markov applied his chains to the distribution of vowels and consonants in A. S. Pushkin's poem *Eugeny Onegin*." (Basharin et.al, in The Life and Work of A. A. Markov)

case of a weighted finite-automaton where each state transition is associated with a probability that shows the likelihood of the chosen path with the variant that the input sequence determines which states the automaton will go through.

A Markov chain is only useful for assigning probabilities for designed sequences without ambiguity. It assumes an important assumption, called *Markov assumption*, where each state probability depends on the previous one:

$$Pr\left(s_{\mathrm{i}}\middle|s_1 \ldots s_{\mathrm{i\text{-}1}}\right) = Pr\left(s_{\mathrm{i}}\middle|s_{\mathrm{i\text{-}1}}\right) \qquad (2.1)$$

A Markov chain is specified by $S \ = (s_{1,} \ldots , s_{\mathrm{N}})$, a set of *N* distinct states with $S_{0,}\ S_{\mathrm{end}}$ as the start and end states, a matrix of transition probabilities $A \ = (a_{01}a_{02,} \ldots a_{\mathrm{nn}})$ and an initial probability distribution $\pi = \ \pi_{1,}\pi_{2}, \ldots , \pi_{\mathrm{N}}$ over states. Each a_{ji} expresses the probability of moving from state *i* to state *j*; and π_{i} is the initial probability that the Markov chain will start in state *i*.

$$\textstyle\sum_{j=1}^{n} a_{\mathrm{ji}} \ = 1 \ \ \forall i \qquad (2.2)$$

$$\textstyle\sum_{j=1}^{n} \pi_{\mathrm{j}} \ = 1 \qquad (2.3)$$

Figure 2.3 show an example of a Markov model with three states to describe a sequence of weather events, observed once a day. The states consist of *Hot*, *Cold* and *Rainy* weather.

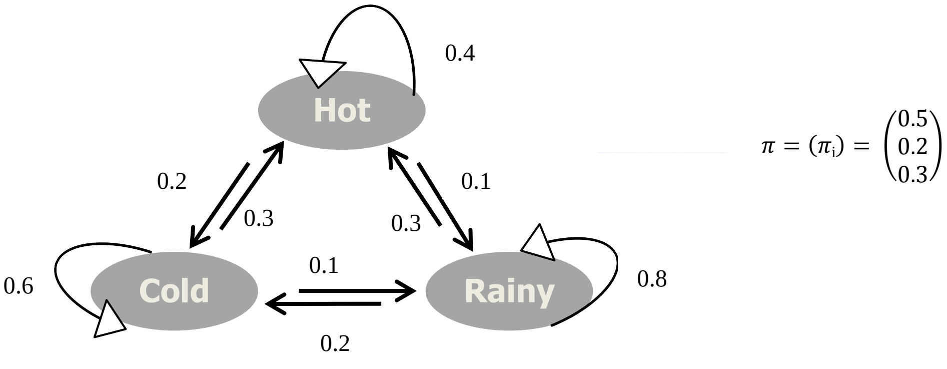

0.4
Hot
$\pi = (\pi_{\mathrm{i}}) = \begin{pmatrix} 0.5 \\ 0.2 \\ 0.3 \end{pmatrix}$
0.2
0.1
0.3
0.3
0.1
0.6
Cold
Rainy
0.8
0.2

Figure 2.3 Markov model with three states

Presuming we would find 3 consecutive *hot* days and 2 *cold* days, the probability of the observed sequence (*hot, hot, hot, cold, cold*) will be:

$$\begin{aligned} Pr(S_1S_1S_1S_2S_2) &= Pr(S_1)P(S_1|S_1)P(S_1|S_1)P(S_2|S_1)P(S_2|S_2) \\ &= \pi_1 a_{11}\ a_{11}a_{21}a_{22} \\ &= \ 0.5 \times 0.4 \times 0.4 \times 0.2 \times 0.6 = 9.6 \ \times 10^{-3} \end{aligned} \qquad (2.4)$$

2.2 The Hidden Markov Model

Each state of a Markov chain corresponds to the probability of a certain observable event happens. However, there are lot of other cases that cannot be directly observable in the real world. For example, in speech recognition we can see acoustic events in the world and then we have to infer the underlying words that are spoken on those acoustic sounds. The presence of those words is called *hidden* events because they are not observed.

The Hidden Markov Model generates an output observation symbols in any given states. This sequence of states is not known where the observation is a probabilistic function of the state. An HMM is specified by a set of states $S = (s_{1,} \dots , s_{\mathrm{N}})$ with $S_{0,} S_{\mathrm{end}}$ as start and end states, a matrix transition probabilities $A = (a_{01}a_{02,} \dots a_{\mathrm{nn}})$ (Eq.(2.2)), a set of observations $O = (O_{1,} \dots , O_{\mathrm{N}})$ correspondent to the physical output of the system being modelled and a set of observation likelihoods $B = b_{\mathrm{i}}(o_{\mathrm{t}})$, each expressing the probability of an observation o_{t} being generated from a state *i*.

$$b_{\mathrm{i}}(o_{\mathrm{t}}) = Pr(o_{\mathrm{t}}|S_{\mathrm{i}}) \qquad (2.4)$$

$$\sum_{t=1}^{n} b_{\mathrm{i}}\ (ot) = 1\ \forall t \qquad (2.5)$$

According to Markov chains an alternative representation of start and end states is the use of an initial probability distribution over states, $\pi = \pi_{1,}\pi_{2}, \dots , \pi_{\mathrm{N}}$ (Eq. (2.3)). To indicate the whole parameter set of an HMM the following abbreviation can be used:

$$\lambda = (A, B, \pi) \qquad (2.6)$$

2.2.1 Models Topology

The topology of models shows how the HMMs states are connected to each other. In Figure 2.3 there is a transition probability between the two states. This is called a *fully-connected* or *ergodic* HMM; any state can change into any other.

Such topology is normally true for the HMMs of part-of-speech tagging; however, there are other HMM applications that do not allow arbitrary state transitions. In speech recognition states can loop into themselves or into successive states, in other words it is not possible to go to earlier states in speech. This kind of HMM structure is called *left-to-right* HMM or *Bakis network* and it is used to model temporal processes that change successively along the time. Furthermore, the most common model used for speech recognition is even more restrictive, the transitions can only be made to the immediately next state or to itself. In Figure 2.4 the HMM states proceed from the left to the right, with self loops and forward transitions. This is a typical HMM used to model phonemes, where each of the three states has an associated output probability distribution.

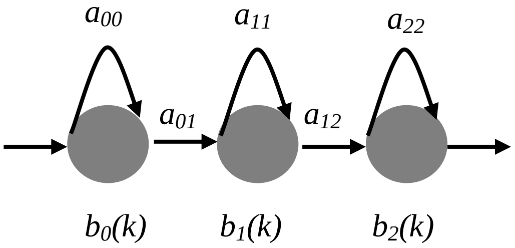


Figure 2.4 Typical HMM to model speech

For a state-dependent left-to-right HMM, the most important parameter is the number of states, which topology is defined according to the available data for training the model and to what the model was built for.

2.2.2 Elementary Problems of HMMs

We can consider as typical three elementary HMMs problems in the present literature and its resolution depends on their appliance. The further sections describe these problems and how they can be faced in the speech recognition domain.

Evaluation Problem

The focus of this problem can be summarized as follows:

What is the probability of a given model that generates a sequence of observations?

For a sequence of observations $O = \{o_1, o_{2\dots}\, o_T\}$ we intend to calculate the probability $Pr(O|\lambda)$ that this observation sequence was produced by the model λ. Intuitively the process is to sum up the probabilities of all the possible state sequences:

$$Pr(O|\lambda) = \sum_{all\ S} Pr(S|\lambda)Pr(O|S,\lambda) \qquad (2.7)$$

In other words, to compute $Pr(O|\lambda)$, first all the sequences of possible states S are enumerated, which corresponds to an observation sequence O, and then we sum all the probabilities of those state sequences.

For one particular state sequence S, the state-sequence probability can be rewritten by applying Markov assumption,

$$Pr(S|\lambda) = \pi_{s1}\, a_{s1s2} a_{s2s3} \dots a_{sT\text{ - }1sT} \qquad (2.8)$$

on the other hand the probability of an observation sequence has been generated from the model λ is:

$$Pr(O|S,\lambda) = b_{s1}(O_1) b_{s2}(O_2) \dots b_{sT}(O_T) \qquad (2.9)$$

The $Pr(O|\lambda)$ calculation using the equation 2.7 is extremely computationally heavy. However it is possible to calculate it efficiently, using the *forward-backward* algorithm [36]. Solving the evaluation problem we know how well a given HMM matches a given observation sequence.

Decoding Problem

This problem is related with the best match between the sequence of observations to the most likely sequence of states.

What is the most probable states' sequence for a certain sequence of observations?

For a given observations' sequence $O = \{o_1, o_2 \dots o_T\}$ and a model λ, the focus is to determine the correspondent states' sequence $S = \{s_1, s_2 \dots s_T\}$. Although there are several solutions to solve this problem, the one that is usually taken to choose the sequence of states with the highest probability of being taken for a certain observation sequence. This means maximizing $Pr(O|S,\lambda)$, equivalent to $Pr(S|O,\lambda)$, in an efficient way using the *Viterbi algorithm* [38].

The solution for the decoding problem is also used for the calculating the probability $Pr(O|\lambda)$ for the possible sequence of states $S \in S$. So, what makes it difficult and distinct from the evolution problem is to find not only the exact solution but the optimal one. The Viterbi works recursively, thus, it takes and points the best path for the most likely state sequence.

Estimation Problem

The estimation problem is considered as the third problem and consists on finding a method to determine the model parameters in order to optimize $Pr(O|\lambda)$. There is any optimal procedure for such a task; even so the most used solution implies the creation of a baseline model and an estimation iterative method, where each new model generates the sequence of observations with a higher probability than the previous one. The estimation problem can be summarized as follows:

How do we adjust model's parameters to maximize $Pr(O|\lambda)$?

For a given sequence of observations $O = \{o_1, o_2 \dots o_T\}$ the $\lambda = (A, B, \pi)$ parameters must be estimated in a way of maximizing $Pr(O|\lambda)$, which can be calculated by the *Baum-Welch algorithm* also known as *forward-backward* [37].

The Baum-Welch algorithm employs iteratively new parameters $\bar{\lambda}$ after the maximization of,

$$Pr(O|\bar{\lambda}) \geq Pr(O|\lambda). \qquad (2.10)$$

The estimation is applied up to a certain condition, e.g. there are no considerable improvements between two iterations.

2.3 HMMs Applied to Speech

HMM-based speech recognition systems consider the recognition of an acoustic waveform as a probabilistic problem where the recognizable vocabulary has an associated acoustic model. Each of these models gives the likelihood of a given observed sound sequence that which was produced by a particular linguistic entity.

To compute the most probable sequence of words $\widehat{W} = w_1 w_2 \dots w_m$ given by an acoustic observation sequence $O = O_1 O_2 \dots O_n$ we take the product of both probabilities for each sentence, and choose the best sentence that has the maximum posterior probability $Pr(W|O)$, expressed by Eq. (2.11).

$$\widehat{W} = \arg\max_w \; Pr(W|O) = \arg\max_w \frac{Pr(W)Pr(O|W)}{P(O)} \qquad (2.11)$$

Since $Pr(O)$ does not change into each sentence since it is carried out with a fixed observation O the prior probability $Pr(W)$, computed by the language model, and the observation likelihood $Pr(O|W)$, computed by the acoustic model, the above maximization is equivalent to the following equation.

$$\widehat{W} = \arg\max_w \; Pr(W)Pr(W|O) \qquad (2.12)$$

To build a HMM-based speech recognizer it should exist accurate acoustic models $Pr(O|W)$ that can reflect the spoken language to be recognized efficiently. This is closely related with phonetic modelling in a way that the likelihood of the observed sequence is computed in given linguistic units (words, phones or subparts of phones). This means that each unit can be thought as an HMM where the use of Gaussian Mixture Model computes each HMM state, corresponding to a phone or subphonetic unit.

In the decoding process the best match between the word sequence W and the input speech signal O is found. The sequence of acoustic likelihoods plus a word pronunciation dictionary are combined with a language model (e.g. a grammar, see 1.1.3). The most ASR systems use the Viterbi decoding algorithm. Figure 2.5 illustrates the basic structure of an HMM recognizer as it processes a single utterance.

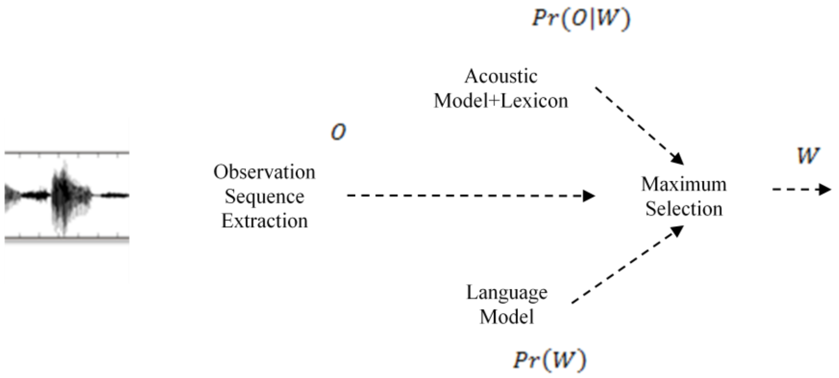


Figure 2.5 Speech recognizer, decoding an entity

2.4 How to Determine Recognition Errors

The most common accuracy measure for acoustic modelling is the *Word Error Rate* (WER). The word error rate is based on how much the word returned by the recognizer differs from a correct transcription (taken as a reference). Given such a correct transcription, the next step is to compute the minimum number of word *substitutions*, word *insertions*, and word *deletions*. The result of this computation will be necessary to map the correct and hypothesized words, and it is then defined as it follows:

$$\text{Word Error Rate} = 100\% \times \frac{Subs + Dels + Ins}{\text{Nº of words in correct transcript}} \qquad (2.13)$$

To evaluate a recognizer performance during the training stage we may want to use a small sample from the initial corpus and to reserve it for testing. Splitting the corpus into a test and training set is normally carried through in the data preparation stage (see section 2.5.4) before training a new acoustic model. If it is possible, the same speakers should not be used in both training and testing sets. The testing stage is explained in the section 2.6.

2.5 Acoustic Modelling Training

To accomplish the ASR task is essential the acoustic models training. It was used the *Autotrain* toolkit, based on the HTK, for building HMMs. Autotrain produces acoustic models for the Yakima speech decoder which is a phone-based speech recognizer engine. The choice of modelling the acoustic information based on phones is commonly used since the recognition process is based on statistical models, HMMs. There are simply too many words in a language, and these different words may have different acoustic realizations and normally there are not sufficient repetitions of these words to build context-dependent word models. Modelling units should be *accurate* to represent acoustic realization, *trainable* because it should have enough data to estimate the parameters of the unit, and *general* so that any new word can be derived from a predefined unit inventory. Phones can be modelled efficiently in different contexts and combined to form any word in a language.

Phones can be viewed as speech sounds, and they are able to describe how words are pronounceable according to their symbolic representation [39]. These individual speech units can be represented by diverse phone formats, where the *International Phonetic Alphabet* (IPA) is the standard system which also sets the principles of transcribing sounds. *Speech Assessment Methods Alphabet* (SAMPA) is another representation inventory that is often used for phone-based recognizers since it is machine-readable.

Acoustic model training involves mapping models to acoustic examples obtained from training data. Training data comes in the form of a set of waveform files and orthographic transcriptions. A pronunciation dictionary is also needed, which provides a phonetic representation for each word in the orthographic label. This is required for the training of the phone-level HMMs.

2.5.1 Speech Corpora

For training acoustic models, it is necessary a considerable amount of speech data, called a *corpus*. Corpus (plural Corpora) in linguistics is related to great collection of texts. These can be in *written* or *spoken* form; raw data type (just plain text, with no additional information) or with some kind of linguistic information, called mark-up or annotated corpora. The resources can be various such as newspapers, books or speech, it just depends on the study of target usage. Corpora can be classified as *monolingual* if there is only one language as source, *bilingual* or *multilingual* if there are more than one language. The *parallel* or *comparable* corpora are related to the same corpora but presented in different languages. In order to differentiate the spoken form from the written form language, it was ruled the words *utterance* and *sentence* correspondingly. In SR context corpora come in the shape of transcribed speech (i.e. speech data with a word level transcription).

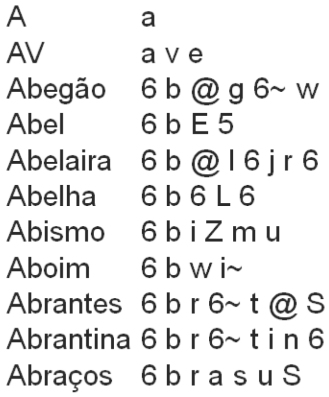
On acquiring or designing a speech corpus is important that data is appropriate for the target application and so the resulting system may have some limitations. If the corpus reflects the target audience or matches with the frequently used vocabulary, recognition will provide better recognition results. The characteristics, which a suitable corpus should consider and may influence the performance of a speech-based application, are related with speech signal variability (see 1.1.1). For example it should take into account the following categories: isolated-word or continuous-speech, speaker-dependent or speaker-independent, vocabulary-size or either the environment domain.

Another reason that makes the acquisition process a rough task is the transcription and annotation stage. For each utterance there is a correspondent orthographic transcription, often performed manually, using the simple method of hand writing which was recorded. These transcriptions also contain annotation that marks or describes non predictable or involuntary speech sounds, such as background noise or speech, misspelled words, etc.

To perform the transcription and annotation process of the acquired European Portuguese corpora in the SIP project, the author has used a tool developed by MLDC. The SIP project is explained with more detail in Chapter 4.

2.5.2 Lexicon

A lexicon is a file containing information about a set of words. Depending on the purpose of the lexicon, the information about each word can include orthography, pronunciation, format, part of speech, related words, or possibly other information. In this case it is referred as a phonetic dictionary that lists the phonetic transcriptions of each word (it represents how the word can be pronounced in a certain language). Figure 2.6 shows an EP lexicon sample using the SAMPA phonetic inventory.



A	a
AV	a v e
Abegão	6 b @ g 6~ w
Abel	6 b E 5
Abelaira	6 b @ l 6 j r 6
Abelha	6 b 6 L 6
Abismo	6 b i Z m u
Aboim	6 b w i~
Abrantes	6 b r 6~ t @ S
Abrantina	6 b r 6~ t i n 6
Abraços	6 b r a s u S

Figure 2.6 Phonetic transcriptions of EP words using the SAMPA system

When a model is trained with a new speech corpus, the transcriptions associated with the corpus can contain words that are not included in the acoustic model training lexicon. These missing words must be added to the training lexicon with a pronunciation. *Letter-to-sound* (LTS) rules are used to generate pronunciations of new words that are not in the pronunciation lexicon. These rules are mappings between letters and phones that are based on examples in the LTS training lexicon. However LTS-generated pronunciations should be validated and corrected by a native linguist expert.

It was adopted two LTS training methods: the *classification and regression trees* (CART) based-LTS methodology and the Graphoneme (Graph) LTS method. CART *[52*] represents an important technique that combines rule-based expert knowledge and statistical learning. On the other hand, Graph uses graphonemes trigram concept to train LTS rules.

Annex 1 describes thoroughly the adopted process in creating a phonetic lexicon of 100 thousand words for the European Portuguese language. This compilation was performed by the author and supported by a linguist expert for selecting and validating the pronunciations automatically generated.

2.5.3 Context-Dependency

In order to improve the recognition accuracy, most *Large Vocabulary Continuous Speech Recognition* (LVCSR) systems replace the idea of context-independent models with context-dependent HMMs. Context-independent models are known as monophones. Each monophone is trained for all the observations of the phone in the training set independently of the context in which it was observed. The most common context-dependent model is a triphone HMM, and it represents a phone in a particular left and right context. The left context maybe be either the beginning of a word or the ending of the preceding one, depending on whether the speaker has paused between words or not. Such triphones are called *cross-word triphones*. The following example shows the word *CAT* represented by a monophone and triphone sequences:

CAT	k	ae	t	Monophone
CAT	sil-k+ae	k-ae+t	ae-t+sil	Triphone

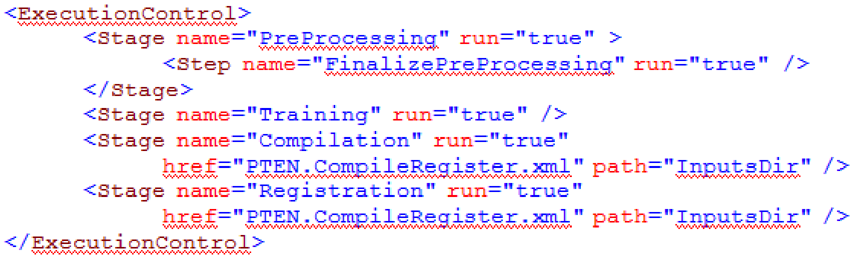
Triphones capture an important source of variation and they are normally more accurate and faster than monophones, but they are also much larger model sets. For example if we have a phoneset with 50 phones we would need circa 50^3 triphones. To train up such a large system we would need a huge impractical amount of training data. To get around this problem as well the problem of data sparsity, we must reduce the number of triphones that are needed to train. So, we share similar acoustic information between parameters of context dependent models, called *clustering*, and *tying* subphones whose contexts are in the same cluster.

2.5.4 Training Overview

Autotrain can be described as a set of tools designed to help the development of SR engines. It is based on HTK tools to allow power and flexibility in model training for advanced users but at the same time it facilitates the training task by providing a framework whose developers and linguists can take advantage. This tool is configured using XML files and executed through PERL batch scripts.

The first contact with the Autotrain tool was through English and French tutorials which are end-to-end examples of how to use the AutoTrain toolkit. With this material, each step of the training process (outputs and whose files are required as input) can be observed. It was also possible to learn how to prepare raw data, train the acoustic model, build the necessary engine datafiles (compilation) and register the engine datafiles for the Microsoft Yakima decoder.

The building of a HMM recognition system using Autotrain localization process can be divided into four main: *Preprocessing, Training, Compilation* and *Registration*. The whole execution is controlled by the code within the tag <ExecutionControl> in the main XML file *(languageCode).Autotrain.xml* (Figure 2.7).



```
<ExecutionControl>
      <Stage name="PreProcessing" run="true" >
            <Step name="FinalizePreProcessing" run="true" />
      </Stage>
      <Stage name="Training" run="true" />
      <Stage name="Compilation" run="true"
            href="PTEN.CompileRegister.xml" path="InputsDir" />
      <Stage name="Registration" run="true"
            href="PTEN.CompileRegister.xml" path="InputsDir" />
</ExecutionControl>
```

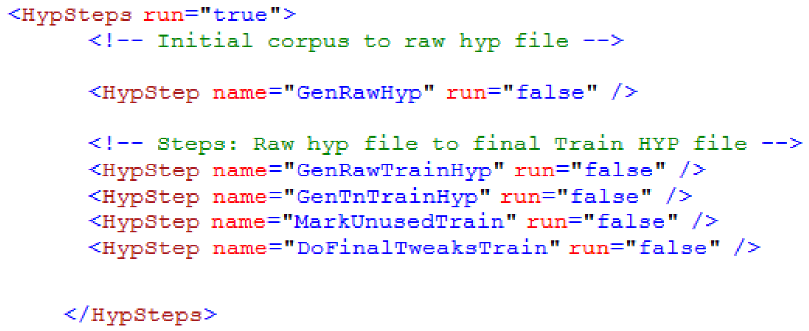
Figure 2.7 Autotrain execution control code

Preprocessing Stage

After acquiring an appropriate speech database the next step is to organize a training area and prepare the data into a suitable form for training. The preparation of data is essential and the first thing to do is to prepare the input speech files into the Microsoft waveform format (.wav). All the corpora (both training and test sets) must be in a supported format, and should be converted if necessary. The Sox tool [56] is an audio converter that is freely available on the Internet, and used to convert raw audio files into .wav format.

Then a *Hyp* file is generated and contains all the corpus information such as wave file name, speaker gender information and word level transcriptions. It also specifies if an utterance is to be used in training, testing or ignored. Initially orthographic transcriptions are un-normalized and require some normalization before the training begins. Normalization consists in selecting and preparing the raw HYP file information. A Hyp file example with some guidelines for transcriptions normalization can be seen in Annex 2.

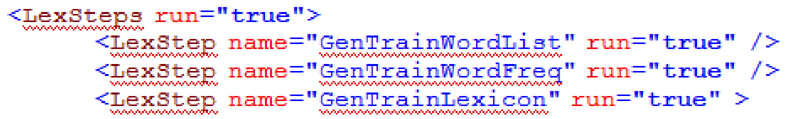
In Autotrain this process is controlled by a configuration XML file (Figure 2.8) and executed through a batch script.The <HypSteps> tag controls the generation and validation of a HYP file. At the beginning HYP file generation is based on Corpus metadata, referred as MS Tables. This first version (raw HYP) is obtained from two MS Tables, *UtteranceInformationTable* and *SpeakerInformationTable*, which contain all the relevant corpus information about each recorded utterance, speaker identifier, microphone, recording environment, dialect, gender and orthographic transcription.The following steps concern the normalization of training utterances, the extraction of unused utterances and the exclusion of bad files such as empty transcriptions, missing acoustic files or poor acoustic quality files.



```
<HypSteps run="true">
      <!-- Initial corpus to raw hyp file -->

      <HypStep name="GenRawHyp" run="false" />

      <!-- Steps: Raw hyp file to final Train HYP file -->
      <HypStep name="GenRawTrainHyp" run="false" />
      <HypStep name="GenTnTrainHyp" run="false" />
      <HypStep name="MarkUnusedTrain" run="false" />
      <HypStep name="DoFinalTweaksTrain" run="false" />

    </HypSteps>
```

Figure 2.8 <HypSteps> tag controls the generation and validation of a HYP file

Preprocessing stage also controls the training lexicon generation, which is a pronunciation lexicon containing all the words that appear in the transcription file (.Hyp file). The transcribed words that are not found in the main language phonetic dictionary are generated by LTS and hand checked by a linguistic. <LexSteps> also controls the generation of a word list and word frequency list of the training corpus words (Figure 2.9).



```
<LexSteps run="true">
      <LexStep name="GenTrainWordList" run="true" />
      <LexStep name="GenTrainWordFreq" run="true" />
      <LexStep name="GenTrainLexicon" run="true" >
```

Figure 2.9 <LexStep> tags controlling the generation of the training dictionary

Summarizing some files have to be provided before the training process starts:

- Spoken Utterances – audio files in .wav format.
- Transcription file (.HYP) – for each audio file there is an associated transcription, the .HYP file maps each .wav file to its respective transcription. The following example means that the *wy1* wave file is in the directory *data*, the speaker gender is indeterminate *(I)* and "UM" is the audio transcription.

  wy1 data 1 1 I TRAIN <PlaceHolder> <PlaceHolder> UM

- Pronunciation lexicon (.DIC) – For all words contained in the transcription file (.hyp) there is a respective pronunciation according to a specific phoneset.

Abelha	aex b aex lj aex

Abismo aex b i zh m u

- Phoneset (mscsr.phn) – Describes the possible phones for a specific language.
- Question set file (qs.set) – The question set file is essential for clustering triphones into acoustically similar groups. As an example of a linguistic question:

  QS "L_Class-Stop" { p-*,b-*,t-*,d-*,k-*,g-*}

Training Stage

Acoustic model training involves mapping acoustic models (using phones) with equivalent transcriptions. This kind of phone models is context-dependent; it makes use of triphones instead of monophones.

The models used have as topology HMMs of three states: each state consume a speech segment (at least 10ms) and represents a continuous distribution probability for that piece of speech. Each distribution probability is a Gaussian density function and is associated with each emitting state, representing the speech distribution for that state. The transactions in this model are from left to right, linking one state to the next, or self-transactions. Figure 2.10 illustrates the used model topology.

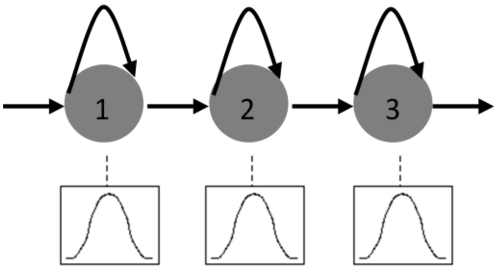


Figure 2.10 Used HMM topology

Similar acoustic information is shared through HMMs by sharing/tying states. These shared states, called *senones*, are subphonetic units context dependent and equivalent to a HMM state of a triphone. This means that each triphone is made up of three senones and it contains a model of a particular sound. During the training process the number of senones are defined according to the hours of speech of training data, as well as the number of mixtures of those tying states to ensure that the whole set of acoustic information is estimated properly.

The training stage can be divided into several sub-stages. At first the coding of parameters takes place. The wave files are split into 10 ms frames for feature extraction to produce a set of .mfc files (speech parameters). These files contain speech signal representations called Mel-Frequency Cepstrum Coefficients (MFCC) [*53*]. MFCC is a representation defined as the real cepstrum of a windowed short-time signal derived from the *Fast Fourier Transform* (FFT) of that signal. Each frame or speech representation encodes speech information in a form of a feature vector.

For training a set of HMMs, every file of training data must have an associated phone level transcription. The starting point of phone transcription is an orthographic transcription in HTK label format, a Master Label File (MLF) which is a single file containing a complete set of transcriptions. This allows the same transcriptions to be used with different versions of the speech data to be stored in different locations.

The training begins by converting word level transcriptions into monophone level transcriptions. Once reasonable monophone HMMs were created, a *forced alignment* of the training data can be performed. Concerning this, a new phone level MLF is created in which the choice of pronunciations depends on the acoustic evidence. This new MLF can be used to perform a final re-estimation of the monophone HMMs. These models are iteratively updated by traversing the training data repeatedly and mapping the models to the monophone labels in the transcription.

After producing an initial monophones model, the respective cross-word triphones are cloned for each monophone. This is done in two steps: first the monophone transcriptions are converted into cross-word transcriptions and the cross-word triphones re-estimated to produce initial single-mixture models. Then similar acoustic states of these triphones are clustered and tied as a guarantee that all the state distributions can be robustly estimated. Since the system size is vastly reduced at this stage, we can increase the number of mixtures per senone. This leads to an initialized cross-word acoustic model, which is used to run through the training data and re-label the transcriptions to allow multiple pronunciations. After this these cross word transcriptions are re-used to update the cross-word acoustic model leading to the final cross-word triphone system once again. Figure 2.11 represents the training process described above.

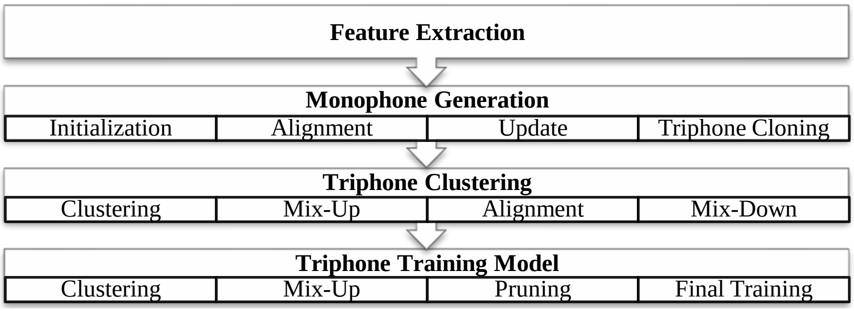


Figure 2.11 Training acoustic models flowchart

Compilation Stage

Once the training stage is complete, the trained acoustic models (MMF files) and related data files are compiled and then registered. These are summarized as follows:

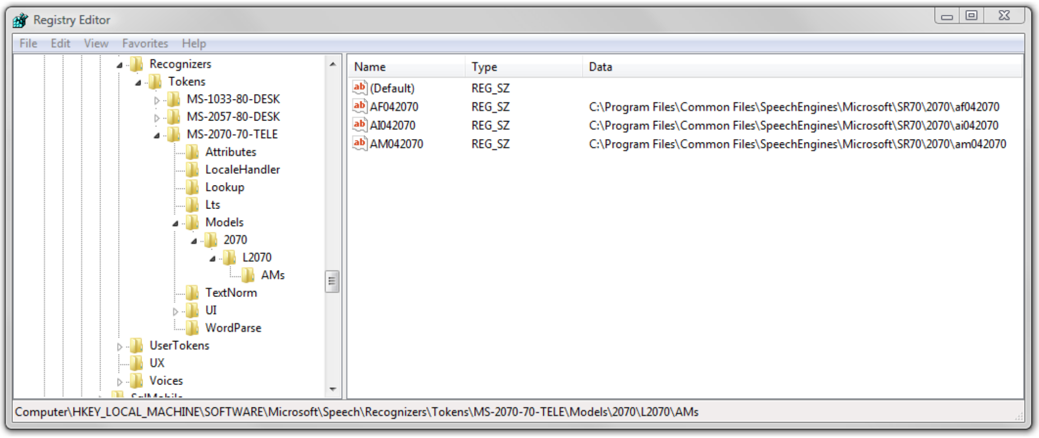
- L<Languagecode>.phn - phone set converter
- L<Languagecode>.smp - senone map file
- L<Languagecode>.cw - cross word models
- lsr<Languagecode>.lxa - lexicon
- r<Languagecode>sr.lxa - lts rules
- A<Languagecode>.am – acoustic model

Registration stage

In this stage it is performed the registration of the SAPI engine dlls in addition to the compiled files. Engine Registration performs the following actions:

- Registering the SAPI engine dlls
- Registering the SAPI Phone Converters
- Setting up the SAPI Engine Token in the registry with the correct attributes for the platform
- Setting up the SAPI Engine Token to point to the compiled data files

The engine token is registered under as illustrated in Figure 2.12. After this step it is possible to run speech recognition on the specified language.



Name	Type	Data
(Default)	REG_SZ	
AF042070	REG_SZ	C:\Program Files\Common Files\SpeechEngines\Microsoft\SR70\2070\af042070
AI042070	REG_SZ	C:\Program Files\Common Files\SpeechEngines\Microsoft\SR70\2070\ai042070
AM042070	REG_SZ	C:\Program Files\Common Files\SpeechEngines\Microsoft\SR70\2070\am042070

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Recognizers\Tokens\MS-2070-70-TELE\Models\2070\L2070\AMs

Figure 2.12 Registered engine

2.6 Testing the SR Engine

There are two types of accuracy tests: verification tests and validation tests. Verification tests are used to monitor the progress of the acoustic model development. This is tied to a specific corpus which means each time a particular speech corpus is used to train an acoustic model, a corresponding accuracy test should be built from this corpus. Before any training begins the corpus is partitioned into a set for testing and one for training.

Validation tests are constructed from data to represent how the SR system performs in the context of specific user scenarios. Each scenario is represented by data that is selected to the system requirements. For example, data collected in a noisy office is selected for measuring the accuracy of an SR system in a noisy scenario. This type of tests requires detailed specifications for all the user scenarios and sufficient data to provide test coverage for all of them.

This work has only used verification testing for determining if the addition of new training data or either changing model parameters (see Chapter 3) improves accuracy for the test set (e.g. Table 1) initially defined. If the result of the test shows an accuracy regression, after the model has been trained with additional data, it will mean that this new training set is not helping the model.

2.6.1 Separation of Test and Training Data

It is important that test files are not used for model training. Additionally, data of the speakers should not be split through training and test sets, which means that all the files belonging to each speaker should be in the test set or in the training set.

Selecting data for verification test the percentage that should be taken from the initial corpus is based on the number of speakers. Normally it is 5-10% of the total data and this must reflect the same distribution as the whole corpus. For example, if the gender distribution of the corpus is 40% females and 60% males, the test set should also match with this distribution as close as possible.

If the corpus does not include information about speakers' gender, accents, or recording environments, the test set can be randomly selected from all the speakers in the corpus.

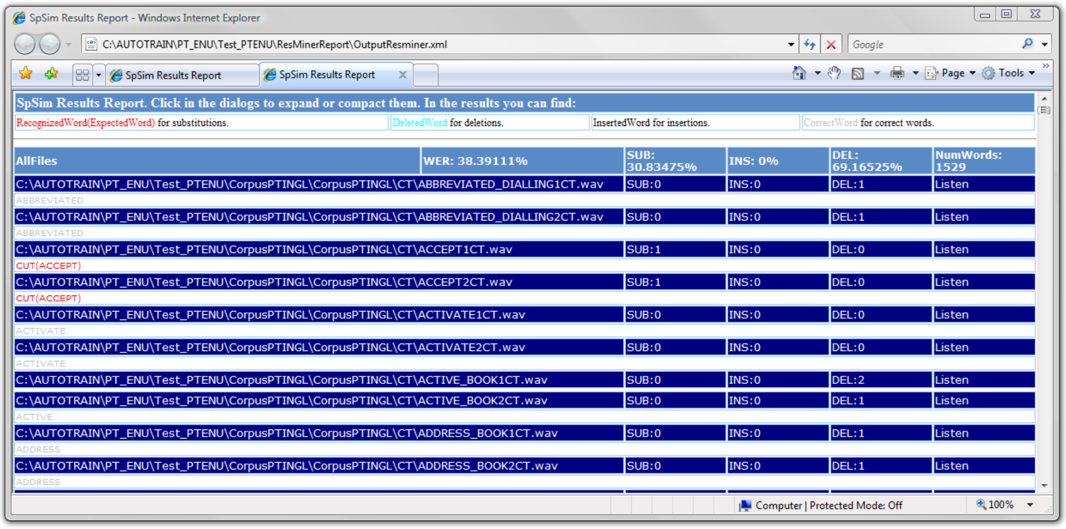
2.6.2 Developing Accuracy Tests

After selecting the proper test data and having prepared it according to the normalization of the guidelines reported in Annex 2, the next step is to write the grammar, because the SR engine to recognize the speech data must have a grammar that specifies what's accepted as a valid utterance. For testing the developed models a CFG grammar was written to parse the recognizer output.

Grammars can be specified in two different syntaxes, W3C or SAPI. These are mainly different in syntax, i.e. the names of the tags are different, and only slightly different in the structure, but the concepts of building a good grammar are the same. Specification for W3C grammar can be found on [57] and SAPI grammar on [58].

Once the grammar is specified and checked, the accuracy of the test analysis can be performed using the *ResMiner* tool (provided by SCG). This tool obtains the accuracy measures using XML script configuration files whose specify the grammar that should be loaded and the reference transcriptions that will be compared with the recognition result.

The output of *ResMiner* is a XML that has WER and the percentage of substitutions, insertions and deletions for the group of utterances presented in the configuration files. Figure 2.13 illustrates an output of the *ResMiner* execution.


SpSim Results Report - Windows Internet Explorer
C:\AUTOTRAIN\PT_ENU\Test_PTENU\ResMinerReport\OutputResminer.xml
Google
SpSim Results Report
SpSim Results Report
Page
Tools
Computer | Protected Mode: Off
100%

SpSim Results Report. Click in the dialogs to expand or compact them. In the results you can find:

RecognizedWord(ExpectedWord) for substitutions.	DeletedWord for deletions.	InsertedWord for insertions.	CorrectWord for correct words.

AllFiles	WER: 38.39111%	SUB: 30.83475%	INS: 0%	DEL: 69.16525%	NumWords: 1529
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ABBREVIATED_DIALLING1CT.wav		SUB:0	INS:0	DEL:1	Listen
ABBREVIATED					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ABBREVIATED_DIALLING2CT.wav		SUB:0	INS:0	DEL:1	Listen
ABBREVIATED					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ACCEPT1CT.wav		SUB:1	INS:0	DEL:0	Listen
CUT(ACCEPT)					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ACCEPT2CT.wav		SUB:1	INS:0	DEL:0	Listen
CUT(ACCEPT)					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ACTIVATE1CT.wav		SUB:0	INS:0	DEL:0	Listen
ACTIVATE					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ACTIVATE2CT.wav		SUB:0	INS:0	DEL:0	Listen
ACTIVATE					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ACTIVE_BOOK1CT.wav		SUB:0	INS:0	DEL:2	Listen
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ACTIVE_BOOK2CT.wav		SUB:0	INS:0	DEL:1	Listen
ACTIVE					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ADDRESS_BOOK1CT.wav		SUB:0	INS:0	DEL:1	Listen
ADDRESS					
C:\AUTOTRAIN\PT_ENU\Test_PTENU\CorpusPTINGL\CorpusPTINGL\CT\ADDRESS_BOOK2CT.wav		SUB:0	INS:0	DEL:1	Listen
ADDRESS					

Figure 2.13 ResMiner output

2.7 Conclusions

This chapter can be divided into two distinct parts: the first describes the technology for training acoustic models. The second one presents the several stages of the training process.

It approaches basis technology used in this work, HMMs; the components that allow producing new acoustic models together; and the procedures of speech recognition.

So it is described the main application of this work – Autotrain. This training tool consists of a set of stages for the system development. Each stage was thoroughly described.

Chapter 3

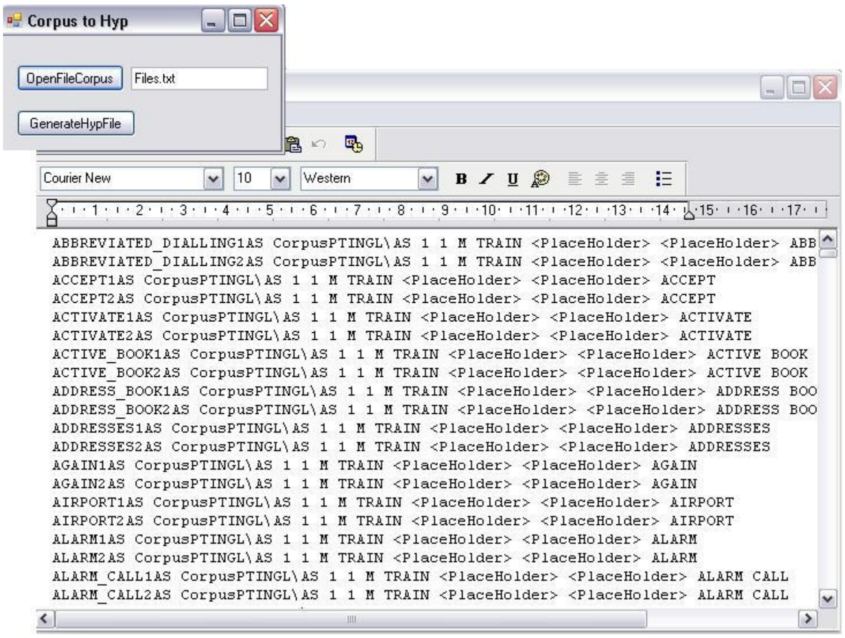
Comparison of Native and Non-native Models: Acoustic Modelling Experiments

In this chapter different acoustic modelling methods are explored. They test their efficiency for recognition improvement on non-native speech. The results refer to experiments with cross-word triphone models which were obtained in a process reported in [1] and explained in the precedent chapter. Recognition was done using the Viterbi algorithm [38], used for obtaining the best sequence of states that match the sequences of speech frames that correspond to a certain unit. This study was based on an experimental corpus of English spoken by European Portuguese speakers. This corpus is part of a larger one used in the Teixeira and Trancoso [4].

Model sizes depend on how many hours of training data are available. Considering the amount of data, it was defined that the resultant models would have a total of 1500 tying states (senones). For an initial number of mixtures we have a total of 12 mixtures and as a final smoothing stage we reduce the total average of mixtures of the final system to 8 mixtures. For testing the several SR systems we have defined a set of data dedicated to testing (see Table 1). As we are talking about command and control systems, a CFG grammar was built, with all the sequences of words found in the test set (Annex 3).

3.1 Data Preparation

In order to improve the preprocessing stage the author has developed an application for generating the normalized Hyp file for the used corpora. Figure 3.14 shows an execution example of *CorpusToHyp* with the generated Hyp file.



```
ABBREVIATED_DIALLING1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ABB
ABBREVIATED_DIALLING2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ABB
ACCEPT1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACCEPT
ACCEPT2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACCEPT
ACTIVATE1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVATE
ACTIVATE2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVATE
ACTIVE_BOOK1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVE BOOK
ACTIVE_BOOK2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVE BOOK
ADDRESS_BOOK1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ADDRESS BOO
ADDRESS_BOOK2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ADDRESS BOO
ADDRESSES1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ADDRESSES
ADDRESSES2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ADDRESSES
AGAIN1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> AGAIN
AGAIN2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> AGAIN
AIRPORT1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> AIRPORT
AIRPORT2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> AIRPORT
ALARM1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ALARM
ALARM2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ALARM
ALARM_CALL1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ALARM CALL
ALARM_CALL2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ALARM CALL
```

Figure 3.14 CorpusToHyp – Execution example and generated Hyp file

The corpus files were renamed (each file has a unique identification) and organized in different directories according to the correspondent speaker's session. The procedure for training this corpus is the same as reported in 2.5.4. The training process begins after creating the XML input files and preparing the phone set, question set, pronunciation lexicon and Hyp file.

3.1.1 Training and Test Corpora

This study was based on an experimental corpus of English spoken by European Portuguese speakers. This corpus is part of a larger one used in the Teixeira and Trancoso [4]. There are approximately 20 speakers (10 male and 10 female) for each accent, but only the male sub-set corpus was used in these experiments. A native English Corpus [4] was used to accomplish the experiments related with the application of the adaptation techniques or updating models. The audio files were sampled at 8 kHz with 16 bits-linear precision. Each speaker has recorded approximately 227 English isolated words twice. The training and the test set are then separated to build a combined model for a speaker-independent speech recognition system. Table 1 shows the implied corpus and the partition for training and testing data set in this study. The phone sets of the languages presented in this study are defined using the SAMPA phonetic alphabet.

Data	Partition	Speakers	Utterances	Minutes
Non-native Data	Training	8	3468	35
	Testing	3	1221	12
Native Data	Training	7	3476	34
	Testing	2	996	9

Table 1 Database overview

3.2 Baseline Systems

Both non-native and native acoustic models were trained with the respective training set described in Table 1. The training lexicon which lists the phonetic pronunciation of all the words in the corpus uses the English phone set. The parameters and training procedure are the same for the two models. The non native and native speech engines were tested with the same corpus. Table 2 shows the performance on the non-native and native test set when the both models are used. The remaining scores show better recognition performance when non-native models are used for the foreign accent.

Models	Non-Native Models	Native Models
Non-Native test set	6.28%	13.41%
Native test set	22.89%	4.09%

Table 2 Accuracy rate on non-native and native data (WER %)

3.3 Experiments an Results

3.3.1 Pooled Models

Non-native speech recognition can be viewed as a speaker independent's recognition problem whose traditional approach has been to pool all the speech data from as many speakers as possible as if it belonged to a single speaker. Pursuing this idea the native model was retrained with the available non-native data (pooled models). As we can see (Figure 3.15) the improvement in pooling the native and non-native training data indicates that recognition of non-native data can profit from native data. Since both corpora have almost the same training of utterances, one way of weighting the non-native training utterances is to set up the weight parameter of the non-native training corpus.

The optimal weighting factor was found to be 2.0 for non-native data where recognition scores reveal a Word Error Rate (WER) of 6.02% (non-native test) and 4.17% (native test). The recognition performance is slightly better when these results are compared with the English baseline system (Table 2). In [50] a pooled model using English native data and German accent shows an increasing of 1.2% in accuracy.

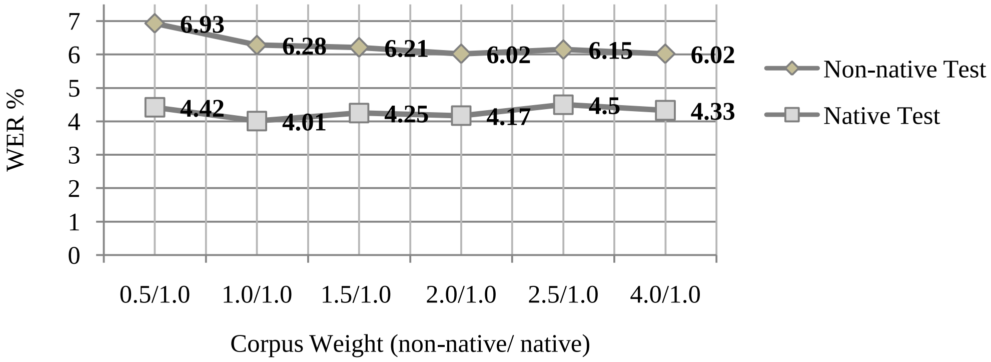


Figure 3.15 Pooled models using different corpus weights for non-native corpus

3.3.2 Adaptation of an English Native Model

The adaptation of acoustic models reduces the mismatch between a certain model set and the adaptation data. The adaptation can be at speaker’s level, environment or characteristics of a group of speakers such as speakers with foreign accent. The most used techniques are the MLLR and the MAP. MLLR computes a set of transformations, where one single transformation is applied to all the models in a transformation class. In other words, it estimates a set of linear transformations for the context and variance parameters of a Gaussian mixture HMM system.

The effect of these transformations is to shift the meanings of the components and to alter the variances in the initial system so that each state in the HMM system can be more likely to generate the adaptation data. In MAP adaptation we need a prior knowledge of the model parameter distribution. The model parameters are re-estimated individually requiring more adaptation data to be effective. When larger amounts of adaptation training data become available, MAP begins to perform better than MLLR, due to this detailed update of each component.

As we had a small amount of data in this experiment, we have used only the MLLR method, applied to English native model (baseline model) adapted to non-native corpus. Each model adaptation was iterated 4 times. Adaptation has improved the performance on non-native recognition, revealing 6.21% WER for non-native test. In a native English test no changes were found, giving the same WER as the native model, 13.41%.

3.3.3 Mapping English Phonemes into Portuguese Phonemes

When a foreign language is spoken the native phonological structure can be changed or either unfamiliar sounds are adapted to similar/closer ones of their native phoneme inventory [13] [14]. For example, the English pharyngeal voiceless fricative in <hit> is commonly not articulated by Portuguese native speakers who speak in English because this phoneme is not present in the Portuguese phoneme set. This also depends on the speaker’s proficiency, which will determine how different a native accent is from a foreign accent.

In order to get better recognition results on non-native accents, English phonemes were mapped into Portuguese phonemes (see Annex 4). The mapping was done by a linguist expert who defined which phoneme inventory should be taken into account to describe a standard Portuguese English pronunciation. This phoneme inventory was selected bearing in mind the pronunciation of a Portuguese prestigious group/community, with a good knowledge of the English language. As a prestigious group we mean the higher literacy level group that uses a dominant variant or pattern dialect of a given language [49] [51]. Using the phonetic inventory SAMPA, 33 phones were defined to transcribe the English language when spoken by native Portuguese speakers. The resultant phone set is presented in Annex 4.

The majority of the English phones suffered a direct mapping, except those that represent non Portuguese existing sounds. One example of that is the dental voiceless fricative [T] (e.g. <thriller>, <thirties>), that in European Portuguese language is converted into a dental voiceless plosive [t] or alveolar voiceless fricative [s]. We consider that the prestigious group recognizes this sound so we have included it in our new phone set. Another non existing sound in the European Portuguese is the pharyngeal voiceless fricative [h] (e.g. <hang>). However, for the prestigious group there is, for example, differentiation in pronunciation between <and> and <hand>, which made us include this phone in the English-Portuguese mapped phone set. Finally, another important difference we need to consider is the approximant [r] (e.g. <red>). This phone does not exist in the European Portuguese phone set. In spite of this fact, when a Portuguese speaker uses English words as <red>, we can say [R E d] (PT SAMPA), in other cases such as <car> we say [k a r] (PT SAMPA). For this reason we added this phone to the new mapped phone set.

Afterwards new word transcriptions of the corpus vocabulary were required, following an accurate process accomplished by the linguist expert. The initial English phonetic transcription was directly mapped to the Portuguese form (using the new phone set), which is not enough for a reliable representation of the English spoken with a Portuguese accent. An improved phonetic transcription was accomplished by modifying the phonetic transcription of each word, taking into account the Portuguese prestigious accent using English. An example of that adaptation will be the way an English or a Portuguese speaker will pronounce words that end with a plosive consonant such as [t] (e.g <art>). In English we should transcribe the word <art> like [Q r t] (PT SAMPA) but in Portuguese we say [Q r t @] (PT SAMPA).

Results reveal 7.26% of WER for the new trained models using the phone set described above. The recognition accuracy has decreased, compared it with the baseline non-native system or the pooled model experiment, but it is still far from the English native model when tested with a non-native test set. This is an encouragement for continuing to explore this subject.

Another experiment was to train a pooled model using this new phone set, but instead of using the English native model, we have experimented it with a Portuguese native model. The new phone set uses the same phones as the Portuguese native model as well the speakers of the additional training data are also Portuguese, doing this we are representing a European Portuguese recognition system that also supports English words with a Portuguese accent.

The Portuguese native model was trained with 87 hours of speech for a total of 553 speakers (266 male and 287 female). The results of such an experiment were encouraging, as the system reveals 9.81% of WER on testing with the non-native test set. This value is still above the baseline English system results, which means that an accurate phonetic representation may improve recognition performance of non-native speakers.

The following graphic (Figure 3.16) gives the best results and its proportion between methods. As we can infer from the last experience, where we have used the mapped phone set to train new models, the test using the English native test corpus was not performed. In these experiments we work with a different phone set from the one used in the English native model, so the results related with testing those models with English native speakers are not relevant for the experiment.

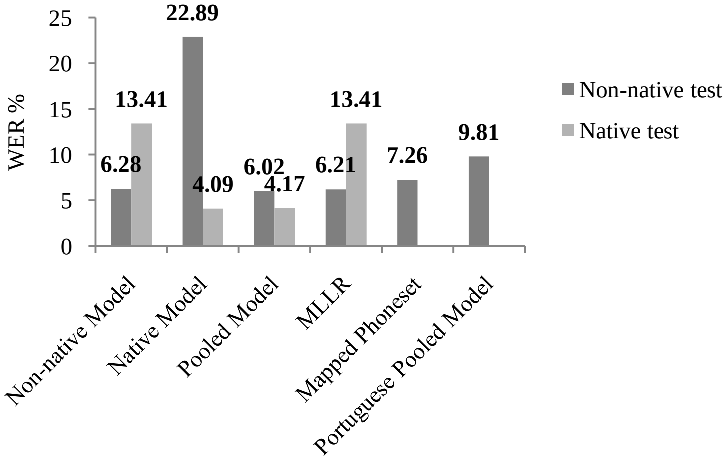


Figure 3.16 Best results of the different experiments

3.4 Conclusions

In this chapter we have explored several ways of adapting automatic speech recognition systems to non native speakers. The results show that a small amount of data can be successfully used for the improvement in the recognition of non-native accent. Even though some applied methods reveal worst performance results when compared to the non-native baseline models, there are considerable improvements in the English native models recognizing non-native accents.

Chapter 4

Collection of Portuguese Speech Corpora

Preparing not only high-quality training but also testing data begins with the careful selection of an appropriate speech corpus. A corpus selection criterion requires the following speaker, recording and content characteristics.

- Speaker's characteristics: gender, accents, and ages.
- Recording characteristics: microphone type, recording environment, sampling rate, and file format.
- Content characteristics: prompt categories and vocabulary domains.

Some corpora acquisitions were performed in order to improve telephonic acoustic models for command and control telephony applications. The SIP and Auto-attendant projects supplied great platforms for retrieving and preparing speech corpora. These are explained in further subsections.

For each collected and prepared corpus, each speaker was assigned to an unique *speaker ID* number; each utterance was consecutively numbered with a unique *utterance ID*; utterances by a single speaker were grouped into one or more *sessions* and each corpus was assigned to an unique *corpus ID.*

During corpora compilation and all the tasks concerned with training process, were developed useful applications for preparing and validating Autotrain input files. Those were performed in the Microsoft Visual Studio .NET framework [54], using the object-oriented language, C#.

4.1 Research Issues

A research work about the existing Portuguese text and speech corpora was made. The result was a survey organized according to the different resources and different types of corpora (written text, transcription of spoken texts, speech and parallel/comparable corpora). Spoken corpora are divided into the speakers' recordings made on the telephone (fixed or mobile) network, or through a microphone, whereas the written corpora are wordlists, lexicons, plain and annotated texts. Each corpus was described concerning its size, type of information, availability, sources and costs. The document was often updated due to new daily information regarding new corpora or new sources. The last update of this document can be consulted in Annex 5. However, part of it concerns an internal Microsoft repository whose nature is confidential and has to be omitted.

Afterwards some of these corpora were obtained, focusing on wordlists, with the intention of creating a large lexicon which will be used to train and generate new acoustic models, after a phonetic transcription and analyzed by a linguist.

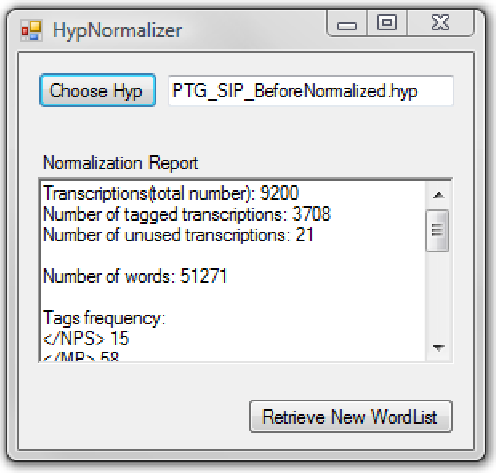
Concerning the speech acoustic analysis and its contribution for the development of TTS and SR systems, it was made a survey of the existing speech analysis and transcription software. For each tool its functionalities and characteristics were described with more emphasis in those that are related to the alignment and transcription of speech corpora. All these documents are in Annex 5.

4.2 SIP Project

The SIP project is a collection process of generic speech data, including methodologies and tools for the acquisition of telephony speech that can be used for training and/or testing of acoustic models especially applied for command and control telephony applications. The process has been specifically applied to the case of the acquisition of telephony speech corpus for the needs of Exchange UM - Unified Messaging (OVA-Outlook Voice Access experience) and it assumes that the proper acquisition will be held in the Microsoft corpnet. The author was involved in this project regard o data preparation and acoustic modelling.

After collecting the entire EP corpora the first stage was to transcribe and annotate all the audio files manually. This was performed using a quality control tool reported in section 2.5.1. Then when all the audio was transcribed the tool outputted the Hyp file correspondent to the corpus. This Hyp file was rechecked and normalized within a new quality check stage, before training a new acoustic model with the collected data.

The author has developed the *HypNormalizer* tool (Figure 4.17) to skip the Preprocessing stage and improve the Hyp file normalization inside this project.



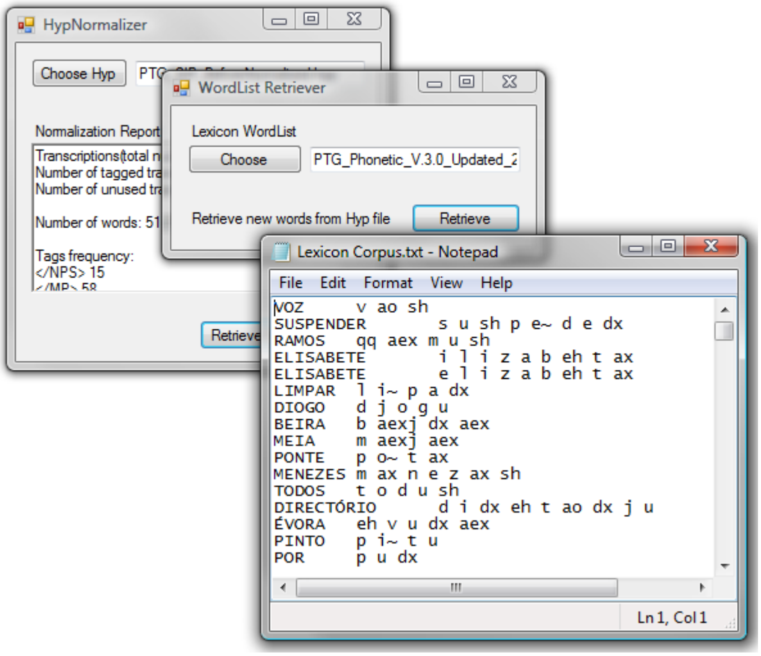
PTG_SIP_BeforeNormalized.hyp

Normalization Report

```
Transcriptions(total number): 9200
Number of tagged transcriptions: 3708
Number of unused transcriptions: 21

Number of words: 51271

Tags frequency:
</NPS> 15
</MP> 58
```

Figure 4.17 HypNormalizer execution sample

This tool generates a normalized Hyp file reporting some statistics about the characteristics of transcriptions (e.g. tags frequencies, number of words). It also outputs tagged transcriptions into a different file just for fast control. It also allows the compilation of the lexicon training corpus. Figure 4.18 shows an example of the execution of the lexicon compilation correspondent to the collected corpora.



PTG_Phonetic_V.3.0_Updated_2

```
VOZ      v ao sh
SUSPENDER        s u sh p e~ d e dx
RAMOS    qq aex m u sh
ELISABETE        i l i z a b eh t ax
ELISABETE        e l i z a b eh t ax
LIMPAR   l i~ p a dx
DIOGO    d j o g u
BEIRA    b aexj dx aex
MEIA     m aexj aex
PONTE    p o~ t ax
MENEZES m ax n e z ax sh
TODOS    t o d u sh
DIRECTÓRIO       d i dx eh t ao dx j u
ÉVORA    eh v u dx aex
PINTO    p i~ t u
POR      p u dx
```

Figure 4.18 Training lexicon compilation using Hyp file information

Using the EP phonetic lexicon the application compares the words of Hyp file transcriptions and gets back the correspondent phonetic transcriptions into a new file.

All the words presented in the Hyp file that do not have phonetic transcription are outputted into a *NewWords.txt* file to facilitate automatic transcription through LTS rules. These new words are then revised by a linguistic expert.

4.3 EP Auto-attendant

The EP Auto-Attendant was developed by MLDC for its appliance in the Portuguese MS Subsidiary. It is a Speech Server based application which enables users to call a specific number and interact with a virtual operator. This operator allows the users to call a specific person saying his first and last name. The application logs all the incoming calls and stores them in a SQL Server database. Figure 4.19 shows the EP Auto-attendant architecture, according to the given call workflow:

- Caller places a phone call into the system and asks for a colleague by his name.
- After the successful recognition the system asks for confirmation and transfers the call to the destination callee.
- Callee receives the call and immediately starts talking with the caller.

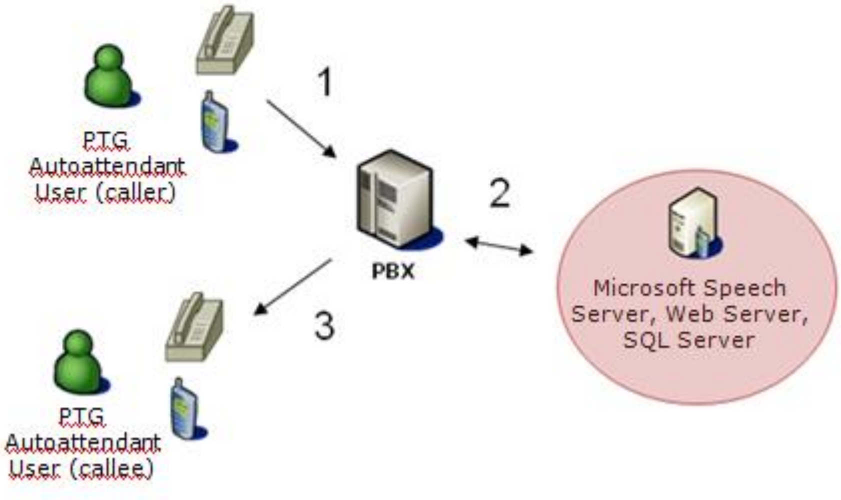

PTG Autoattendant User (caller)
1
PBX
2
Microsoft Speech Server, Web Server, SQL Server
3
PTG Autoattendant User (callee)

Figure 4.19 The EP Auto-attendant system architecture

Concerning the MLDC purposes for acquiring speech corpora, this project consists of extracting a speech corpus from the previously referred Database. The principle is to use the logs as speech corpora for training and/or testing new acoustic models, to be applied in the command and control of telephony applications. To perform this task a brief study of the Speech Server Database was made. Figure 4.20 shows the Entity Relations among the tables from where the corpus was retrieved.

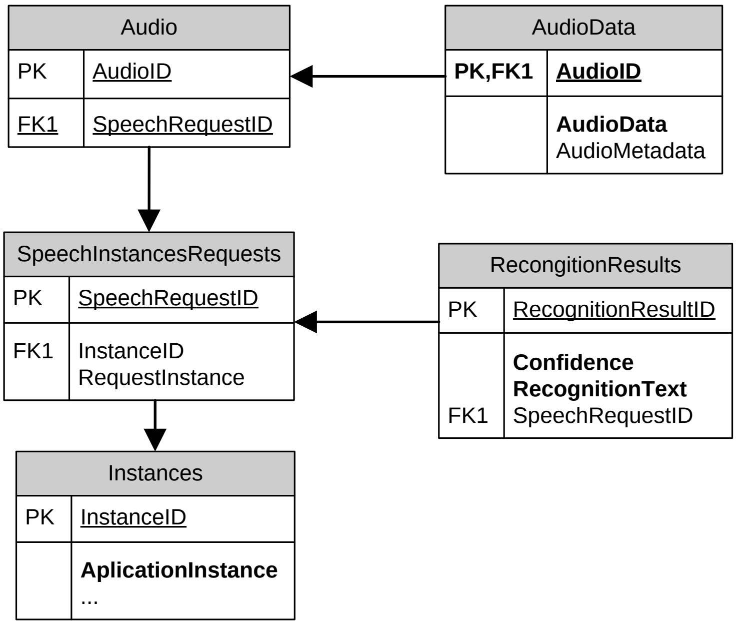


Figure 4.20 Entity relationship diagram

The columns *AudioData* and *RecognitionText* correspond to the audio and respective transcriptions identified by a unique *AudioID.* The column *AplicationInstance* determines the session number of the record. The following query represents the request of the Database.

```
SELECT  AudioData.AudioId, AudioData.AudioData, RecognitionResults.RecognitionText,
             RecognitionResults.Confidence, Instances.ApplicationInstance
FROM    Audio INNER JOIN AudioData ON Audio.AudioId = AudioData.AudioId INNER JOIN
          SpeechInstanceRequests ON Audio.SpeechRequestId = SpeechInstanceRequests.SpeechRequestId
          INNER JOIN  RecognitionResults ON
          SpeechInstanceRequests.SpeechRequestId = RecognitionResults.SpeechRequestId
          INNER JOIN Instances ON SpeechInstanceRequests.InstanceId = Instances.InstanceId
WHERE     (RecognitionResults.Confidence > - 1)
```

The audio files were retrieved using an application, *FetchAudio*, which communicates with the database, extracts the audio and its transcriptions and generates a Hyp file (transcription file). After the extraction the author realised that parts of the prompts of original operator were also recorded in each file.

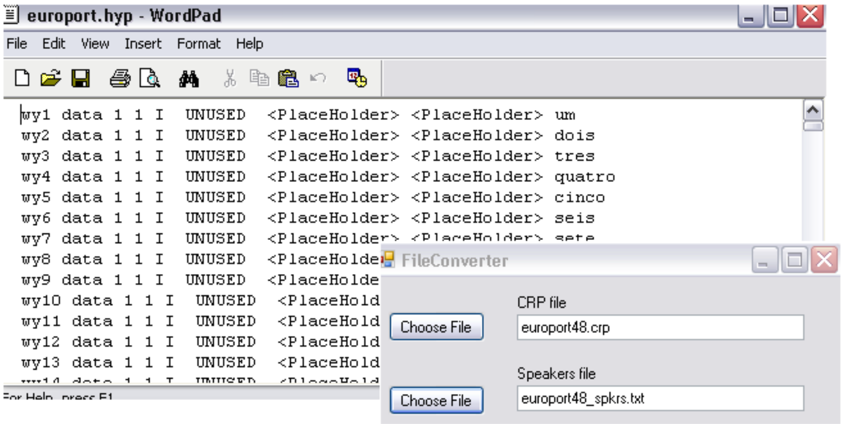
To solve this problem MLDC used *MatLab* tool [55]. In a graph, which represents the amplitude of the signal along the time, the significant audio part represents the interval where the maximum amplitude values are observed. Applying a derivative to the speech signal we can get the interval where the variation is the highest one. This interval represents the start of the actual user's voice so all the previous audio data can be cut.

Afterwards all audio files were listened and verified by the author considering audio noises, glitches, transcription and speakers' information. To improve the performance of this task it was used an application (developed by MLDC), conceived to listen to each audio file and allowing the user to correct there problems. The retrieved corpus is a telephony corpus with 2393 utterances organized by 1246 sessions and about 45 minutes of speech. The majority of speakers are male.

4.4 PHIL48

The PHIL48 was the first corpus to be prepared and trained from the scratch by the author's project. It is a telephony corpus with 20 521 utterances, 3 hours and 20 minutes of speech and 7 hours of sound. The audio files are sampled at 8 kHz for 16 bits linear. Firstly it was accomplished an application to generate the correspondent Hyp file. Figure 4.21 shows an execution example of *FileConverter*.

- *FileConverter* – it generates a Hyp file using two input files, the .crp file (audio file transcription and respective location of the audio file) and a .txt file (informs about speakers and respective audio files).



```
wy1 data 1 1 I  UNUSED  <PlaceHolder> <PlaceHolder> um
wy2 data 1 1 I  UNUSED  <PlaceHolder> <PlaceHolder> dois
wy3 data 1 1 I  UNUSED  <PlaceHolder> <PlaceHolder> tres
wy4 data 1 1 I  UNUSED  <PlaceHolder> <PlaceHolder> quatro
wy5 data 1 1 I  UNUSED  <PlaceHolder> <PlaceHolder> cinco
wy6 data 1 1 I  UNUSED  <PlaceHolder> <PlaceHolder> seis
wy7 data 1 1 I  UNUSED  <PlaceHolder> <PlaceHolder> sete
wy8 data 1 1 I  UNUSED  <PlaceHolde
wy9 data 1 1 I  UNUSED  <PlaceHolde
wy10 data 1 1 I  UNUSED  <PlaceHold
wy11 data 1 1 I  UNUSED  <PlaceHold
wy12 data 1 1 I  UNUSED  <PlaceHold
wy13 data 1 1 I  UNUSED  <PlaceHold
wy14 data 1 1 I  UNUSED  <PlaceHold
```

CRP file: europort48.crp

Speakers file: europort48_spkrs.txt

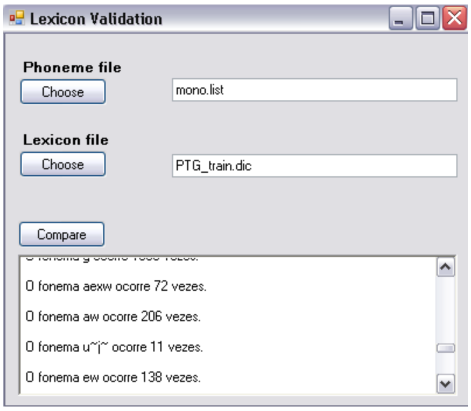
Figure 4.21 FileConverter - execution example

Then all the required input files were created such as the Autotrain XML inputs files, pronunciation lexicon, the phone set for Portuguese language and the correspondent question set file. At this point the training recipe can be run.

4.5 Other Applications

Other small applications were developed in order to help the data preparation before training the acoustic models.

- *LexiconValidation* - The purpose is to compare the two input files (the list of correct phones and the word list with the correspondent phone) and give as result how many times each phone occurs and which words have invalid phones. Figure 4.22 shows an example of application execution.



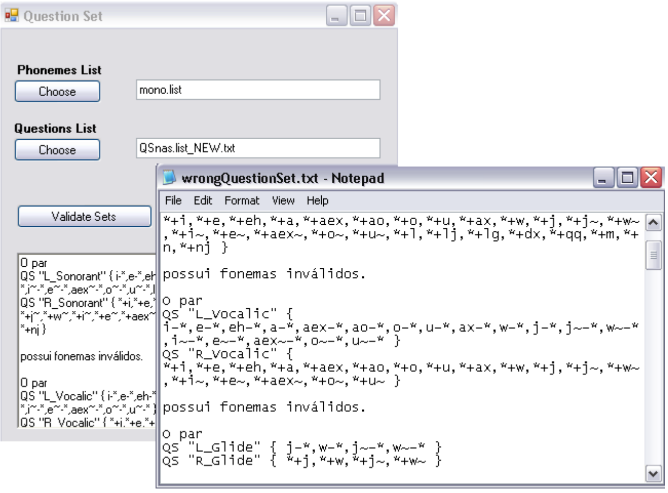
Phoneme file: mono.list

Lexicon file: PTG_train.dic

```
O fonema aexw ocorre 72 vezes.
O fonema aw ocorre 206 vezes.
O fonema u~j~ ocorre 11 vezes.
O fonema ew ocorre 138 vezes.
```

Figure 4.22 LexiconValidation - execution example

- *QuestionSet* - This application validates the sets of phonemes in terms of their structure, size and correctness. The result is a list of questions sets, where these characteristics are not respected. Figure 4.23 shows an application execution example.



mono.list

QSnas.list_NEW.txt

```
O par
QS "L_Sonorant" { i-*,e-*,eh
*,j~-*,e~-*,aex~-*,o~-*,u~-*,l
QS "R_Sonorant" { *+i,*+e,*
*+j~,*+w~,*+i~,*+e~,*+aex~
*+nj }

possui fonemas inválidos.

O par
QS "L_Vocalic" { i-*,e-*,eh-*
*,j~-*,e~-*,aex~-*,o~-*,u~-* }
QS "R_Vocalic" { *+i,*+e,*+
```

```
*+i,*+e,*+eh,*+a,*+aex,*+ao,*+o,*+u,*+ax,*+w,*+j,*+j~,*+w~
,*+i~,*+e~,*+aex~,*+o~,*+u~,*+l,*+lj,*+lg,*+dx,*+qq,*+m,*+
n,*+nj }

possui fonemas inválidos.

O par
QS "L_Vocalic" {
i-*,e-*,eh-*,a-*,aex-*,ao-*,o-*,u-*,ax-*,w-*,j-*,j~-*,w~-*
,i~-*,e~-*,aex~-*,o~-*,u~-* }
QS "R_Vocalic" {
*+i,*+e,*+eh,*+a,*+aex,*+ao,*+o,*+u,*+ax,*+w,*+j,*+j~,*+w~
,*+i~,*+e~,*+aex~,*+o~,*+u~ }

possui fonemas inválidos.

O par
QS "L_Glide" { j-*,w-*,j~-*,w~-* }
QS "R_Glide" { *+j,*+w,*+j~,*+w~ }
```

Figure 4.23 QuestionSet - execution example

4.6 Conclusion

This chapter describes several enrolled activities that involve telephonic corpora acquisition. To accomplish that goal some tools were developed to fetch and validate the collected speech corpora. Annotation and manual transcription tasks were involved in the SIP project.

All these tasks require a lot of time and several quality control stages. Their accuracy is essential, a good data acquisition and preparation is reflected in the performance of good recognizers.

Chapter 5

Conclusion

This work has explored several aspects of non-native speech and its interaction with the acoustic modelling components of a speech recognizer. The main goal of this thesis has been to understand the ways in which the speech of non-native speakers of the English language, as a group, may differ from that of native speakers, and to attempt to modify a speech recognizer to handle non-native speech input better. This work was based on an experimental corpus of English spoken by European Portuguese speakers in the MS Speech Recognition Engine domain. The following sections summarize our main results and suggest directions for future work.

5.1 Summary

In the area of acoustic modelling, we have used several methods that have shown a reduction in the word error rate on non-native speech recognition. At the beginning we have explored the behaviour of an English native system (using the English phone set) when it was tested with native and non-native data. The English native model has shown a drop on performance when tested with English spoken by Portuguese speakers' data. The recognition scores reveal 4.09% WER for native data test set and 13.41% WER for non-native data test set. HMMs can be improved retraining with suitable additional data. Pursuing that aim a recognizer with English native utterances as well as the English spoken by Portuguese speakers was trained. The resultant model was tested with non-native and native model and has 6.02% WER and 4.17% WER respectively.

Adaptation techniques can be used to reduce the mismatch between native models and the adaptation data. The most used techniques are the Maximum Likelihood Linear Regression MLLR and the MAP. MLLR technique computes a set of transformations to shift the component meanings and to alter the variances in the initial system so that each state in the HMM system can be more likely to generate the adaptation data. In MAP adaptation a prior knowledge about the model parameter distribution. These parameters are re-estimated individually and they required more adaptation data to be effective. Considering the size of the available corpora, the MLLR method was chosen for the adaptation using the non-native training data. Results reveal an increase on WER, 6.21% for non-native test and 13.41% on native test, when compared with the results of baseline system.

Pronunciation adaptation was also studied through adequate correspondences between phone sets of the foreign and target languages. The result was a new phone set, consequence of the mapping between the English and the Portuguese phone sets. This was achieved by a linguistic expert who defined a phoneme inventory bearing in mind the pronunciation of a Portuguese prestigious group/community. Using the SAMPA phone format, 33 phones were defined to transcribe the English language spoken by native Portuguese speakers. The initial English phonetic transcription was first mapped to the Portuguese form, and then each word transcription was modified by a linguist for a reliable representation of the Portuguese accent. Results reveal 7.26% of WER for new trained models using the phone set described above and tested with the non-native data.

A new pooled model was also obtained using the new phone set and it was trained with a mixture of Portuguese native data and non-native data. Consequently it was obtained a recognition system of European Portuguese that also supports English words with Portuguese accent. The results were encouraging, as the system reveals 9.81% of WER when it was tested by the non-native test set.

Even though some applied methods reveal worst performance results when compared to the baseline non-native trained models, there are considerable improvements in the English native models recognizing non-native accents. This work exposes that a small amount of data can be successfully used for the improvement in non-native accent recognition, which is potentially useful for the development of speech recognition systems in domains in which non-native data are limited.

In the area of corpora acquisition the author was involved in two projects: SIP and EP Auto-attendant, whose participation has included retrieving new telephonic data and data preparation, to train new acoustic models within the OVA domain for the Exchange 2007 Mail Server.

In the area of lexical modelling, a European Portuguese lexicon was compiled using a semi-automatic incremental method. This methodology used LTS rules for the automatic pronunciation generation in batches of 10 thousand words; each of these batches were revised and corrected by a linguistic expert. Each of these sets has also improved LTS rules after the manual revision of each 10 thousand words.

5.2 Future Work

We believe that in the chosen methods there are several possible future directions of research. An area in which further study is needed is the pronunciation of how EP speakers speak in English. A more detailed study than that one we have done may reveal some importance of foreign words spoken in a Portuguese context such as brands or software products. It would be interesting to perform an EP native recognizer that supports some English terms with the same accuracy, exploring the use of more non-native training data for those terms and expanding the pronunciation lexicon with new English words phonetically transcribed with the new phoneset. This lexicon expansion can be performed using the adopted methods (LTS rules) to generate automatic phonetic pronunciations.

In this thesis, we have also tried to achieve an improved speaker-independent baseline recognizer for non-native speakers. Although the results show that it is possible to obtain significant gains in performance by modelling all the non-native speakers as a single group, there is much accent variability within speakers to obtain a significant corpus to model it separately. Adaptation techniques have shown good results for outlining the problem. For a conversational system like Exchange 2007 Mail Server in which the speaker usually remains constant in a multi-utterance interaction, an evaluated adaptation technique supported with more adaptation data can improve recognition in the English terms scenario. Collecting new adaptation data it will be possible to retrieve new and possible results using techniques such as MAP adaptation or to apply both MLLR and MAP techniques with the aim of taking advantage of both techniques.

In short an area that was not explored was to model female corpora performance in order to compare recognition performance among the male models presented in this work. Much work is still needed to bring the recognition of non-native speech to the level of native speech recognition and any accent-specific method can help this difficult task.

Concerning the lexicon compilation and the corpora collection, these two main components for any development concerning speech-based applications, have been extended to other languages for further SR and TTS development within MLDC purposes.

Acronyms

MS – Microsoft

MSFT – Microsoft

MLDC – Microsoft Language Development Center

PEI – Projecto de Engenharia Informática

EMEA – Europe, Middle East and Africa Microsoft region

SCG - Speech Components Group, placed at Redmond

API - Application Programming Interface

SAPI - Speech Application Programming Interface

HTK - Hidden Markov Model Toolkit

HMM – Hidden Markov Model

IPA – International Phonetic Alphabet

SAMPA – Speech Assessment Methods Alphabet

TTS - Text to Speech

MLF - Master Label File

MMF – Master Macros File

PM – Program Manager

SR – Speech Recognition

OS – Operating System

MLLR – Maximum Likelihood Linear Regression

MAP – Maximum a Posteriori

EP – European Portuguese

LTS – Letter-to-Sound

CFG – Context Free Grammars

SIP – Speech International Program

LVCSR – Large Vocabulary Continuous Speech Recognition

FFT – Fast Fourier Transform

MFCC - Mel-Frequency Cestrum Coefficients

Bibliography

[1] Morton, R.: The Training Guide, A guide to training Acoustic Models. Internal Microsoft Document

[2] Young, S., Kershaw, D., Odell, J., Ollason, D., Valtchev, V., Woodland, P.: The HTK Book (1999)

[3] Teixeira, C., Trancoso, I., Serralheiro, A.: Recognition of Non-Native Accents. In Eurospeech, vol. 5, pp. 2375–2378 (1997)

[4] Teixeira, C., Trancoso, I.: Word Rejection using Multiple Sink Models. In: Proc. ICSLP, pp. 1443–1446, Banff (1992)

[5] Teixeira, C.: Reconhecimento de Fala de Oradores Estrangeiros. PhD Thesis, Universidade Técnica de Lisboa, (1998)

[6] Huang, X., Acero, A., Hon, H.-W.: Spoken Language Processing: a guide to theory, algorithm, and system development. Prentice Hall, (2001)

[7] Tomokiyo, L. M,: Recognizing Non-native Speech: Characterizing and Adapting to Non-native Usage in Speech Recognition. Ph.D. thesis, Carnegie Mellon University, (2001)

[8] Lee, K., et al.: Speaker-Independent Phone Recognition Using Hidden Markov Models. IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 31 nº 11, (1989)

[9] Glass, J. R., Hazen, T. J.: Telephone-based Conversational Speech Recognition in the Jupiter Domain. In Proc. ICSLP '98, pages 1327-1330, Sydney, Australia, (1998)

[10] Alleva, F., et al.: Can Continuous Speech Recognizers Handle Isolated Speech?. Speech Communication, pp. 183-189, (1998)

[11] Trancoso, I., Moore, R.: Tutorial and Research Workshop on Speech under Stress. Proceedings of the ESCA Nato, ESCA-ETWR INESC, Lisboa

[12] Humphries, J., Woodland, P., Pearce, D.: Using Accent-specific Pronunciation Modelling for Robust Speech Recognition. In Proc. ICSLP '96, pages 2324-2327, Philadelphia, (1996)

[13] Flege, J. E., Schirru, C., MacKay, I.: Interaction between the native and second Language Phonetic Subsystems. Speech Communication 467–491 (2003)

[14] Bartkova, K., Jouvet, D.: Multiple models for Improved Speech Recognition for Non-native Speakers. In: SPECOM (2004)

[15] Leeuwen, D. A., Orr, R.: Speech recognition of Non-native Speech Using Native and Non-native acoustic models. In MIST, (1999)

[16] Huang, C., Chen, T., Li, S., Chang, E., Zhou, J.L.: Analysis of speaker variability. Proc. European Conference on Speech Communication and Technology vol. 2, pp. 1377–1380, Denmark, (2001)

[17] Witt, S., Young, S.: Offline Acoustic Modelling of Non-native Accents. In Proc. Eurospeech, (1999)

[18] Witt, S., Young, S.: Language Learning Based on Non-native Speech Recognition. In Proc. Eurospeech, Rhodes, (1997)

[19] Boll, S.: Signal Suppression of Acoustic Noise in Speech Using Spectral Subtraction. IEEE Transactions on Acoustics, Speech and Signal Processing, ASSP-27:113-120, (1979)

[20] Morgan, N., Bourlard, H.: Continuous Speech Recognition Using Multilayer Perceptrons with Hidden Markov Models. In Proc. Int. Conf. on Acoustic Speech and Signal Processing, volume 1, pp. 413-416, Albuquerque, (1990)

[21] Widrow, B., Glover, J. R., McCool, J. M.: Adaptive Noise Cancelling: Principles and Applications. Proceedings IEEE, 63:1692-1716, (1975)

[22] Sakoe, H., Chiba, S.: Dynamic Programming Algorithm Optimization for Spoken Word Recognition. IEEE Trans. on Acoustics, Speech and Signal Processing, pp.43-49, (1978)

[23] Zavaliagkos, G., et al.: A Hybrid Segmental Neural Net/Hidden Markov Model System for Continuous Speech Recognition. IEEE Trans. on Speech and Audio Processing, pp. 151-160, (1994)

[24] Rabiner, L. R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE 77(2), 257–286 (1989)

[25] Fung, P., Liu, W.K.: Fast Accent Identification and Accented Speech Recognition.

Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 221–224, (1999)

[26] Fischer, V., Gonzalez, J., Janke, E., Villani, M., Waast-Richard, C.: Towards Multilingual Acoustic Modeling for Large Vocabulary Continuous Speech Recognition. In Proc. of the IEEE Workshop on Multilingual Speech Communications, Kyoto, Japan, (2000)

[27] Fischer, V., Janke, E., Kunzmann, S., Ross, T.: Multilingual Acoustic Models for the Recognition of Non-native Speech. In Proceedings of the Automatic Speech Recognition and Understanding Workshop, (2001)

[28] Kunzmann, S., Fischer, V., Gonzalez, J., Emam, O., Gunther, C., Janke, E.: Multilingual Acoustic Models for Speech Recognition and Synthesis. IEEE, ICASSP, (2004)

[29] Leggetter, C. J., Woodland, P. C.: Speaker Adaptation of HMMs Using Linear Regression, (1994)

[30] Kohler, J.: Multi-lingual Phoneme Recognition Exploiting Acoustic-phonetic Similarities of Sounds. Proc. Int. Conf. on Spoken Language Processing, pp. 2195-2198, Philadelphia, (1996)

[31] Nguyen, P., Gelin, P., Hunqua, J.C., Chien, J.T.: N-best Based Supervised and Unsupervised adaptation for Native and Non-native speakers in Cars. IEEE Proceedings, vol 1, (1999)

[32] Gauvain, J.L., Lee, C.H.: Maximum a Posteriori Estimation for Multivariate Gaussian Mixture Observation of Markov Chains. IEEE Transactions on Speech and Signal Processing, nº 2, pp. 291-298, (1994)

[33] Zavaliagkos, G., Schwartz, R., Makhoul, I.: Batch, Incremental and Instantaneous Adaptation Techniques for Speech Recognition. In Proc. ICASSP, (1995)

[34] Livescu, K., Glass, J.: Lexical Modelling of Non-Native Speech for Automatic Speech Recognition. In ICASSP, (2000)

[35] Steidl, S., Stemmer, G., Hacker, C., Nöth, E.: Adaptation in the Pronunciation Space for Non-Native Speech Recognition. In ICSLP, Korea, (2004)

[36] Baum, L. E., Petrie, T.: Statistical Inference for Probabilistic Functions of Finite-

state Markov Chains. Annuals of Mathematical Statistics, 37(6), 1554–1563, (1966)

[37] Baum, L. E.: An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of Markov Processes. In Shisha, Inequalities III: Proceedings of the Third Symposium on Inequalities, University of California, Los Angeles, pp. 1–8., Academic Press, (1972).

[38] Viterbi, A. J.: Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. IEEE Trans. on Information Theory, 13(2), pp. 260-269, (1967)

[39] Jurafsky, D., Martin, J.: Speech and Language Processing, An introduction to Natural Language Processing, Computational Linguistics and Speech Recognition, (2000)

[40] Ward, W.: The CMU Air Travel Information Service: Understanding Spontaneous Speech. In Proc. DARPA, Speech an Natural Language Understanding Workshop, (1990)

[41] Lippmann, R. P., Gold, B.: Neural-net Classifiers Useful for Speech Recognition. In IEEE International Conference on Neural Networks, (1987)

[42] Yu, H.J., Oh, Y.H.: A Neural Network for 500 Vocabulary Word Spotting Using Acoustic Subword Units. In Proc. Int. Conf. on Acoustic Speech and Signal Processing, Munique, (1997)

[43] Strik, H., Cucchiarini, C.: Modeling Pronunciation Variation for ASR: Overview and Comparison of Methods. Proc. of the Workshop Modeling Pronunciation Variation for Automatic Speech Recognition, Kerkrade, 137-144, (1998)

[44] Arslan, L.M., Hansen, J.H.L.: Selective Training in Hidden Markov Model Recognition. IEEE Transactions on Speech and Audio Processing 7(1), 46–54 (1999)

[45] Yakima Medium-level Description. The Yakima Speech Recognition Engine. Internal Microsoft Document

[46] Jelinek, F.: A fast sequential decoding algorithm using a stack. IBM Research Journal of Research and Development, (1969)

[47] Levinson, S. E., Rabiner, L. R., Shondi, M. M.: An introduction to the application of the theory of probabilistic function of a Markov process to automatic speech recognition. Bell Syst. Tech. J. vol 62, n°4, pp. 1035-1074, (1983)

[48] Juang, B. H.: On the Hidden Markov Model and dynamic time warping for speech recognition - A unified view. AT&T Tech. J. vol 63. N°7, pp. 1213-1243, (1984)

[49] Ferreira: Variação linguística: perspectiva dialectológica. in Faria et al. Introdução à Linguística Geral e Portuguesa, Lisboa, Caminho: 483, (1996)

[50] Wung, Z., Schultz, T., Waibel, A.: Comparison of Acoustic Model Adaptation Techniques on Non-Native Speech, IEEE. In: ICASSP (2003)

[51] Fromkin, Rodman: Introdução à Linguagem, Coimbra, Almedina: 273

[52] Breiman, L., et al.: Classification and Regression Trees. Pacific Grove, CA, Wadsworth, (1984)

[53] Davis, S., Mermelstein P.: Comparison of Parametric Representations for Monosyllable Word Recognition in Continuously Spoken Sentences, IEEE Trans. on Acoustics, Speech and Signal Processing pp. 357-366, (1980)

[54] MSDN Library for Visual Studio 2005, http://msdn2.microsoft.com/en-us/vcsharp/

[55] MatLab, http://www.mathworks.com/

[56] Audio Converter - http://sox.sourceforge.net/

[57] W3C grammar, http://www.w3.org/TR/speech-grammar/

[58] Sapi 5 grammar, http://msdn2.microsoft.com/en-us/library/ms723635.aspx

Annex 1

The Expansion of European Portuguese Lexicon

Objective

The objective of this document is to describe thoroughly the adopted process in creating a 100k word phonetic lexicon.

The main sources of this Lexicon have been two, namely:

- SpeeCon project which includes a 17 K word phonetic lexicon (whose corpora results have been licensed to Microsoft).
- 118 K World list of Natural Language Group, and it is the resource used by Office 14 Spelling tools.

Adopting a Method

The entire 118k lexicon was exported to a txt file and split into 10k word groups, so that each group could be approached separately. The first of these groups is phonetically transcribed, using LTS. A phonetic transcription is generated for each word and then manually revised. This is obviously much simpler than to add phonetic information from scratch and it allows us to monitor the improvement of LTS rules. Once the first batch of 10k words is revised, LTS rules are retrained and then used to generate the transcription of the second 10k word batch, and so forth, until we reach the final 10 k word batch. The entire process is depicted in the Figure 1.

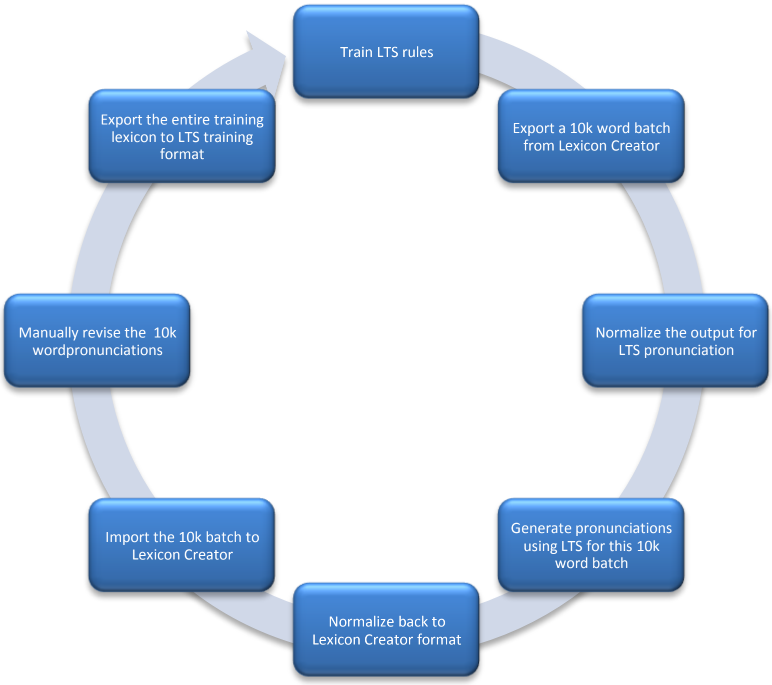

Train LTS rules
Export a 10k word batch from Lexicon Creator
Normalize the output for LTS pronunciation
Generate pronunciations using LTS for this 10k word batch
Normalize back to Lexicon Creator format
Import the 10k batch to Lexicon Creator
Manually revise the 10k wordpronunciations
Export the entire training lexicon to LTS training format

Figure 1- The lexicon creation process.

LTS Training Method

The initial lexicon, which was shipped with the EP Speecon corpus and revised by a linguist, has circa 17 thousand words. All the entries which could not be used to train LTS rules, were removed and consequently reaching the 15k words of the original lexicon.

The next step was the definition of a testing corpus with 1152 words. This corpus was selected according to the initial lexicon. It was always used with the same testing corpus for all the experiments. Thus it is possible the comparison among accuracy values, though it does not reflect the nature lexicons of the further iterations with updated lexicons.

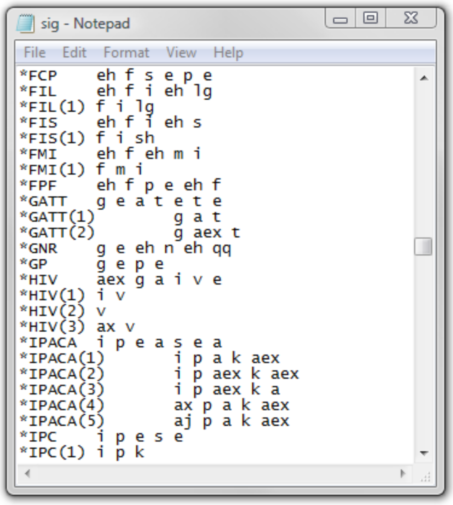
Before training the new LTS rules the lexicons are updated using the last revised words. MergeLexicon tool (see figure 4) is used to merge the underlying lexicon with the new slot of words. Table 1 shows the evolution of the Lexicon updating process from the first lexicon file to the last one.

It was adopted two training LTS methods: the CART and Graph methods. The LTS rules built by the Graph method were used to generate the new phonetic pronunciations for the new 10 thousand words on each iteration. These rules are in a runtime compilation format which requires a test compatible tool to generate the pronunciations using the new LTS rules. We use *BuildLex.exe* to obtain pronunciations of the speech engine with the new LTS rules. The use of BuildLex.exe is controlled using the following command:

- *BuildLex –vocabfn vocabulary.lst –engine "Microsoft Portuguese (Portugal Telephony) v7.0 Server" –ltsonly true –out WordListWithPronunciations.out*

This means that the word pronunciation uses the registered engine *"Microsoft Portuguese (Portugal Telephony) v7.0 Server"*

Figure 2 represents the output file of *BuildLex* execution which generates several possible phonetic pronunciations for each word in the batch of 10 thousand words.



```
*FCP    eh f s e p e
*FIL    eh f i eh lg
*FIL(1) f i lg
*FIS    eh f i eh s
*FIS(1) f i sh
*FMI    eh f eh m i
*FMI(1) f m i
*FPF    eh f p e eh f
*GATT   g e a t e t e
*GATT(1)        g a t
*GATT(2)        g aex t
*GNR    g e eh n eh qq
*GP     g e p e
*HIV    aex g a i v e
*HIV(1) i v
*HIV(2) v
*HIV(3) ax v
*IPACA  i p e a s e a
*IPACA(1)       i p a k aex
*IPACA(2)       i p aex k aex
*IPACA(3)       i p aex k a
*IPACA(4)       ax p a k aex
*IPACA(5)       aj p a k aex
*IPC    i p e s e
*IPC(1) i p k
```

Figure 2 – An example of the output execution BuildLex file

To give the output file to the linguist we have to normalize it to the adequate format of LexiconCreator. To perform it, we use the *WPronunciationFilter.exe* tool (see figure 6). The results of the test to the CART method are presented in Table 1. The accuracy tests of the LTS rules using the graph method were not performed yet.

	LTS Lexicon (# words)	LTS Training Lexicon (# words)	LTS testing Lexicon (# words)	CART Method Acuraccy tests		Graph Method Acuraccy Tests
				WER (%)	PER (%)	
1st lexicon	14 712	13 560	1 152	**9.16**	**1.96**	_
2nd	20 577	19 425	1 152	**7.37**	**2.39**	_
3rd	29 194	28 042	1 152	**6.16**	**2.02**	_
4th	35 837	34 685	1 152	**6.16**	**1.91**	_
5th	43 675	42 523	1 152	**6.77**	**2.12**	_
6th	51 657	50 505	1 152	**5.72**	**1.76**	_
7th	60 349	59 197	1 152	**5.29**	**1.72**	_
8th	67 932	66 780	1 152	**5.38**	**1.67**	_

9th	74 293	73 141	1 152	**5.29**	**1.71**	–
10th	81 758	80 606	1 152	**4.51**	**1.49**	–
11th	89 328	88 176	1 152	**5.47**	**1.68**	–
12th	101 152	**100 000**	1 152	**5.03**	**1.63**	–
13th	102 082	**100 930**	1 152	**4.77**	**1.56**	–
14th	103 510	**103 510**	1 152	–	–	–

Table 1 – Experimental Results

In the last three experiments the accuracy values start to vary non-linearly. In the last experiment the rules were not able to compile because the lexicon has exceeded the number of words.

LTS Rules

LTS rules are used to generate pronunciations for new words. The rules are mappings between letters and phones. The rules are trained using data-driven techniques which means that the formalization of a rule is based on data, in this case the correspondence between the words and its phonetic transcription.

Before training the training LTS lexicon must be prepared

- 90% - 95% of the available lexicon is selected for training
- The remaining is reserved for test performance
- Words must reflect the general phonetic pattern of the language in study
- The lexicon must not contain abbreviations, numerals, non-pronounceable acronyms or truncated/misspelled words

CART Method

Training

The first step is to align letters or groups of letters to the phonemes that represent their pronunciation. To accomplish this it is used the LTSalign.exe with the following configuration command:

- *LTSAlign config.txt LTS_TraingLexicon.dic 2070_train.smp*

To train a classification Tree it was used the following command:

- *ltstrain.exe -ni letter.sym -no phone.sym -fi letter.q -fo phone.q -p 2070_train.smp -l TREEdir*

In the compilation step it was used the following command:

- *ltscomp.exe letter.sym phone.sym letter.q phone.q TREEdir\tree.tree 0 50000 0.0 2070_train.smp w2070.lts*

The values 0 5000 0.0 correspond to the deeper level of nodes and the shallow level of nodes for generating a richer set of pronunciations. Size can be specified for both levels and for the purpose of pruning the classification tree.

Testing

For testing the LTS rules trained with this method two different tools are used. The following commands represent an example of execution:

- *ltstst.exe w2070.lts TestingFile.dic*

- *ltstst r1033.lts TestFile.dic >> outputFile.out*

LTStst.exe reads the compiled classification tree file and generates LTS pronunciations for the entries in the dictionary for testing.

- *ltsscore testFile.dic outputFile.out resultsBase*

LTSscore outputs the following three files, where *resultsbase* is the name specified as the third parameter in the command line when during the running of LTSscore:

- A results base file labelled *resultsbase*, containing raw match results;
- A confusion matrix file labelled *ressultsbase*.xls;
- A statistics file labeled *resultsbase*_stats.xls, containing statistics such as error rate;

GRAPH Method

This method generates LTS compiled rules format (.lxa). This is the process for building graphoneme LTS rules.

Training

The first step is to align the Training lexicon using the same tool as the CART method (ltsalign.exe). Then we have to generate the Graphoneme LTS using the following command:

- *LM.bat 2070_train.smp grph2070.tlm*

  The lm.bat is the batch file where all the necessary files and perl scripts are pointed to the respective files.

To build the Graphoneme LTS we need a *spell.txt* file in Unicode and a compiled phoneset *L2070.phn*. For that purpose we use the *bldlts.exe* tool:

- *bldlts.exe 2070 graphoneme grph2070.tlm spell.txt L2070.phn LTS.lxa*

  The LTS.lxa is the runtime compiled LTS rules

Testing

For testing these rules there is the LTStest.exe tool. As we have been facing some problems with the use of this tool due to incapability of the engine, we have not performed any tests yet.

Developed Tools

In this section all the tools, which were developed in the scope of this work, will be presented.

- **WPronunciationNormalizer**

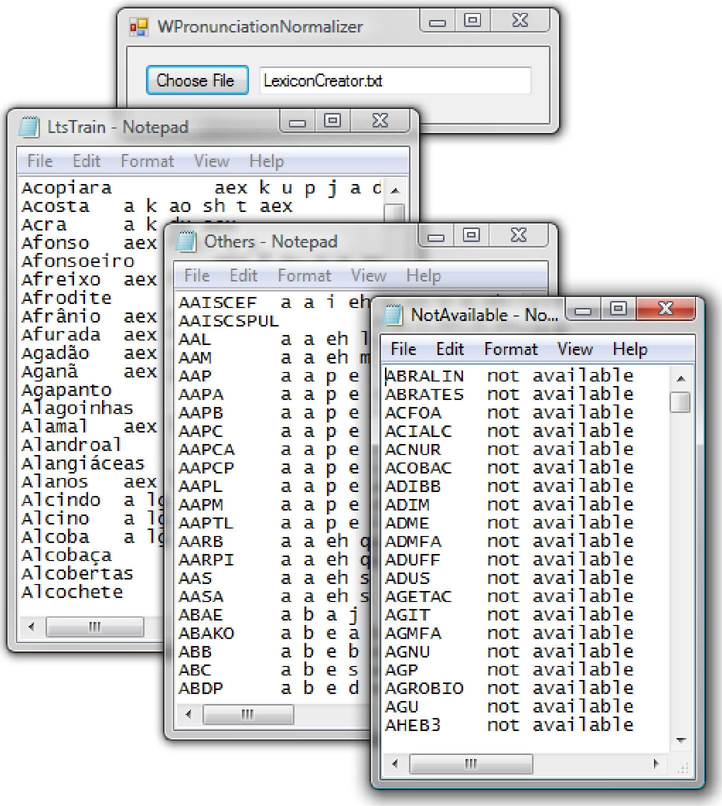
The WPronunciationNormalizer.exe is a C# application used to normalize the file containing all non-default annotations exported by Lexicon Creator. Figure 3 shows the three output files in the required format.

Input – *LexiconCreator.txt* - exported file

Outputs – *LTStrain.txt* – entries that can be used to train LTS rules.

- *Others.txt* – entries that have phonetic information and that have been verified, but cannot be used to train LTS rules.

- *NotAvailable.txt* – entries that have phonetic information but that were not verified.



```
Acopiara              aex k u p j a d
Acosta     a k ao sh t aex
Acra       a k
Afonso     aex
Afonsoeiro
Afreixo    aex
Afrodite
Afrânio    aex
Afurada    aex
Agadão     aex
Aganã      aex
Agapanto
Alagoinhas
Alamal     aex
Alandroal
Alangiáceas
Alanos     aex
Alcindo    a l
Alcino     a l
Alcoba     a l
Alcobaça
Alcobertas
Alcochete
```

```
AAISCEF   a a i eh
AAISCSPUL
AAL           a a eh l
AAM           a a eh m
AAP           a a p e
AAPA          a a p e
AAPB          a a p e
AAPC          a a p e
AAPCA         a a p e
AAPCP         a a p e
AAPL          a a p e
AAPM          a a p e
AAPTL         a a p e
AARB          a a eh q
AARPI         a a eh q
AAS           a a eh s
AASA          a a eh s
ABAE          a b a j
ABAKO         a b e a
ABB           a b e b
ABC           a b e s
ABDP          a b e d
```

```
ABRALIN  not available
ABRATES  not available
ACFOA    not available
ACIALC   not available
ACNUR    not available
ACOBAC   not available
ADIBB    not available
ADIM     not available
ADME     not available
ADMFA    not available
ADUFF    not available
ADUS     not available
AGETAC   not available
AGIT     not available
AGMFA    not available
AGNU     not available
AGP      not available
AGROBIO  not available
AGU      not available
AHEB3    not available
```

Figure 3 – WpronunciationNormalizer usage sample

- **Merge Lexicons**

This is a C# application used to merge and compare lexicons, indicating all the differences between them.

Input File1 – *MainLexicon.txt* – This lexicon will be taken as a reference on comparing both files.

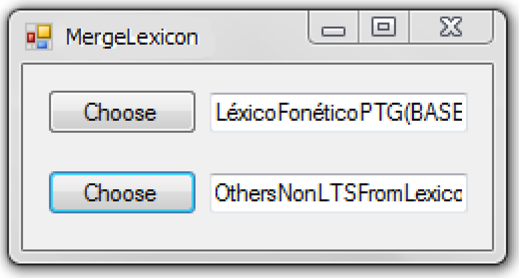
Output File1 – *LexiconWithoutRepetitions.txt*

- *RepeatedWords.txt* – All double occurrences from *Input File1*

Input File2 – *NewLexicon.txt* – This lexicon will be compared with *Input File1*.

Output File2 – *NewWords.txt* – All the entries from InputFile1 that were not encountered in *Input File2*

- *WordsOnBothLexicons.txt* – All the entries present in *Input File1* and in *Input File2*



LéxicoFonéticoPTG(BASE

OthersNonLTSFromLexicc

Figure 4 - Merge Lexicon

- **Lexicon Validation**

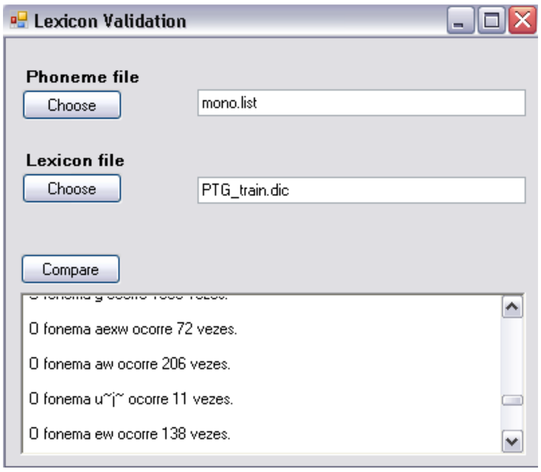
This tool compares the Phone Set with the Phonetic Lexicon, outputting the number of times each phone occurs and any invalid phone occurrence. Figure 5 shows an execution example.

Input File1 – *Mono.list* – Phone Set list*;*

Input File2 – *Lexicon.dic* – Phonetic Lexicon to be validated;

Outputs – *Statistics.txt* – Information on the number of times each phone occurs;

- *wrongPhones.txt* – Listing all the entries containing wrong phones.



Phoneme file: mono.list

Lexicon file: PTG_train.dic

```
O fonema aexw ocorre 72 vezes.
O fonema aw ocorre 206 vezes.
O fonema u~j~ ocorre 11 vezes.
O fonema ew ocorre 138 vezes.
```

Figure 5 – Lexicon Validation

- **WPronunciationFilter**

The purpose of WPronunciationFilter is to choose the best transcription of the Buildlex.exe output file.

Input File– *WordListWithPronunciations.out* – Word list containing all the phonetic transcriptions generated by BuildLex, with several possibilities for each entry.

Output – *CleanedWordList.txt* – Listing the entries and their phonetic information, with only one possibility per entry.

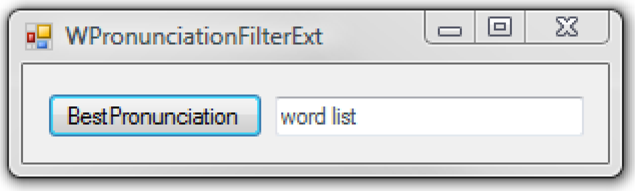


Figure 6 – WPronunciationFilter

Annex 2

Normalizing Hyp File Transcriptions

The normalization of the text consists of modifying Hyp file transcriptions to represent a language and recording characteristics in a standardized way. Standardization of transcriptions is useful to ensure that the training tools handle the transcriptions correctly so that they can correspond to audio files in a right way.

Capitalization

All the words should be converted to the upper case, including words with accented characters. If a single-case is used, transcription will make lexicon processing simpler.

Numbers, Money, Acronyms, etc.

If transcriptions contain such representations and have not been transcribed orthographically, these must be converted to their orthographically form. For example, 5 € should be transcribed “FIVE EUROS”.

Letters

Spelled words should be normalized in the transcription files to include a dot. For example:

P O R T U G A L > P. O. R. T. U. G. A. L.

Abbreviations

Spelled abbreviations may be transcribed as single words or as sequences of letters. For example:

MSFT > M. S. F. T.

Against what was spoken if the corpus was transcribed correctly there would not be any ambiguity, for example if someone says Microsoft, the transcription should contain Microsoft and never in any case its acronym.

Mispronounced words

Mispronounced words are most commonly transcribed with an asterisk (*) preceding the word. Each corpus can have its own transcription mark-up language so this may vary. For example if the speaker said Praris but meant Paris, this word can be transcribed as “*Paris”.

Noise and filler words

The HYP file also contains annotation of information, which is composed of written tags that mark or describe the speech sounds, e.g. prompt echo, background noise or speech, misspelled words, etc. Each corpus has its own list of tags for example:

- [NON] – non primary speaker
- [PE] – prompt echo
- [NPS] – non primary speaker
- [SIL] – silence

These can be left as they are with no normalization. They will be added to the lexicon as words which require a special pronunciation.

Unintelligible words

Unintelligible sections of transcriptions are often marked by ** in the transcriptions. If the file only contains unintelligible speech and nothing else, then it should be removed from the Hyp file.

Hyp File example

```
ACCEPT1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACCEPT
ACCEPT2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACCEPT
ACTIVATE1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVATE
ACTIVATE2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVATE
ACTIVE_BOOK1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVE BOOK
ACTIVE_BOOK2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ACTIVE BOOK
ADDRESS_BOOK1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ADDRESS BOOK
ADDRESS_BOOK2AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ADDRESS BOOK
ADDRESSES1AS CorpusPTINGL\AS 1 1 M TRAIN <PlaceHolder> <PlaceHolder> ADDRESSES
```

Annex 3

CFG for Accuracy Testing of Experiments

```
<grammar root="words" xml:lang="en-US" version="1.0"
xmlns="http://www.w3.org/2001/06/grammar" tag-format="semantics-ms/1.0">

       <rule id="words" scope="public">
              <one-of>
      <item>ABBREVIATED</item>
      <item>ACCEPT</item>
      <item>ACTIVATE</item>
      <item>ACTIVE</item>
      <item>ADDRESS</item>
      <item>ADDRESSES</item>
      <item>AGAIN</item>
      <item>AGENCY</item>
      <item>AID</item>
      <item>AIRPORT</item>
      <item>ALARM</item>
      <item>ALTERNATIVE</item>
      <item>ANNOTATE</item>
      <item>ART</item>
      <item>AUTOMATIC</item>
      <item>BACK</item>
      <item>BACKWARD</item>
      <item>BACKWARDS</item>
      <item>BIGGER</item>
      <item>BITMAP</item>
      <item>BOOK</item>
      <item>BOOKING</item>
      <item>BUSY</item>
      <item>BY</item>
      <item>CALCULATOR</item>
      <item>CALENDAR</item>
      <item>CALL</item>
      <item>CANCEL</item>
      <item>CANTEEN</item>
      <item>CENTRE</item>
      <item>CHECK</item>
      <item>CLOCK</item>
      <item>CLOSE</item>
      <item>COMPUTER</item>
      <item>CONFERENCE</item>
      <item>CONFIGURE</item>
      <item>CONNECT</item>
      <item>CONTENTS</item>
      <item>CONTINUE</item>
      <item>CONTOUR</item>
      <item>CONVERT</item>
      <item>COPY</item>
      <item>CUT</item>
      <item>DEACTIVATE</item>
      <item>DELETE</item>
      <item>DEPARTMENT</item>
      <item>DESTINATION</item>
      <item>DIAL</item>
      <item>DIALLING</item>
      <item>DIARY</item>
      <item>DIRECTORY</item>
      <item>DISK</item>
      <item>DIVERSE</item>
      <item>DIVERSION</item>
      <item>DIVERT</item>
      <item>DIVIDED</item>
      <item>DONE</item>
      <item>DOT</item>
      <item>DOWN</item>
      <item>ECONOMIC</item>
      <item>EDIT</item>
      <item>EDITOR</item>
      <item>EIGHT</item>
      <item>ENTERTAINMENT</item>
      <item>EQUALS</item>
      <item>ERASE</item>
      <item>EXCHANGE</item>
      <item>EXTEND</item>
      <item>FACSIMILE</item>
```

```
<item>FILE</item>
<item>FIND</item>
<item>FIRST</item>
<item>FIVE</item>
<item>FIXED</item>
<item>FOLLOW</item>
<item>FORWARD</item>
<item>FOUR</item>
<item>FRIDAY</item>
<item>GENERAL</item>
<item>GO</item>
<item>HANG</item>
<item>HELP</item>
<item>HOSPITAL</item>
<item>HOST</item>
<item>HYPERLINKS</item>
<item>IMMEDIATE</item>
<item>IMPORT</item>
<item>IN</item>
<item>INDEX</item>
<item>INSERT</item>
<item>INTERNATIONAL</item>
<item>INTERRUPTION</item>
<item>KEYBOARD</item>
<item>KEYPAD</item>
<item>LAST</item>
<item>LEFT</item>
<item>LIBRARY</item>
<item>LINK</item>
<item>LOAD</item>
<item>LOCAL</item>
<item>LOTTERY</item>
<item>MAIN</item>
<item>MAKE</item>
<item>MANAGER</item>
<item>MEETING</item>
<item>MENU</item>
<item>MESSAGES</item>
<item>MICRO</item>
<item>MINUS</item>
<item>MIRROR</item>
<item>MISCELLANEOUS</item>
<item>MODE</item>
<item>MONDAY</item>
<item>MOVE</item>
<item>MULTIPLIED</item>
<item>NATIONAL</item>
<item>NEXT</item>
<item>NINE</item>
<item>NO</item>
<item>NOTEBOOK</item>
<item>NOTEPAD</item>
<item>NUMBER</item>
<item>OH</item>
<item>OKAY</item>
<item>ONE</item>
<item>OPEN</item>
<item>OPERATOR</item>
<item>OPTIONS</item>
<item>OTHER</item>
<item>OUT</item>
<item>OUTGOING</item>
<item>OUTPUT</item>
<item>PAGE</item>
<item>PASTE</item>
<item>PEN</item>
<item>PERSON</item>
<item>PERSONNEL</item>
<item>PHONE</item>
<item>PLUS</item>
<item>POINT</item>
<item>POLITICAL</item>
<item>PREVIOUS</item>
<item>PRINT</item>
<item>PROGRAM</item>
<item>PURCHASING</item>
<item>QUIT</item>
```

```
      <item>RAILWAY</item>
      <item>RECORDER</item>
      <item>REDIAL</item>
      <item>REDISPLAY</item>
      <item>REDO</item>
      <item>REPEAT</item>
      <item>REPLY</item>
      <item>RESTRICTION</item>
      <item>RETURN</item>
      <item>RIGHT</item>
      <item>ROOM</item>
      <item>RUB</item>
      <item>RUBBER</item>
      <item>SATURDAY</item>
      <item>SAVE</item>
      <item>SEARCH</item>
      <item>SECURITY</item>
      <item>SELECTOR</item>
      <item>SEND</item>
      <item>SET</item>
      <item>SETTINGS</item>
      <item>SETUP</item>
      <item>SEVEN</item>
      <item>SHIFT</item>
      <item>SHOW</item>
      <item>SIX</item>
      <item>SMALLER</item>
      <item>SOUND</item>
      <item>SPLIT</item>
      <item>SPORT</item>
      <item>SPREADSHEET</item>
      <item>START</item>
      <item>STARTING</item>
      <item>STATION</item>
      <item>STOCK</item>
      <item>STOP</item>
      <item>STYLUS</item>
      <item>SUNDAY</item>
      <item>SYSTEM</item>
      <item>TAXI</item>
      <item>TECHNICAL</item>
      <item>TELEPHONE</item>
      <item>TELEVISION</item>
      <item>TEN</item>
      <item>TEXT</item>
      <item>THAT</item>
      <item>THE</item>
      <item>THREE</item>
      <item>THURSDAY</item>
      <item>TIME</item>
      <item>TIMES</item>
      <item>TO</item>
      <item>TRAINING</item>
      <item>TRANSFER</item>
      <item>TRANSFORMATION</item>
      <item>TRAVEL</item>
      <item>TRAY</item>
      <item>TUESDAY</item>
      <item>TWO</item>
      <item>UNDO</item>
      <item>UP</item>
      <item>VARIOUS</item>
      <item>VOICE</item>
      <item>WAKE</item>
      <item>WALK</item>
      <item>WEATHER</item>
      <item>WEDNESDAY</item>
      <item>WITH</item>
      <item>YES</item>
      <item>ZERO</item>
<item>ABBREVIATED DIALLING</item>
<item>ACTIVE BOOK</item>
<item>ADDRESS BOOK</item>
<item>ALARM CALL</item>
<item>ALARM CLOCK</item>
<item>BITMAP EDITOR</item>
<item>CALL THE</item>
```

```
<item>COMPUTER CENTRE</item>
<item>CONFERENCE WITH</item>
<item>CONFERENCE WITH THE</item>
<item>DIVERT TO</item>
<item>DIVERT TO THE</item>
<item>DIVIDED BY</item>
<item>FILE SYSTEM</item>
<item>FIRST AID</item>
<item>FIXED DESTINATION</item>
<item>FOLLOW LINK</item>
<item>GO BACK</item>
<item>HANG UP</item>
<item>IN TRAY</item>
<item>MAIN MENU</item>
<item>MAKE LINK</item>
<item>MEETING ROOM</item>
<item>MULTIPLIED BY</item>
<item>NO REPLY</item>
<item>OUTGOING RESTRICTION</item>
<item>PERSONNEL DEPARTMENT</item>
<item>PURCHASING DEPARTMENT</item>
<item>RAILWAY STATION</item>
<item>RUB OUT</item>
<item>SET ALARM</item>
<item>SET CLOCK</item>
<item>SHIFT LEFT</item>
<item>SHIFT RIGHT</item>
<item>SHOW ROOM</item>
<item>STARTING POINT</item>
<item>START MICRO</item>
<item>START VOICE</item>
<item>STOCK EXCHANGE</item>
<item>STOP MICRO</item>
<item>STOP VOICE</item>
<item>TECHNICAL MANAGER</item>
<item>TEXT EDITOR</item>
<item>TRANSFER TO</item>
<item>TRANSFER TO THE</item>
<item>TRAVEL AGENCY</item>
<item>WAKE UP</item>
        <item>D. N. C.</item>
        <item>D. X. F.</item>
        <item>M. I.</item>
        <item>I. O.</item>

          </one-of>
        </rule>
</grammar>
```

Annex 4

Phone set European Portuguese/English

Author: Daniela Braga
Role: Linguist Expert
Plosives/stop consonants:

EP	SAMPA EN	PT_EN	Example
p	p	p	pen
t	t	t	tea
k	k	t	cat
b	b	b	bad
d	d, D	d	did, that, they
g	g	g	get, sing

Affricates

EP	SAMPA EN	PT_EN	Example
-	tS	Ch t sh	chair, chairman
-	dZ	Jh d zh	January, jackpot

Fricatives

EP	SAMPA EN	PT_EN	Example
f	f	f	fall
v	v	v	van
-	T	th[4]	thin, thief
s	s	s	see, start, texts
z	z	z	zip, terrains
sh	S	sh	corporation, selfish
zh	Z	zh	vision, Asia
-	h	h[5]	hi-fi,

Nasals

EP	SAMPA EN	PT_EN	Example
m	m	m	man
n	n	n	now, sing
nj		-	*sonho*
-	N	-[6]	sing

[4] O Dicionário da Academia das Ciências de Lisboa (DACL) não reconhece a fricativa interdental nas pronúncias portuguesas de <thriller>, <thirties>, convertendo-a numa oclusiva dental [t]. Eu não concordo que seja a articulação de prestígio entre portugueses que falam Inglês.

[5] A articulação nula desta fricativa aspirada está atestada no DACL, mas eu penso que a articulação de prestígio por portugueses realiza esta consoante.

[6] Ladefoged (2001: 54) diz que fonologicamente no Inglês se pode considerar esta nasal única [ng] como uma sequência de dois fonemas /n/ e /g/. Assim, resolvi aproveitar esses fonemas do EP.

Approximants

EP	SAMPA EN	PT_EN	Example
l	l	l	leg, call, hello
lj		-	*alho*
nj		-	
dx	r	r	red, far, prediction
qq		-	carro
j	j	j	yard
w	w	w	wet

Vowels

EP	SAMPA EN	PT_EN	Example
i	i	i	see, sea, sexy
i	I	i	sin, sing, sit
eh	e@	eh	hair, Terek, ten
aex	3:	aex	fur, heard
-	eI	e j	scale, say, raise
i		i	saying, scenario, simply
-	@U	ow	fellow, scenario, bone, gold
	OI	aoj	noise, boy, loyal
u	u:	u	good, too, lubricant, mature
	U@	J u	Mathew, news
ao	Q	ao	God, star, father, guard, doctor
eh	{	eh	mad, cat, parrot
aex	V	aex	cup, peanut, rough
ao	Q	ao	dog, board
	aU	aw	house, about, rouse
aex	@	aex	router, Boston, Bosnia, another
	aI	aj	flight, rise, my
i~			
e~			
6~			
o~			
u~			
a			
e			

Comentários:

1. Há opiniões diferentes em relação ao número de vogais consideradas para o Inglês; ao contrário do Português, em que as vogais se dividem essencialmente em orais e nasais, no Inglês elas são breves ou longas. Ladefoged considera 15 (em American English) (p.74) divididas em 10 tensas (*tense*) e 5 relaxadas (*lax*) (pp.80-81). Em Sampa contam-se 20 (7 breves “checked” e 13 longas “free”).

2. Fez-se o “matching” dos dois phone sets tendo como referência o phone set do Português Europeu, uma vez que o objectivo do trabalho é reconhecimento de fala de falantes portugueses a falar Inglês.

3. **Número total de fonemas**: **33** (a nível vocálico há várias neutralizações – realizações iguais de fonemas diferentes do Inglês; não há vogais nasais nem ditongos nasais em Inglês)

	PT_EN
1	p
2	t
3	t
4	b
5	d
6	g
7	ch
8	jh
9	f
10	v
11	th
12	s
13	z
14	sh
15	zh
16	h
17	m
18	n
19	l
20	r
21	j
22	w
23	i
24	eh
25	aex
26	ej

27	ow
28	aoj
29	u
30	ju
31	ao
32	aw
33	aj

References:

- Casteleiro, M. (coord.) 2001,Dicionário da Academia das Ciências de Lisboa, Lisboa: Verbo.
- Wehmeier, S. (editor) 2005, Oxford Advanced Learner's Dictionary (7th Edition), Oxford: Oxford University Press.
- SAMPA for English: http://www.phon.ucl.ac.uk/home/sampa/english.htm
- Ladefoged, P. 2001, A course in Phonetics (4th Edition), Boston: Heinle & Heinle.

Annex 5