Electronics and Computer Engineering Dept.

Telecommunications Division

Technical University of Crete

AUTOMATIC SPEECH TRANSCRIPTIONS OF GREEK BROADCAST NEWS

zafeiris malafouris

Supervisor: Alexandros Potamianos

Co-supervisor: Vasilios Digalakis

Committee member: Athanasios Liavas

Chania, June 2007

Zafeiris Malafouris, “Automatic Speech Transcriptions of Greek Broadcast News”, *Submitted in partial fulfillment of the requirements for the degree of Engineer at the Technical University of Crete* , © Greece, June 2007

AUTOMATIC SPEECH TRANSCRIPTIONS OF GREEK BROADCAST NEWS

June 2007

Technical University of Crete
Department of Electronics and Computer Engineering
Telecommunications Laboratory
University Campus - Kounoupidiana
731 00 Chania
Hellas (Greece)

SUPERVISORY COMMITTEE

Supervisor: Alexandros Potamianos

Co-supervisor: Vasilios Digalakis

Committee member: Athanasios Liavas

ACKNOWLEDGMENTS

Ας μου επιτραπεί το κομμάτι των ευχαριστιών, σε αντίθεση με την υπόλοιπη διπλωματική, να γραφτεί στα Ελληνικά, στο πρώτο ενικό και σε πιο 'χαλαρή' (informal) γλώσσα.

Αρχικά λοιπόν, και όπως συνηθίζεται άλλωστε, θα ήθελα να ευχαριστήσω τον κύριο Ποταμιάνο, τον επιβλέποντα καθηγητή μου, που με τις γνώσεις αλλά πάνω από όλα με τη θέληση του βοήθησε ώστε να έχουμε μία άψογη συνεργασία (ελπίζω και ο ίδιος να έχει την ίδια άποψη...) και η όλη δουλειά να κυλίσει ομαλά και σε εύλογο χρονικό διάστημα. Κι όταν λέω εύλογο, δεν εννοώ τον ενάμιση χρόνο που πέρασε από τότε που μου έδωσε το θέμα μέχρι να έρθει η πολυπόθητη στιγμή της παρουσίασης του, αναφέρομαι στους 7 μήνες πραγματικής δουλειάς που έγινε.

Το ότι το έσκασα στην Ιβηρική, για παράδειγμα, για πάνω από ένα μήνα δεν πρέπει να το μετράμε, παρόλο που πήγα για να προετοιμαστώ για τη διπλωματική και να δω την αντίστοιχη δουλειά που είχαν κάνει οι ισπανοί φοιτητές. Να ευχαριστήσω εδώ το παρεάκι του ιβηρικού ταξιδιού, έλληνες και ξένους, για τις μοναδικές εμπειρίες που μου χάρισαν, που σίγουρα θα αποτελέσουν εφόδιο για την ζωή, καθώς και τις πανέμορφες αναμνήσεις που θα με κρατάνε πάντα αισιόδοξο στις δύσκολες ώρες.

Ευχαριστώ επίσης την ομάδα του εργαστηρίου για την πολύτιμη βοήθεια της, από τις μικρές μικρές συμβουλές σε διάφορα εργαλεία λογισμικού μέχρι τους καφέδες που πάντα υπήρχαν όποτε χρειάστηκε, αλλά και τα ξενύχτια που κάναμε στο εργαστήριο δουλεύοντας.

Δεν θα παρέλειπα να ευχαριστήσω τους φίλους μου, τις παλιοσειρές, που μαζί τους πέρασα ως τώρα πολλές ωραίες στιγμές, αυτές που απλά έδωσαν νόημα σε κάθε κόπο, κούραση, τσαντίλα που εμφανίστηκαν κατά τη διάρκεια του δρόμου προς το πτυχείο.

Ευχαριστώ επίσης την BESTοπαρέα για τα όμορφα ταξίδια που κάναμε, για τις εκδηλώσεις που διοργανώσαμε μαζί, τις εποικοδομητικές κουβέντες που είχαμε αλλά και για τα άτομα που γνώρισα μέσω αυτού του συλλόγου, εντός και εκτός συνόρων.

Τέλος, και πάνω από όλα, θα ήθελα να ευχαριστήσω τους γονείς μου για την συμπαράσταση όλα αυτά τα χρόνια, αλλά και τον αδερφό μου τον Ευθύμη που με επισκεπτόταν συχνά πυκνά και ήταν εκεί σε ωραίες και δύσκολες στιγμές.

ABSTRACT

This Diploma Thesis has been developed in partial fulfillment of the requirements for the Degree of Engineer at the Electronics and Computer Engineering Department of the Technical University of Crete. In the current thesis we deal with the problem of automatic transcription of broadcast news in Greek, which is part of the Large Vocabulary Continuous Speech Recognition (LVCSR) field.

In recent years there has been increasing interest in developing Automatic Speech Recognition (ASR) systems for speech found in real sources such as broadcast news or telephone conversations. Only when the statistical pattern recognition and the Hidden Markov Model (HMM) approach started to get into speech recognition aspects, major progress allowed shifting the focus of research into these less restrictive domains.

To support the research and development associated with this task, it was necessary a representative audio as well as text corpus to be collected. The former is needed for the training of the acoustic models of the phonemes which were our basic acoustic units, while the latter is used for creating the language model of the system. To implement and evaluate our recogniser we used the HTK toolkit [13] which is primarily designed for building HMM-based speech recognisers.

In the current research, one can have a detailed report of the methods we followed to collect the corpora needed and build and train the language and acoustic models. Taking a look at the presented evaluation tests, we will come to the conclusion that speech recognition methods are of great promise, however, further improvements are required in order for robust systems with high successful recognition rates to be constructed.

The motivation for working on this thesis was the fact that making progress on such a challenging task, many difficulties in ASR must have been understood and overcome up to a point. The purpose was to become familiar with the principles of speech recognition and experiment with real tools building and evaluating a recogniser from scratch.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

Speech recognition aspects concern researchers from all over the world for more than half a century. Systems implemented in 1950s could recognise digits or only a few vowels from a specific user. Later on, LVCSR speaker independent systems started to become the core activity of the speech recognition community and in the mid-1980s statistical modeling methods take the speech recognition area to higher levels replacing the deterministic, template-based approaches. Over the last few years, speech recognition technology is being increasingly used within telephone networks and intelligent systems to provide automatisation with various applications including voice dialing (e.g., "*Call home*"), call routing (e.g., "*I would like to make a call*") and other command recognition, simple data entry (e.g., entering a credit card number), preparation of structured documents (e.g., a radiology report) and content-based spoken audio search (e.g. "*find a broadcast where particular words were spoken*").

Broadcast news, telephone conversations and other sources of "found" speech under real conditions are of great scientific interest for building speech recognition applications that are not restricted to a small grammar but they are based on large vocabulary.

Automatic transcription of broadcast news poses a number of challenges for LVCSR systems. The data in broadcast is characterised by a variety of speaking styles, people, environmental conditions, not to mention the channel and microphone influences. In particular, there are utterances recorded under environmental noise, which is very common in reportages from the street with a lot of people around or under bad weather conditions, or studio noise such as cough or any paper riffling, as well as background music (e.g. at the beginning of the broadcast, or before the sports and finance report). The speech can be planned and read fluently or spontaneously, which introduces hesitations and different duration among the same words. To make matters worse, there are cases where speakers are non native which gives diversity on the articulation of the words. Hardware quality, like telephones and microphones used during the reportages and broadcast, plays also a significant role in the level of noise and quality of speech that comes to the receiver.

As one can easily understand, we have to create a system that is capable to overcome these obstacles and be able to recognise low quality speech. A detailed description of such a system and the principles based on which it was developed are presented here. To have representative evaluation of the recogniser, we used a number of different test sets which cover a large variety of acoustic signals. After we created the baseline system, we studied an approach of language model adaptation the basic idea of which, is to feed the training corpus with speech that is recorded during the same period as the evaluation data.

To give an idea of how we worked on this project, first we downloaded text from online newspaper, which constitutes the data for the language model training. `Perl` scripts were used where it was necessary for any text to be edited and come to the appropriate format. At the same time, we recorded television broadcast news which later on we manually transcribed in order to create the training data of the acoustic models. Then we were ready to proceed with the language and acoustic model[1] training. All the acoustic training tools as well as the Viterbi decoding tool we used to evaluate our models, were available at the HTK toolkit and for the language modeling we used the SRILM toolkit.

[1] Language model is the a-priori probability for words to be spoken, while acoustic model is the likelihood to observe this data given the words that were spoken indeed. See section 1.1 for details

OUTLINE

The development and performance of our system as well as the theoretical background are presented in this thesis which is organised throughout the next pages as follows:

CHAPTER 1 *Background - General Aspects*. In the first chapter we present a short review of the fundamentals of speech recognition, the Hidden Markov Model approach and the main problems this method is challenged to solve.

CHAPTER 2 *Data Preparations*. Here we describe the data collection and preparation for the text and audio corpus used for training the system.

CHAPTER 3 *Acoustic Models*. In this chapter we analyse all the stages of the phoneme-based acoustic models training using the HTK toolkit.

CHAPTER 4 *Language Models*. Here we give a description of the construction of the bigram language model as well as an adaptation method we used to build a dynamic language model.

CHAPTER 5 *Experiments - Evaluation*. All the evaluation tests made on our system are presented and discussed in detail in this chapter.

CHAPTER 6 *Conclusion - Future Work*. Finally, there is a short review of the system developed for this thesis, and some ways to improve its performance are proposed.

Many of the parts of this thesis were studied and developed in cooperation with O. Tsergoulas, the diploma thesis of whom [12] was quite similar and a lot of the materials presented here were useful to his work as well.

1

BACKGROUND - GENERAL ASPECTS

1.1 SPEECH RECOGNITION USING HIDDEN MARKOV MODELS

Until recently, LVCSR systems were not designed for multiple speakers and words should be spoken separately, with a short pause between them. Modern systems use statistical methods that have proved to be much more efficient. The most widely used one is the Hidden Markov Model (HMM) approach. The structure and the training algorithms developed for HMMs lead to high performances even for speakers that have not appeared in the training data.

In this chapter we present the basic rules of HMMs, based on which the project of the thesis was designed. To have a more in depth study of the method take a look at Rabiner and Juang's book [1].

1.1.1 *The main idea*

Speaking for ASR applications, a word network is constructed which is the grammar consist of all the acceptable sequence of words. Each of these sequences corresponds to a number of HMMs. When new data are about to get recognised, the system estimates the probability this data to have been observed from each of these HMMs. The output of the system is the sentence with the highest probability.

In statistical terms, when we want to recognise an utterance, we seek the word sequence $\overline{W}$, having observed the sequence $\overline{X}$. In other words, we look for the sequence $\overline{W}$ that maximizes the probability $P(\overline{W}|\overline{X})$.

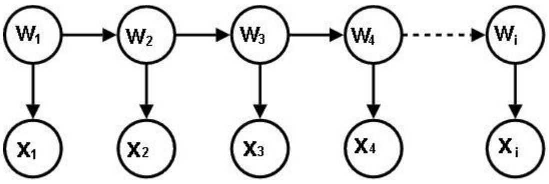


Figure 1. A first order Hidden Markov Model

Using the *Bayes' formula* to calculate this value we have:

$$P(\overline{W}|\overline{X}) = \frac{P(\overline{W})P(\overline{X}|\overline{W})}{P(\overline{X})} \tag{1.1}$$

Bayes' formula shows that the probability for $\overline{W}$ to have been spoken having $\overline{X}$ as an evidence depends on both how probable is the exact $\overline{W}$ to appear and the likelihood to observe data $\overline{X}$ caused from $\overline{W}$. The first term of the product, $P(\overline{W})$, is called *the language model* of the system and it is a probability distribution over strings that reflects how often this sequence of words occurs, while the second term, $P(\overline{X}|\overline{W})$, is called *the acoustic model* and it is the one that tells us how likely is to observe each evidence from the given sequence of words[1]. In chapters 3 and 4 we will see in detail how we can find these values and train our system.

1.1.2 *The three problems for HMMs*

Most of the cases that the HMM method is used in real-world applications involve one or more of the following three problems:

1. What is the probability of a sequence $\overline{X}$ to be observed?
2. Which is the most probable sequence of states $\overline{W}$ that caused the observation sequence $\overline{X}$?

[1] The value $P(\overline{X})$ in eq. 1.1 does not determine the optimal $\overline{W}$. It is just a scale factor which reassures that $P(\overline{W}|\overline{X})$ will be between 0 and 1 since it is a probability.

3. How can the model parameters be adjusted from the training data?

Let us have a brief description of these questions and the solution that the HMM approach propose.

Question 1: Calculation of $P(\overline{X}|\lambda)$

For a given model λ, we would like to find the probability of one sequence $\overline{X}$ to be observed by this model. The solution would be useful for knowing how possible is to measure values $\overline{X}$ from model λ or, if we have more than one models, which of these is more likely to have produced the observation sequence $\overline{X}$.

In order to find the solution, we have to check all the available paths through the HMM states and compute how probable is $\overline{X}$ to have been occurred from this path. If we sum up these probabilities, we will have the solution we seek. The most famous algorithms that solve this first problem are *the Forward procedure* and *the Backward procedure*. As it may seem, the former processes the observations from the first one $x_{t=1}$ to the last one $x_{t=T}$, while the latter acts conversely.

Both algorithms conclude to the same result, with the same computational complexity on the order of N^2T, where N is the number of states and T is the period or the number of observations.

Question 2: Which is the $\overline{W}$ *that maximizes* $P(\overline{W}|\overline{X}, \lambda)$*?*

This question is looking for the *optimal* state sequence $\overline{W}$ of the model λ, meaning the most probable one, that might have produced a given observation sequence $\overline{X}$. Looking closer to this problem, one can find a similarity with the previous one. In problem 1 we calculated the probability of each state sequence to produce $\overline{X}$, and then we summed them up. If we just skip the last sum, the answer we seek for problem 2 is the state sequence that produced the maximum of these probabilities.

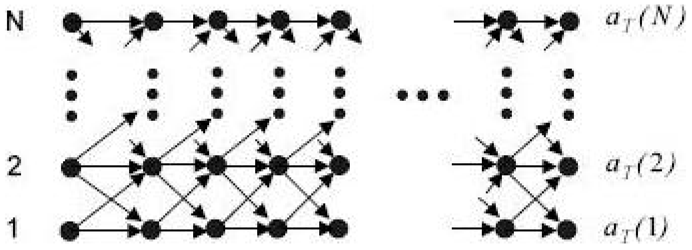


Figure 2. Trellis Diagram for the Viterbi decoding

However, when it comes to implementation aspects, the problem is not as easy as it may seem. If we used the Forward algorithm to solve problem 2, we would need to make some changes. First of all, we do not want the last sum, so we replace this step with the arg max function that could find us the $\overline{W}$ that has the maximum $P(\overline{W}|\overline{X})$. We need also to store this sequence of states for each path. This would lead to huge space waste. Imagine that a 3-state HMM with three observations has 27 possible paths and for each one there should be stored the likelihood of each path to have produce $\overline{X}$ as well as the state sequence itself.

Viterbi decoding has come to give a more proper solution to the problem. This algorithm has the same computational complexity as the Forward algorithm, N^2T, but it stores only N paths (instead of N^T). For each of the N states in time t, it keeps only the optimal path $\overline{W} = \{w_1, \dots, w_t\}$ that leads to that state. In time T, the path that has the highest probability holds the optimal state sequence. Finally, a *backtracking* step is needed to retrieve this sequence $\overline{W}$ from the array that it keeps all the N paths.

Question 3: Estimation of parameters $\alpha_{ij}, b_i(x_t), \pi_i$

The information we need in order to define an HMM is the $\alpha_{ij}, b_i(x_t)$ and π_i parameters. α_{ij} is the probability to have a transition from state i to state j, $b_i(x_t)$ is the probability to observe x_t when being in state i, and π_i is the probability the initial state to be the i^{th} one.

The problem this question poses is how we can estimate the values of the parameters $\lambda = \{A, B, \pi\}$ from the training data. *Forward-Backward algorithm* is the combination of the two, previously mentioned algorithms that helps us to estimate the probability to be at time t on a particular state. *Baum-Welch algorithm* is an iterative procedure that takes initial values for the parameters, uses Forward-Backward over the training data and a maximization step to compute new values for these parameters. The procedure continues until a convergence criterion is satisfied. Baum-Welch is actually the *Expectation - Maximization algorithm (EM)* for HMMs.

2

DATA PREPARATION

In the current chapter we will see how the data for both the language and the acoustic model were collected as well as got prepared in order for the system to get trained. The whole procedure was held keeping in mind that the better the data we feed the system, the more well trained it will become. The corpora development analysed below was held in cooperation with O. Tsergoulas since his thesis [12] was equally interested on it.

2.1 LANGUAGE MODEL DATA

In section 1.1 we tried to explain what the language model is from a statistical point of view. In a speech recognition system we could say that the language model is the one that gives prior information about the context of a particular word, or how probable is for that word to be found inside a specific context. Such information is very useful for example in cases that two or three words are almost equally probable to have been spoken, judging only from what the acoustic model "heard", and only by looking at the probability of these words to appear after the previously recognised words we can make a decision.

In this thesis we create a recogniser for television broadcast news. We need data in pure text, as much relevant with news as possible. The best source for such a text was Greek electronic newspaper that was found online. Although a newspaper article is written in a more formal style than the way reporters speak, it is the closest to what we need corpus we could create.

Table 1. Data collection for the Language Model.

Newspaper	*Text downloaded (MB)*	*News period*
Eleftherotypia	215	1997 - 1999
Ta Nea	170	2000 - 2006
To Vima	65	2000 - 2006

We downloaded text from three different newspapers, the most popular Greek ones: *"To Vima", "Ta Nea", "Eleftherotypia"*. News of political or social interest were selected as primary goals, but we also collected a small amount of sports and financial news. In the first two sources the news were written in the period 1^{st} of January 2000 to 31^{st} of March 2006. From "Ta Nea" we found text from all the week apart from Sundays for which we downloaded news from "To Vima". "Eleftherotypia" fed us with text written during the period 1997 - 1999 and had been collected for the "Logotypographia" project [9]. In total, we managed to create a corpus of 450 MB that now needs some processing in order to become valuable and useful.

We wrote few *Perl* scripts with the rules that will bring our text in the desirable format. These rules are:

- Make sure that only Greek letters are used, even for non-Greek words.

- One sentence per line. Speech recognisers usually take simple utterances as a unit for processing. After each sentence, a `'\n'` character should be added. This is not as easy as it sounds. Every dot is not necessarily a full stop, it can appear in words (e.g. acronyms) or in numbers.

- Expand all acronyms. Substitute each one (*etc., mr., . . .*) with what it stands for. If it is the first letter of a name, write the whole one or erase it if you do not know it.

- Write all numbers in full text, dates included. Of course, words like *millions* or *point* should be added if needed.

- Remove all punctuation marks. Only the stress mark is allowable.
- Make sure all text is written in lower case.

Data is now ready to train the language model. Each line of the LM corpus file has one sentence. How often a word appears in this text and close to what context, will determine the prior probabilities of the model. The training procedure will be discussed in chapter 4.

2.2 ACOUSTIC MODEL DATA

Acoustic model training contains information only about how the words sound in different circumstances, spoken by different people. Therefore, we need to feed the training procedure with labeled audio text.

2.2.1 *News Recording*

We chose the main broadcast news of the day (at 8.00 or 9.00 pm) from *"NET"*, *"ET1"* and *"SKAI"* channels. These were the broadcasts with the less multi-speaker parts because no more than two people appear at the same conversation. Mainly, there is planned speech from the studio and reportages, with the presence of background noise, from outside the studio.

We collected in total *20 hours* of audio data, excluding advertisements. For this purpose we used the *CRYPTO MPEG PC TV RADIO* card of the laboratory. We recorded 40 hours of broadcast news (both audio and video) but we used only the 20 hours. The audio properties are shown in Table 2.

Now we have to proceed with a correct labeling and then we will split all the data to training and evaluation sets.

Table 2. Audio settings.

Attribute	*Value*
Audio format	PCM
Sample rate	16 kHz
Sample size	16 bit
Bit rate	256 kbps
channels	1 mono

2.2.2 *Manual Transcription - Labeling*

There are many different conditions under which speech utterances have been recorded. There can be clear speech or speech under background noise or music. Speaker may read, which will cause a more fluent speech, or talk while thinking which can cause spontaneous talking or even hesitations and mis-articulations. Although we chose channels that their broadcast have no multi-speaker conversations, still we cannot avoid few utterances that more than one person is speaking.

The *Transcriber tool* were used for segmentation, labeling and transcription of the audio files. The rules that we put for the LM corpus development (see section 2.1) are followed also here. Moreover, for every sentence, the name and the gender of the speaker are noted, also if he is Greek or not and the way he speaks (either planned of spontaneously)[1]. As far as environmental conditions are concerned, we have to define if there is background noise (BGN), or music (BGM), or the speech is clean, if there are multiple speakers or if there is no speech at all. If the speech is from a telephone conversation, it should be noted as well. These characteristics are summed up on Table 3. Finally, utterances should be segmented so that all the turns from sentence to sentence and sections from one "environment" to the other are marked.

There are also few rules that we followed at the transcription part.

[1] If the person is not known we give the name *unk_s_n_num*, where *unk* is for unknown user, *s* denotes speaker's sex (either *m* or *f*), *n* is his nationality (Greek or non-Greek) and *num* is the number we give to distinguish him from the others.

Table 3. Conditions of the audio utterances.

Condition	*Possible values*
speaker's name	*name* or *unk_s_n_num*
speaker's gender	*m* or *f*
speaker's nationality	*g* or *n*
speech	*planned* or *spontaneous*
environment	*report*, *multispeakers*, *no_transcriptions*, *BGN* or *BGM*
channel	*telephone* or *studio mic*

Table 4. Sound events during the speech.

Event	*Symbol*
breath	[BREATH]
instant noise	[NOISE]
hesitation	@ϵ@
mis-articulation	*correct articulation*
non-finished words	[FRAGMENT]
bad reading	[TAG_BAD_READING]

During the speech, even if this is clear, recorded from the studio, there can be some events that disturb the homogeneity. Table 4 enumerates these events and the way we mark them among the transcriptions. Denoting these events of disturbance is as important as the rest of the transcriptions, because, as we will examine later (chapter 3), a model for each of these events will be also trained, in order for the system to recognise them and ignore them.

2.2.3 *Creating sets of data*

After we have finished with the transcriptions, we use Transcriber to split the `wav` files to separate ones, one for each utterance. Then we create a database with all the information provided from the transcriber

Table 5. The database after trs file is parsed.

wav directory	name	sex	native or not
/speech/NET_001.wav	Xoukli	female	native
/speech/NET_124.wav	rep_m_g_01	male	native
/speech/NET_138.wav	Papoulias	male	native
/speech/NET_280.wav	Xoukli	female	native

Table 6. The rest of the database.

condition	planned	studio	transcription
BGN	spontaneous	studio	Καλησπέρα σας, ένταση...
BGM	planned	telephone	Όπως σας είπα, εχτές...
clear	planned	studio	Κατατέθηκε στη βουλή...
clear	spontaneous	studio	Νομίζω όμως γιάννη ότι...

tool, that is, the wav file's directory, the conditions we have noted and the transcription itself. This database is the pool of our data from where we created all the training and test sets separating them with conditions criteria. The output of the Transcriber Tool is an "xml-like" file having a lot of useless information, flags, time limits, comments. From this file we need to keep only the useful material with all the conditions mentioned above and of course the transcription of the speech itself. Tables 5 and 6 can give an idea of the database format.

To create files with sets of data we will use this database. We will create one set-file for each condition, that is, one set for planned speech, one for spontaneous, one for speech under background noise (BGN) etc. Again, few Perl scripts will easily do the job for us. In these set files we will store only the wav file's directory and the transcription, since the rest information was useful only to make this separation. Table 7 shows which are the final categories and how many sentences each one has. Note here that around 10% of the sentences were not transcribed correctly or they did not have useful information (e.g. only noise), so they did not match to any of the categories.

Table 7. We divided all utterances into the following categories.

Category	*Number of utterances*	*Total time*
studio planned clear	1306	3h 51m
studio spontaneous clear	470	1h 23m
studio BGN	3366	9h 56m
BGM	417	1h 14m
telephone	712	2h 6m
telephone BGN	345	1h 1m
multispeaker	82	14m
non Greek speech	22	4m
non native speaker	52	9m

Training and test sets

It is important to be careful not to mix the utterances of the training and test sets. If one sentence appears in the training set, it should be excluded from the evaluation set, otherwise we would cheat and the results would seem successful but not corresponding to the truth.

We will keep the proportion of 80% of data to be used to train the acoustic models and 20% to evaluate the recogniser[2], so for each category we make two files.

Later we will need to build acoustic models based on three different kinds of data, so we prepare these sets now. For each of the sets numerated below we will make one set for training and one for testing our models.

- *Clear studio* utterances will be the training set for the first model.
- *Clear and BGN studio* utterances for the second one. We just concatenate the two files (clear studio sentences with BGN studio ones) that we have already made.
- *Every kind of data we have*, apart from multi-speaker parts and sentences from non-native speakers[3].

[2] In every 5 sentences we put the first 4 in the training set and the 5^{th} in the test set.

[3] from which we have anyway negligible amount

Up to this point, we have finished with all the work needed to prepare the data for learning and evaluation purposes. The sets we will use to build and test our models are ready and at the following chapters we will focus on the training procedure.

3

ACOUSTIC MODELS

In this chapter we discuss the learning procedure of the acoustic models of our system. We will use the training sets with the recorded utterances (ch. 2) as well as their transcriptions.

3.1 FRONT-END ANALYSIS

Before proceeding with the training, we have to convert the audio waveforms into a sequence of parameter vectors. We assume here that the speech signal is *locally stationary*, that is, during a few msecs its characteristics do not change. Then we can divide it into frames from which we extract parameter blocks. The period between each parameter vector is typically 10 msecs. The length of each segment of the speech signal that determines a parameter vector, often called as *window*, is 25 msecs. As it may seem, successive windows overlap.

There are some useful signal processing techniques one could use to have better results in the characteristic extraction. First, DC mean is removed from each window of the source signal individually. In addition, *pre-emphasizing* could be performed, in order for the attenuation caused by the lips to be balanced. This is done by applying the first order difference equation $s'_n = s_n - Ks_{n-1}$ to the samples $\{s_n, n = 1, N\}$ in each window, where K is the pre-emphasis coefficient. Last, applying *Hamming* windowing to the samples could cause discontinuities at the

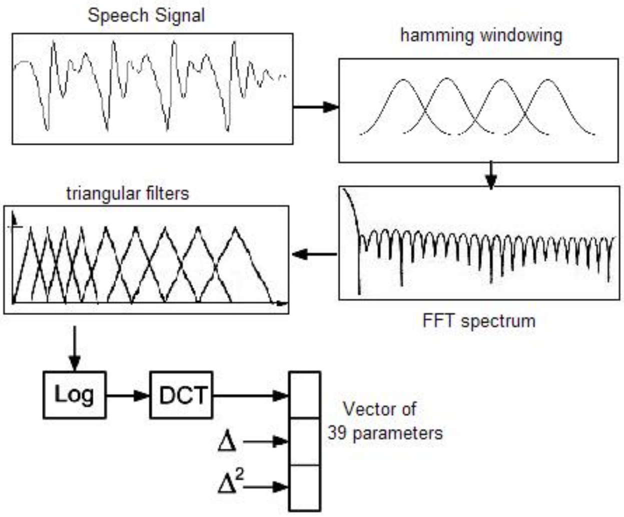


Figure 3. Mel frequency cepstral coefficients

window edges to be attenuated. This transformation is given by

$$s'_n = \left\{0.54 - 0.46\cos\left(\frac{2\pi(n-1)}{N-1}\right)\right\} s_n \tag{3.1}$$

According to psychological research, it has been proved that the human ear resolves frequencies non-linearly across the audio spectrum. Therefore, in order for the parameters to be extracted, the speech signal processing is based on *short time Fourier analysis* which is computed with a series of filterbanks. As can be seen in figure 4, the filters used are triangular and they are equally spaced along the mel-scale which is defined by

$$\mathrm{Mel}(f) = 2595\log_{10}(1 + \frac{f}{700}) \tag{3.2}$$

These filters are correlated with the magnitude of the Fourier transform of each speech window. The filterbank amplitudes m_j are then

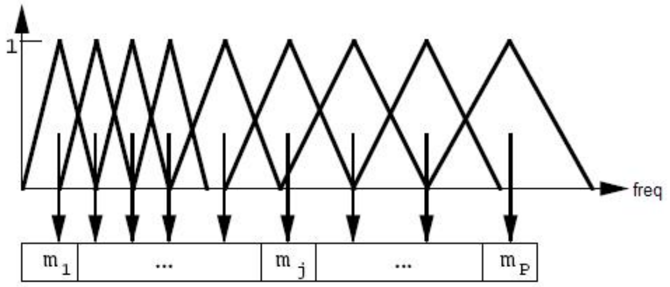


Figure 4. Mel-Scale Filter Bank

Table 8. Signal Processing settings.

Variable	Description
SOURCEFORMAT = WAV	format of wav files
TARGETKIND = MFCC_E_D_A_Z	create MFCC: C0 + Deltas + acceleration + Mean Normalization
TARGETRATE = 100000	frame period 10ms (HTK uses 100ns unit)
WINDOWSIZE = 250000	windows size 25ms
ZMEANSOURCE = TRUE	zero mean source waveform (removes DC)
PREEMFCOEF = 0.97	pre-emfasis coefficient
USEHAMMING = TRUE	use Hamming window
NUMCHANS = 26	number of filterbank channels
CEPLIFTER = 22	cepstral liftering coefficient
NUMCEPS = 12	num of cepstral coefficients
SAVECOMPRESSED = TRUE	compressed output
ENORMALIZE = TRUE	energy normalization

used to compute the *Mel-Frequency Cepstral Coefficients* (MFCC) from the Discrete Cosine Transform

$$c_i = \sqrt{\frac{2}{N}} \sum_{j=1}^{N} m_j \cos\left(\frac{\pi i}{N}(j - 0.5)\right) \tag{3.3}$$

We set up the signal processing parameters according to the Table 8. The final parameter vector that we extract consists of 12 cepstral coefficients, plus the log of the signal energy which is normalised, plus the delta and acceleration coefficients, that is, 39 MFCCs in total. Hence, for every sec of speech, we store 100 vectors of 39 parameters.

3.2 training

The speech waveforms have been converted to MFCCs and they are ready to be processed. In order to start building models, we have to decide which unit to choose for training. One approach could be to create an HMM for each word of the Greek language. For LVCSR systems, like the one we build, this method has two important drawbacks: First, training data would never be enough, so that they contain lots of appearances for each Greek word. Second, if one word does not appear in the training corpus, then it will be eliminated from our system.

The approach of "phoneme-based" modeling overcomes these problems. The phonemes used in most of the european languages are no more than 50, contrarily to words that are some hundreds of thousands. Therefore, the training procedure would be more effective, since certainly there will be occurrences of all the phonemes in a reasonably large training corpus. In addition, thanks to the fact that the system recognises words phoneme by phoneme, new words that were not in the training data can appear and recognised successfully[1].

[1] as long as they appear in the language model and the vocabulary of 60000 words (see ch. 4)

Table 9. The 28 phonemes that the Greek language consists of

A, v, i, s, o, g, E, l, G, m, J, r, n, t, u, z, x, D, k, T, ly, f, p, d, C, b, N, c

Table 10. Our system was also trained to recognise the events listed below

Extra Phoneme	*Description*
sil	silence usually appeared at the beginning and the end of an utterance
hes	speaker's hesitation
bre	speaker's inhalation/exhalation
fra	incomplete word
noi	instant noise
tbr	bad reading
sp	speaker's short pause

The model accuracy can be increased if instead of single phoneme training, we also build *biphones* and *triphones*. Each phoneme is differently pronounced in each context and it is depended on its previous and next phoneme. Although there are $28^3 = 21,952$ triphones, in our broadcast news corpus around 3600 triphones appeared due to language-phonetical constraints[2].

The 35 phonemes for which we built models, are listed in Tables 9 and 10. The former Table is taken from the "Logotypographia" project [9]. On the latter, we have put some extra phonemes so that the system can recognise and ignore these events[3].

The HMM we create for every phoneme looks like the one in figure 5. Every phoneme is divided to three states, the beginning state, where the speaker attempts to pronounce it, the middle state which is the phoneme itself and the exit state where the speech fades as finishing with that phoneme. HTK adds two more states that are non-emitting.

[2] Another reason could be the fact that a good percentage of broadcast news has constant vocabulary with standard words and phrases

[3] In Table 4 we analyse these events that appear in speech.

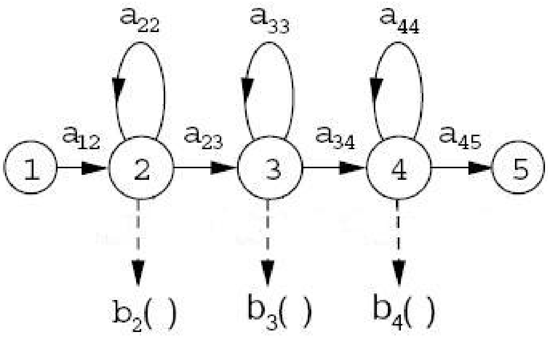


Figure 5. The structure of a 3-state left-right HMM used for a phoneme. Note that HTK adds 2 non-emitting states, the entry and the exit state.

Table 11. Transition matrix of a phoneme HMM

α_{ij}	1	2	3	4	5
1	$0.00e+00$	$1.00e+00$	$0.00e+00$	$0.00e+00$	$0.00e+00$
2	$0.00e+00$	$8.69e-01$	$1.31e-01$	$0.00e+00$	$0.00e+00$
3	$0.00e+00$	$0.00e+00$	$8.93e-01$	$1.07e-01$	$0.00e+00$
4	$0.00e+00$	$0.00e+00$	$0.00e+00$	$8.51e-01$	$1.49e-01$
5	$0.00e+00$	$0.00e+00$	$0.00e+00$	$0.00e+00$	$0.00e+00$

In the transition matrix (Table 11) we see that after each state we can either stay at the same or transit to the right one. This is called *left-right HMM*. All rows sum to 1 apart from the last one that has zeros everywhere since no transitions are allowed out of the final state.

As described in section 1.1, the *Baum-Welch* algorithm is used to estimate the parameters of the model $\lambda = \{A, B, \pi\}$. Too few iterations of the algorithm would result to estimations far from the real probability distributions. On the other hand, too many of them could cause *overfiting*. We made three iterations in each step of the training procedure which proved to have good results.

In our approach we built continuous density models in which each observation probability distribution is represented by a mixture Gaussian density. The probability $b_j(\bar{o}_t)$ of generating observation $\bar{o}_t$ is given by

$$b_j(\overline{o}_t) = \sum_{m=1}^{M} c_{jm}\mathcal{N}(\overline{\mu}_{jm}, \overline{\Sigma}_{jm}) \tag{3.4}$$

where M is the number of mixture components, c_{jm} is the weight of the m^{th} component and $\mathcal{N}$ is a multivariate Gaussian defined as

$$\mathcal{N}(\overline{\mu}_{jm}, \overline{\Sigma}_{jm}) = \frac{1}{\sqrt{(2\pi)^n|\overline{\Sigma}_{jm}|}} e^{-\frac{1}{2}(\overline{o}_t-\overline{\mu}_{jm})'\overline{\Sigma}_{jm}^{-1}(\overline{o}_t-\overline{\mu}_{jm})} \tag{3.5}$$

where n is the dimensionality (in our case $n = 39$, since we have used 39 characteristics-MFCCs from the speech waveform). We will see in chapter 5 that increasing the number of Gaussian mixtures up to $12 - 14$ will lead to better performance.

4

LANGUAGE MODELS

4.1 THEORY OVERVIEW

4.1.1 N-*gram Language Models*

As we saw earlier, in section 1.1, the language model (LM) is a probability distribution over the states, giving us information of how frequently a state sequence can appear. An LM describing spoken language has a probability for every word, e.g. $P(hi) = 0.01$, since once every one hundred words the word *hi* may appear.

The most widely used language models are *n-gram* models, were value *n* is called the order of the LM. For $n = 1$, the LM is called *unigram*, for $n = 2$ *bigram*, for $n = 3$ *trigram* and so on. Although *n* can take any value, in practice, the largest, most popular order is 3.

Let us explain the n-gram method by considering the case of the bigram model, since such a model were used in our project. Suppose sentence *s*, composed from the words $\{w_1, \ldots, w_k\}$, so we have:

$$p(s) = p(w_1)p(w_2|w_1)p(w_3|w_1, w_2)\ldots p(w_k|w_1, \ldots w_{k-1}) \quad (4.1)$$

Since our model is bigram, we make the assumption that every word only depends on the immediately preceding word, often called as *history*. So (4.1) can be written as:

$$p(s) = p(w_1)p(w_2|w_1)p(w_3|w_2)\ldots p(w_k|w_{k-1}) \approx \prod_{i=1}^{k} p(w_i|w_{i-1}) \tag{4.2}$$

The $\approx$ symbol is because of the $i = 1$ case, since we have not defined w_0. We could introduce w_0 as the *beginning of the sentence*, say *BOS*. Therefore $p(w_1|w_0)$ is the probability to have w_1 as the first word of the sentence. Similarly, there is need to introduce the *end of the sentence*, say *EOS* in order for all the sentences to sum to 1. To understand this, assume the following example: Take $s_1 = \{$Hello there$\}$ and $s_2 = \{$Hello there my dear friend$\}$ as the only sentences of a given text. As we see, s_1 is included in s_2. The probability of $p(s)$ is estimated by counting how many times s appeared in the text, normalised over all the sentences. According to these, from (4.2) we have $p(s_1) = 2/2 = 1$ and $p(s_2) = 1/2 = 0.5$. This leads us to $\sum_s p(s) = 1.5$. If we had introduced *EOS*, this problem would not have occurred.

Back to our main problem, to estimate $p(w_i|w_{i-1})$ we need to count how many times w_i appeared in a given text after w_{i-1}, let us say $c(w_{i-1}, w_i)$, and divide it with the number of times that w_{i-1} occurred, say $c(w_{i-1})$. We have:

$$p(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})} \tag{4.3}$$

The available text based on which our model is trained, is called *training set*. This method of estimating probability $p(w_i|w_{i-1})$ is called *maximum likelihood* (ML) estimate.

4.1.2 *Smoothing*

Consider a case where a sentence have not appeared in the training data. This could happen when the available text is not big enough or it is not relevant with the test set. Then, according to the ML estimation we have $p(s) = 0$ which would lead to no transcription whatever the acoustic signal is. To avoid this problem we use the technique of *smoothing*. As it may seem from the term used, this technique describes ways of adjusting the ML estimate of probabilities so that more accurate numbers occur. In practice, not only do smoothing techniques prevent zero probabilities, but they also improve the accuracy of the model.

One of the most famous and efficient methods is *Kneser and Ney's* (1995) technique [16]. According to this method, it is assumed that the form of the bigram model is given by

$$p_{KN}(w_i|w_{i-1}) = \frac{\max\{c(w_{i-1}, w_i) - D, 0\}}{c(w_{i-1})} + \frac{D}{c(w_{i-1})} N_{1+}(w_{i-1}, *) p_{KN}(w_i) \tag{4.4}$$

where $0 < D \leqslant 1$ is a fixed discount and $N_{1+}(w_{i-n+1}^{i-1}, *)$ denotes the number of unique words that follow history w_{i-n+1}^{i-1}. Symbol N_{1+} is there to remind that we seek for the number of words that have one or more counts. Note that w_{i-1} represents the history of the word w_i, therefore, for higher orders n of the model, w_{i-1} in (4.4) becomes $\{w_{i-n+1}^{w_{i-1}}\}$ and the last term $p_{KN}(w_i)$ becomes $p_{KN}(w_i|w_{i-n+2}^{w_{i-1}})$.

For the language model we built in our project, we used the *modified* Kneser-Ney technique, proposed by Chen and Goodman (1998) [17]. It is proved [20] that this method leads to both smaller perplexities and Out-of-Vocabulary (OOV) rates. In their approach, instead of having one single discount value D, three different D_1, D_2, D_{3+} are used for either one, two or three or more n-gram counts, respectively. Applying this change to (4.4)[1], we get:

$$p_{mKN}(w_i|w_{i-1}) = \frac{c(w_i|w_{i-1}) - D(c(w_i|w_{i-1}))}{c(w_{i-1})} + \gamma(w_{i-1})p_{mKN}(w_i) \tag{4.5}$$

where $D(c)$ is

$$D(c) = \begin{cases} 0 & \text{if } c = 0 \\ D_1 & \text{if } c = 1 \\ D_2 & \text{if } c = 2 \\ D_{3+} & \text{if } c \geqslant 3 \end{cases} \tag{4.6}$$

In order for the distribution p_{mKN} to sum to 1, γ is given by

$$\gamma(w_{i-1}) = \frac{D_1 N_1(w_{i-1},*) + D_2 N_2(w_{i-1},*) + D_{3+} N_{3+}(w_{i-1},*)}{c(w_{i-1})} \tag{4.7}$$

They also suggested the following optimal values for D_1, D_2 and D_{3+}:

$$\begin{aligned} Y &= \tfrac{n_1}{n_1+2n_2} \\ D_1 &= 1 - 2Y\tfrac{n_2}{n_1} \\ D_2 &= 2 - 3Y\tfrac{n_3}{n_2} \\ D_{3+} &= 3 - 4Y\tfrac{n_4}{n_3} \end{aligned} \tag{4.8}$$

where n_i are the total number of *n*-grams with exactly i counts.

The smoothing schemes presented above as well as the *n*-gram mod-

[1] always speaking for bigram models. For higher order *n* we make the changes mentioned above.

eling were implemented with the tools provided by SRI International. In broadcast news and other large vocabulary applications, trigram or even 4-gram are the most famous and most widely used LMs. However, for our system we built a bigram LM because only this choice was provided by the HTK Toolkit (version 3.3) [13] that we used. Our vocabulary consists of the 60000 most frequent words found in the training data[2].

4.1.3 *Perplexity*

For the evaluation of the language model we built, we will use measures from the field of information theory, *entropy* and *perplexity*. Let us assume that a speaker is a source of information, generating words $\{w_1, w_2, \dots, w_n\}$ from a vocabulary set V. *Entropy* is defined as:

$$H = -\lim_{n \to \infty} \frac{1}{n} \sum_{w_i \in V} p(w_1, w_2, \dots, w_n) \log_2 p(w_1, w_2, \dots, w_n) \quad (4.9)$$

where the sum is over all possible word sequences $\{w_1, w_2, \dots, w_n\}$. If the source is ergodic then for large vocabularies V, (4.9) becomes:

$$H = -\frac{1}{n} \log_2 p(w_1, w_2, \dots, w_n) \quad (4.10)$$

Perplexity is the measure that is in standard use for LM evaluation and is given by:

$$PP = 2^H = p(w_1, w_2, \dots, w_n)^{-\frac{1}{n}} \quad (4.11)$$

To understand this definition, think that if a language model has

[2] 60K is the typical size for LVCSR systems

perplexity X, it means that every given word can be followed by X words with equal probability. Therefore, the lower the perplexity, the closer we are to the true model, which has the lowest possible perplexity. Note that, if the vocabulary is smaller then perplexity also decreases (less words in vocabulary means less words to follow each word as well), which means that perplexities from different vocabularies should not be compared. We can only compare perplexities of different models all with respect to the same text and the same vocabulary. Finally, if the models to be compared are trained with the same data, perplexity correlates with speech recognition word error rate.

4.2 ADAPTATION

4.2.1 *Introduction*

In section 2.1 we created a language model corpus with 450 MB of Greek news found in newspapers that cover the period 1997-2006, as we saw in Table 1. Since we attempt to build a speech recogniser for broadcast news, this corpus comes from the same population as the test data to which we want to apply our model.

However, political and social schemes are changing fast and different persons are under public exposure day by day. Most probably the recogniser will be used to transcribe the news of one specific day, say Tuesday, the 27th of June, 2006. It is likely that some of the news will be the same as Monday's news and some of the names mentioned then, may be also mentioned on Tuesday. On the contrary, the names and facts that were at the currency in 1997 would give little information about the news after 9 years. Therefore, it makes sense to assume that if we had much information about the news of the previous week, or even the news of the following week, we could adapt this information to our language model, so that the prior probabilities of the *n*-grams are re-estimated according to this information.

We go back to the LM corpus creation and we follow the next steps:

1. Get the LM corpus of the 450 MB previously constructed with news from the period 1997-2006.
2. Download the news of the previous and the following weeks of the day on which you want to apply the recogniser.
3. Clear the data from useless information (such as flags of web pages etc.) and modify them according to the rules of section 2.1. The utterances just created compose the "focused" text (F).
4. Multiply these utterances as many times as needed so that the desirable percentage of general/focused data is achieved. For example, if we want to create an LM with 80-20% proportion of general and focused text respectively, then, given the previous LM of 450 MB, the adapted one should be 562 MB, so multiply the new text until it gets 112 MB long.
5. Finally, concatenate this text to the old language model corpus.

4.2.2 *Statistical analysis of the problem*

Let us study this technique from a statistical point of view. Assume the new corpus that we have just designed, the main difference of which, comparing to the baseline one, is that we have added a big amount of data which are focused on the day we want to transcribe. Equations (4.5-4.8) will give us the bigram probabilities for our LM. In order to examine these equations and how they change with the new corpus, we have to study five different cases separately:

1. Both w_i and history w_{i-1} are not in the F text.
2. Word w_i appears in F but history w_{i-1} does not.

3. Word w_i is not in *F* but history w_{i-1} is.

4. Both w_i and history w_{i-1} appear in the text *F* but not in a sequence.

5. Bigram $\{w_{i-1}, w_i\}$ appears in the text *F*.

In order to proceed with the analysis of these cases, let us see the behavior of (4.8) which is the same for all cases. n_i is the number of bigrams that appear i times in total. If a bigram of one of these categories, say n_1, does not appear in the *F* text, then its counts are the same as in the previous corpus so it stays in this category. However, if this bigram is found in text *F*, then it will be multiplied many times with the rest of the *F* text, so its counts will be increased a lot, the bigram will not belong to n_1 category any more and n_1 will decrease. Therefore, n_i either stay the same or decreases. However, this decrease, comparing to the order of n_i, is negligible, so we make the assumption that n_i and thus D_i are the same as in the previous corpus.

Word w_i and history w_{i-1} do not appear in the F *text*

It may happen that a bigram does not appear in the *F* text. In this case, $c(w_i|w_{i-1})$, $D(c(w_i|w_{i-1}))$, $c(w_{i-1})$ and $\gamma(w_{i-1})$ will not change. The only term that changes is the unigram $p_{mKN}(w_i)$ because we have the same number of occurrences of w_i in more text. If the old text is, let us say the 80% of the new one, we have:

$$p_{mKN}(w_i) = \frac{c(w_i)}{\text{\#words in new text}} = \frac{c(w_i)}{\frac{\text{\#words in old text}}{0.8}} = 0.8p_{old}(w_i) \tag{4.12}$$

where $p_{old}(w_i)$ is the probability of w_i in the old corpus. As it was obvious, the probability of a bigram that does not appear in the *F* text decreases.

Word w_i appears in F but history w_{i-1} does not

Again, $c(w_i|w_{i-1})$, $D(c(w_i|w_{i-1}))$, $c(w_{i-1})$ and $\gamma(w_{i-1})$ stay as they were in the training with the previous text and the only thing that $p_{mKN}(w_i|w_{i-1})$ depends on is $p_{mKN}(w_i)$. The occurrences of w_i are more in the larger text. This does not necessarily means that $p_{mKN}(w_i)$ increases. Actually, it depends on the frequency of that word in the *F* text. If it is higher than the frequency of it in the rest of the text, then its probability will increase, otherwise it will stay almost the same or even decrease if in the *F* text it appears only a few times, comparing to the rest of the text.

Word w_i is not in F but history w_{i-1} is

This case is more interesting than the previous ones. $c(w_i|w_{i-1})$, $D(c(w_i|w_{i-1}))$ do not change. $c(w_{i-1})$ will increase since history is multiplied with all the *F* text. $p_{mKN}(w_i)$ will decrease (as in the first case) because its occurrences are the same but we have larger text. $\gamma(w_{i-1})$ needs more discussion before we conclude. In (4.7), $N_1(w_{i-1},*)$ and $N_2(w_{i-1},*)$ will decrease for the same reason that n_i also decreases. In particular,

$$\hat{N}_1(w_{i-1},*) = N_1(w_{i-1},*) - N_1(w_{i-1},*,F) \tag{4.13}$$

where $\hat{N}_1$ is the N_1 in the new LM, and $N_1(w_{i-1},*,F)$ is the number of unique words that follow the history w_{i-1} in the *F* text, which we do not want to count since their occurrences will be by far much more than one, or two. On the other hand, $N_{3+}(w_{i-1},*)$ increases, because the cases that were excluded from the previous categories will go to this one.

$$\hat{N}_{3+}(w_{i-1},*) = N_{3+}(w_{i-1},*) + N_1(w_{i-1},*,F) + N_2(w_{i-1},*,F) \quad (4.14)$$

Since the orders of D_1, D_2, D_{3+} are the same, and $c(w_{i-1})$ increases, we can assume that $\gamma(w_{i-1})$ either stays as it is or decreases. Combining it with the rest changes in (4.5), we come to the conclusion that the probability of such a bigram decreases.

Both w_i and history w_{i-1} appear in the text F *but not in a sequence.*

In this case, $c(w_i|w_{i-1})$ and $D(c(w_i|w_{i-1}))$ stay the same. $c(w_{i-1})$ increases while $\gamma(w_{i-1})$ decreases. $p_{mKN}(w_i)$ depends again on how often w_i appears in *F* comparing to its frequency in the rest of the text. This means that the probability of the bigram decreases although this is not a rule and there are cases (if w_i appears much more times than w_{i-1}) that this probability increases.

Bigram $\{w_{i-1}, w_i\}$ appears in the text F

The probability of the bigram, if this appears in the *F* text, is again not so obvious. If the proportion between the occurrences of the bigram and those of the history only, is high, comparing to the rest of the text, then it is likely that the probability of the bigram will increase. However, it is required that the product $\gamma(w_{i-1})p_{mKN}(w_i)$ is also helpful, that is, the frequency of w_i in *F* should be more times higher than the occurrences of the history w_{i-1}.

From the previous analysis of the bigram probability distribution, one comes to the conclusion that applying this adaptation method, probabilities tend to approach those in the *F* text. We saw many times that what matters more is the comparison of the frequencies of words and bigrams between the whole text and the focused text. This is both reasonable and desirable because we believe that the words that have higher frequencies in this text, are the ones that are more likely to appear in the day the news of which we want to recognise.

In the current research, we investigated the presented method of dynamic language modeling and the experimental results are shown in chapter 5. We built a language model that is focused on a specific day, that is, we created an adapted LM with the information from the previous and the following weeks of that day. However, if one wants to establish the method to a system so that its LM automatically gets updated, only a script should be written putting the steps we followed inside a loop.

5 EXPERIMENTS - EVALUATION

In this chapter we have a discussion about the performance of the speech recognition system we created for the current thesis. We will determine the training and test sets and we will analyse the evaluation results. The recogniser run *Viterbi decoding* (ch. 1) to produce speech transcriptions. The metric we used to estimate the accuracy of the system is

$$\mathrm{Accuracy} = \frac{H - I}{N} \times 100\% \tag{5.1}$$

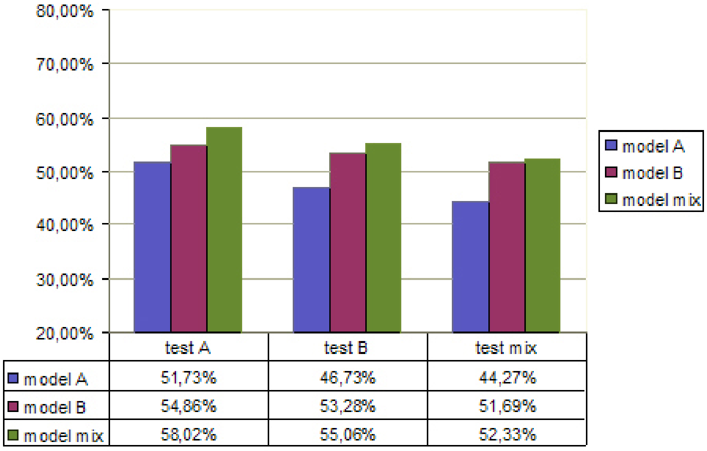
where H is the number of successfully recognised words, I is the number of insertions and N is the total number of labels in the defining transcription files.

5.1 THREE KINDS OF DATA SETS

In Table 7 (chapter 2) all the data we have collected are summed up. Here, we will create three different sets of data with which we will train three acoustic models: *model A*, consisting of *studio, planned and spontaneous, clear* utterances, *model B* consisting of the previous ones *plus background noise studio* ones and *model mix*, consisting of the ones in model B *plus background music, non-native* and *telephone* ones. We split these categories to training and test sets with 80-20% proportion. The sentences that occured for each set are shown in Table 12. Notice that the amount of speech under background noise is big because broadcast news contains a lot of outdoor reportages where people, vehicles etc.

Table 12. Utterances used in training and test sets

set name	*training set (utts.)*	*test set (utts.)*
A	1421	355
B	4114	1028
mix	5060	1263



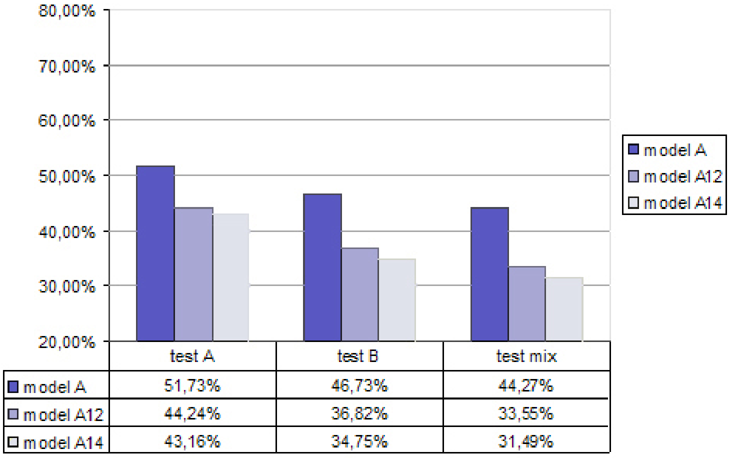
	test A	test B	test mix
model A	51,73%	46,73%	44,27%
model B	54,86%	53,28%	51,69%
model mix	58,02%	55,06%	52,33%

Figure 6. One-Gaussian models tested with the three test sets.

create a noisy environment[1].

The acoustic models are trained with one Gaussian component per model. Each of the models *A*, *B* and *mix* will work on all the three test sets. The results are shown in figure 6. As one would expect, *A* is the "easiest" test set since it contains only clear sentences. As the environment of the speech gets worse, recogniser's performance decreases. Notice also that the more utterances the model is trained with, the higher performance it gets, so, on the same test sets, model *mix* has better results than model *B*, while the latter has better results than model *A*.

[1] The term *studio* does not necessarily implies speech recorded inside the studio. It is used wherever there is use of microphone and not telephone.



	test A	test B	test mix
■ model A	51,73%	46,73%	44,27%
▨ model A12	44,24%	36,82%	33,55%
□ model A14	43,16%	34,75%	31,49%

Figure 7. Model A: 1, 12 and 14 Gaussians per model.

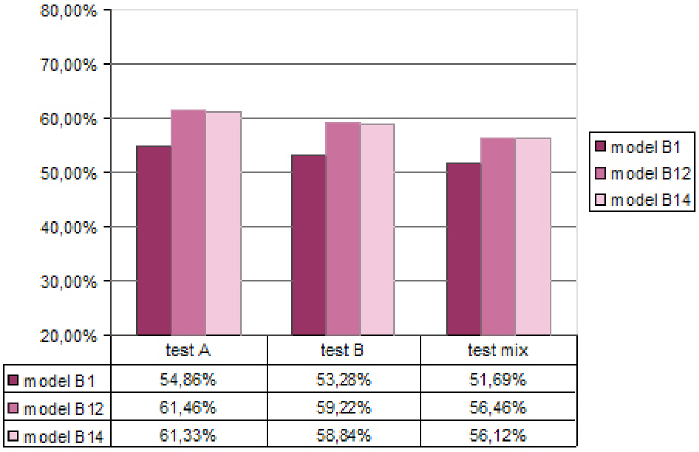
5.2 INCREASING THE NUMBER OF GAUSSIAN MIXTURES

Now, let us examine the behavior of our system as we increase the Gaussian components per model. As it is shown in figure 7, the performance of model *A* gets worse which means that data are modeled better with one Gaussian mixture. On the other hand, model *B* and model *C* (figs. 8 and 9) have much better results. The most important fact is that the performance of the model *mix* with 12 mixtures has an increase of almost 8% on the *mixed* test, which is the one we care more[2]. Moreover, we see that there is no need to have 14 mixtures per model, because not only does the complexity increase, but also the performance does not get higher, indeed there is a slight decrease.

The bottom line is that the best model is the one trained with mixed utterances[3] and consists of 12 Gaussian components. We will keep this model to continue with the rest of the experiments.

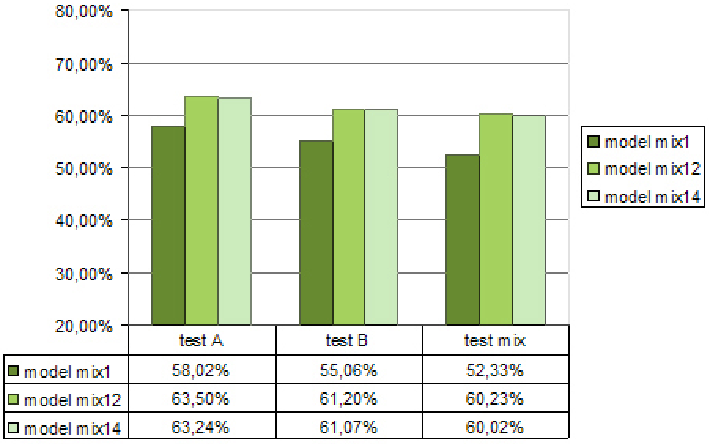
[2] since most of the utterances we will find will not be clear

[3] because mixed utterances are the largest corpus used for acoustic model training



	test A	test B	test mix
model B 1	54,86%	53,28%	51,69%
model B 12	61,46%	59,22%	56,46%
model B 14	61,33%	58,84%	56,12%

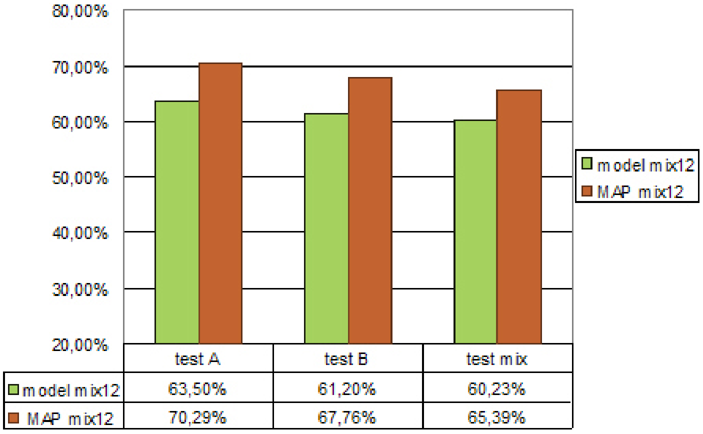
Figure 8. Model B: 1, 12 and 14 Gaussians per model.



	test A	test B	test mix
model mix1	58,02%	55,06%	52,33%
model mix12	63,50%	61,20%	60,23%
model mix14	63,24%	61,07%	60,02%

Figure 9. Model C: 1, 12 and 14 Gaussians per model.

5.3 acoustic models adaptation



	test A	test B	test mix
🟩 model mix12	63,50%	61,20%	60,23%
🟥 MAP mix12	70,29%	67,76%	65,39%

Figure 10. MAP adaptation on Model *Mix*.

As proposed in [12], if *Maximum A-Posteriori* (MAP) adaptation is applied on the acoustic models of our system, a performance increase of more than 5% can occur. This adaptation process is sometimes referred to as *Bayesian adaptation*. MAP adaptation involves the use of prior knowledge about the model parameter distribution. Hence, if we know what the parameters of the model are likely to be (before observing any adaptation data) using the prior knowledge, we might well be able to make good use of the limited adaptation data, to obtain a decent MAP estimate. The results are shown in figure 10.

5.4 language model adaptation

In section 4.2 we analysed a method of making the language model dynamic or updated with the news of the period we are interested in. Here we present the results of this method, applied in our system and evaluated in various test sets.

First, we have to decide the weights of the two training sets, the

Table 13. OOV rates and PPs for different weights of the LM training sets.

	70-30%	*80-20%*	*90-10%*	*95-5%*
OOV	3.07%	3.07%	3.06%	3.47%
PP	203	200	202	204

Table 14. 1st *experiment*: OOV rates and PPs before and after LM adaptation. LM represents the old LM, while LM2 represents the adapted one.

	test A		*test B*		*test mix*	
	LM	*LM2*	*LM*	*LM2*	*LM*	*LM2*
OOV	4.65%	3.34%	4.48%	3.24%	4.56%	3.06%
PP	221	208	229	217	236	202

450MB one, and the *focused* one. As Table 13 shows, there is no big difference on the OOV rates and perplexities as long as we keep the proportion under 90-10%. However, being only 5% of the training corpus, the *focused* text do not improve the system as much as possible. Therefore we keep the model created from 90-10% weights, since it has a slightly lower OOV-rate. Let us see what is the improvement that this model gives.

So far, we have reached an accuracy correct rate of 65.4%, which is a result on the mixed test set of 1263 utterances (Table 12). In figure 11 we see the performance of the recogniser before and after applying the LM adaptation method and on Table 14 we see the improvements on the Out-of-Vocabulary (OOV) rates and perplexities (PP). As we can see, a decrease of 1.5% on the OOV rate, combined with a slightly lower perplexity, can lead to an accuracy improvement of 2-3%. In this test, we compare the language model based on the three newspapers (Table 1), which is the one we use for all the tests up to here, with the language model built from the same corpus but with adapted data as well. Note that the purpose of this test is to show a straight comparison with the results presented in [12] and not to demonstrate our method's performance.

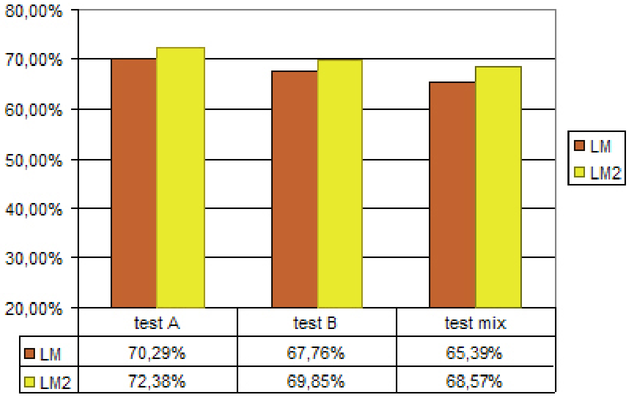


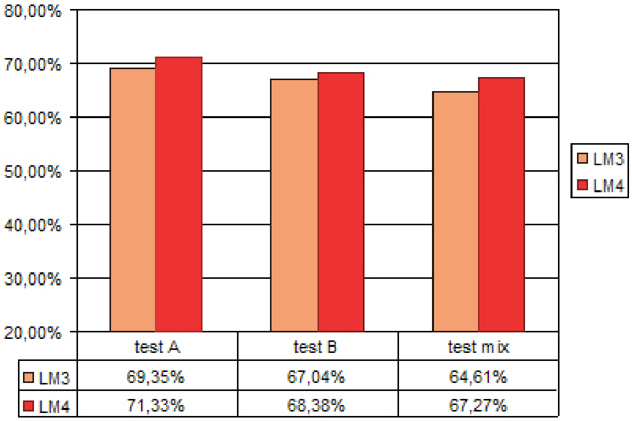
Figure 11. Performance after LM adaptation.

Remember that this method is based on the period we are interested in, for example, if we want to auto-transcribe speech from spring of 2006, we have to update an old LM with data from this period. Therefore, to see the improvement when we compare an old LM with an adapted LM, we have to make sure that the old LM does not already contain data of that period. In other words, the comparison just presented would be more representative if we used as a test set a speech made on 2010, in which case, the adapted LM would be updated with the current news, whereas the old LM would not.

Hence, in order to show the performance of the adaptation method we used, a different test must be held. We will use the corpus only from the "Eleftherotypia" newspaper (*enet*), because this news was written on 1997-1999. We create an adapted model by adding information from the news of 2006. On Table 15 we can see the corresponding perplexities and OOVs. The recognition results are shown in figure 12.

Table 15. 2^{nd} *experiment*: OOV rates and PPs before and after LM adaptation. LM3 is the old LM trained only by *enet*, and LM4 is the adapted one based also at *enet* text.

	test A		*test B*		*test mix*	
	LM3	*LM4*	*LM3*	*LM4*	*LM3*	*LM4*
OOV	5.52%	3.51%	5.51%	3.37%	5.63%	3.19%
PP	236	231	239	246	245	226



	test A	test B	test mix
🟧 LM3	69,35%	67,04%	64,61%
🟥 LM4	71,33%	68,38%	67,27%

Figure 12. LM adaptation results on the 2^{nd} test.

6

CONCLUSION - FUTURE WORK

In the current research, we deal with the problem of automatically transcribing speech from reek broadcast news. We collected data from online newspapers and TV broadcasts and we built an HMM-based speech recognition system from scratch. The principles and theoretical basis on which this project was elaborated were discussed. The baseline system as well as improved versions were presented and analysed in depth.

The final recogniser that has the best performance in terms of word accuracy, reaches results almost as high as 70%, on every kind of speech, either if this is recorded in the studio, or it is an outdoor reportage. This system is based on acoustic models trained for phonemes, biphones and triphones and every model has 12 Gaussian mixtures. Maximum a-posteriory adaptation is used on the acoustic models to increase accuracy. Our language model is bigram and it is trained with a corpus of 65M words and a 60K vocabulary. We applied an LM adaptation method according to which, we boost the unigram and bigram probabilities of the training speech that is recorded on the same period as the evaluation data. This method proved to improve the system, decreasing the WER in most cases about 7% and in some specific ones, more than 30%.

WHAT IS NEXT?

There is a number of actions one could take to improve the performance of the system. First of all, what matters a lot is the quality of the data we collect and we use to train our models. Many of the OOV words appeared in several tests, found to be mis-typed words. Hence, a more accurate manual transcription could lead to a well trained acoustic model set. The amount of this data plays also a significant role, and the 20 hours we used in this project are not enough. Since we attempt to create an LVCSR system, no less than 50 hours in total should constitute the training corpus.

As far as language modeling is concerned, a trigram or 4-gram approach should be tried instead of a bigram LM that we constructed. These approaches are the most common and widely used ones in systems like ours and they seem to have promising results when applied in other languages. Studies on the Greek language have also shown that a 100K vocabulary could prove more efficient than the 60K we used.

If these techniques are applied and the recognition accuracy level increases, we could pass to unsupervised training. Instead of manually transcribing more hours to feed the training corpus, we could use the Viterbi decoding for creating transcriptions automatically which could then be added to the training corpus[1].

Finally, segmentation methods can be studied and applied in order for the system to recognise the kind of speech input. Different models can be trained, adapted to several speech conditions (e.g. telephone speech) or to any speakers that appear more frequently than others. The system, judging from the conditions of the testing utterance, will decide from which model it should be decoded.

[1] as far as the system has a performance of about 80%

BIBLIOGRAPHY

[1] L. Rabiner and B.H Juang, "Fundamentals of Speech Recognition", *Prentice Hall Signal Processing Series*, ©1993 by AT&T. (Cited on page 11.)

[2] D. Jurafski, J. Martin, "Speech and Language Processing", *Prentice Hall*, ©2000.

[3] L. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proceedings of the IEEE, Volume 77, Issue 2, Feb 1989 Page(s):257 - 286.*

[4] P.C. Woodland, T. Hain, S.E. Johnson, T.R. Niesler, A. Tuerk, E.W.D. Whittaker and S.J. Young, "The 1997 HTK Broadcast News Transcription System", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[5] S. Wegmann, F. Scattone, I. Carp, L. Gillick, R. Roth, J. Yamron, "Dragon systems" 1997 Broadcast news transcription system", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[6] Ananth Sankar, "Experiments with a Gaussian Merging-Splitting Algorithm for HMM Training for Speech Recognition", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[7] D. Liu, L. Nguyen, S. Matsoukas, J. Davenport, F. Kubala, R. Schwartz, "Improvements in Spontaneous Speech Recognition", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[8] A. Sankar, F. Weng, Ze"ev Rivlin, A. Stolcke, and R. Rao Gadde, "Development of SRI"s 1997 Broadcast News Transcription System", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[9] V.Digalakis, D.Oikonomidis, D.Pratsolis, N. Tsourakis, C. Vosnidis, N. Chatzichrisafis and V. Diakoloukas "Large Vocabulary Continuous Speech Recognition in Greek: Corpus and an Automatic Dictation System", *EUROSPEECH 2003 - GENEVA*. (Cited on pages 17 and 28.)

[10] P. Beyerlein, X. Aubert, R. Haeb-Umbach, D. Klakow, M. Ullrich, A. Wendemuth and P. Wilcox, "Automatic transcription of english broadcast news", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[11] S. Chen, M. J. F. Gales, P. S. Gopalakrishnan, R. A. Gopinath, H. Printz, D. Kanevsky, P. Olsen and L. Polymenakos, "IBM'S LVCSR system for transcription of broadcast news used in the 1997 HUB4 english evaluation", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[12] O.Tsergoulas "Αυτόματη απομαγνητοφώνηση ακουστικών σημάτων ηχογραφημένα από τηλεοπτικές εκπομπές", *Diploma Thesis, accepted to the Technical University of Crete*, May 2007. (Cited on pages 10, 16, 46, and 47.)

[13] S.Young, G. Evermann, M.Gales, T.Hain, D.Kershaw, G.Moore, J.Odell, D.Ollason, D.Povey, V.Valchev, P.Woodland, "The HTK Book (for HTK Version 3.3)", *Cambridge University Engineering Department*, ©April 2005 (Cited on pages 3 and 35.)

[14] D. Wang, Shrikanth S. Narayanan, "A confidence-score based unsupervised MAP adaptation for speech recognitoin", © *2002 IEEE*

[15] Man-Wai Mak, R. Hsiao and B. Mak,. "A comparison of various adaptation methods for speaker verification with limited enrollment data", *ICASSP 2006, © IEEE*

[16] R.Kneser and H.Ney. 1995. "Improved backing-off for m-gram language modeling", *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, volume 1, pages 181-184* (Cited on page 33.)

[17] S.Chen and J.Goodman, "An empirical study of smoothing techniques for language modeling", *In Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, 1996. (Cited on page 33.)

[18] Yik-Cheung Tam and Tanja Schultz, "Correlated latent semantic model for unsupervised LM adaptation", *ICASSP 2007, © IEEE*

[19] Dong Yu, M. Mahajan, P. Mau, A. Acero, "Maximum entropy based generic filter for language model adaptation", *ICASSP 2005, © IEEE*

[20] D.Oikonomidis, "Language models for speech recognition", *Submitted in partial fulfillment of the requirements for the degree of Master of Science, Technical University of Crete Chania*, December 2002. (Cited on page 33.)

[21] D. Klakow, X. Aubert, P. Beyerlein, R. Haeb-Umbach, M. Ullrich, A. Wendemuth and P. Wilcox, "Language-model investigations related to broadcast news", *DARPA Broadcast News Transcription and Understanding Workshop 1998*

[22] S. Chen, D. Beeferman, R. Rosenfeld, "Evaluation metrics for language models", *DARPA Broadcast News Transcription and Understanding Workshop 1998*