The Design and Application of an Acoustic Front-End for Use in Speech Interfaces

Christoph Gerber

Department of Computing Science
Faculty of Science
University of Glasgow
Glasgow
United Kingdom

August 1996

This thesis is submitted for consideration for the degree of Master of Science

Acknowledgements

I would like to thank my supervisor Professor Keith van Rijsbergen, for his guidance, support and encouragement during my research. It has been a great experience working with Keith. I would also like to thank my second supervisor Dr. Bill Findlay for his support and his valuable comments. The Glasgow Information Retrieval group provided a pleasant environment from which I benefited in many ways. I would like to thank all members of the group.

Abstract

This thesis describes the design, implementation, and application of an acoustic front-end. Such front-ends constitute the core of automatic speech recognition systems. The front-end whose development is reported here has been designed for speaker-independent large vocabulary recognition. The emphasis of this thesis is more one of design than of application. This work exploits the current state-of-the-art in speech recognition research, for example, the use of Hidden Markov Models. It describes the steps taken to build a speaker-independent large vocabulary system from signal processing, through pattern matching, to language modelling. An acoustic front-end can be considered as a multi-stage process, each of which requires the specification of many parameters. Some parameters have fundamental consequences for the ultimate application of the front-end. Therefore, a major part of this thesis is concerned with their analysis and specification. Experiments were carried out to determine the characteristics of individual parameters, the results of which were then used to motivate particular parameter settings. The thesis concludes with some applications that point out, not only the power of the resulting acoustic front-end, but also its limitations.

Contents

Chapter 1

Overview

This thesis presents the design and application of an acoustic front-end. An acoustic front-end is the major component of speech recognition systems. The work has been carried out in the context of Information Retrieval with the long-term goal of building an unconstrained speech interface for text-based Information Retrieval systems. The task can be summarised as "encode the message given the spoken utterance" for every speaker for every utterance. This means in concrete terms that the input is speech in form of a physical signal and the output should be the corresponding text in form of the orthographic transcription. That would have been impossible just a couple of years ago. However, a technology switch in the end of eighties caused an enormous boom in speech recognition research. At present, in the mid nineties, the current state-of-the-art of speech recognition are systems which recognise fluently spoken English speech from any speaker with an word error rate below 8% based on a total vocabulary of 64,000 words. Speech recognition is still far away from perfect.

> *Quoting a speech recognition expert: "It isn't easy to wreck a nice beach!" Sorry that was a small error of the speech recogniser. Of course, it should mean: "It isn't easy to recognise speech!"*

However, recent research has demonstrated that speech recognition will very soon become a generally accepted standard technology. Speech recognition requires knowledge from signal processing, through statistical pattern recognition up to linguistics. Each area takes over a stage in encoding the message out of the signal. Up to now, there is no overall recipe for setting up these individual stages. Each stage still requires detailed understanding and experimentation. This thesis carries out research on the design of the signal processing stage, the statistical pattern recognition stage and concludes with the application of the linguistic stage. The thesis is organised as follows:

- The second chapter presents the motivation for launching speech recognition research out of Information Retrieval.

- The third chapter gives a literature survey of the state-of-the-art of pure speech recognition research and investigates speech recognition research in the context of Information Retrieval.

- The fourth chapter describes the speech recognition technology in detail from the signal processing step up to the modelling of linguistic units. It shows how many components are involved on the acoustic side of speech recognition.

- The fifth chapter describes the design of the acoustic front-end. It is based on the theory of chapter 4. The emphasis is on the immense parameter space which has to be set up. A combination of experiments and literature study is used to conquer the parameter space. The chapter is completed with an evaluation.

- The sixth chapter outlines the design and implementation of a speech recognition evaluation interface. This interface is needed for the evaluation of the acoustic front-end described in chapter 5 in a real environment.

- The seventh chapter, states the domain beyond the acoustic front-end. It shows the benefit of the application of standard language modelling techniques. Experiments show some interesting features of phoneme-based speech recognition also in regard to text-based Information Retrieval. This chapter also demonstrates what can be done with the acoustic front-end in general.

- The final chapter contains the conclusions.

Chapter 2

Project Description

2.1 Introduction

Speech is the most natural, convenient way of exchanging information. In contrast to typing speech communication requires no special skills, both hands are free for other tasks. Learning to speak and to understand a language is not done to communicate with machines but to communicate between humans. Speech is human culture.

> *"The universal goal of all disciplines of Computing Science should be to adjust the machines to the human not the opposite".*

The aim of speech recognition (SR) is to create machines that can receive spoken information and respond appropriately to that information. Such machines approximate human's communication.

Written language is derived from speech and like speech is itself a human culture. Today, as most text documents are created and stored electronically, hand writing has been replaced by typing but still requires the knowledge of a written language. From the perspective of Computing Scientists, writing can be regarded as a special technique for storing information. If an ideal speech-to-text translator existed then a written language would lose its attractiveness to some extent. Anyway, such an ideal system is not available yet and therefore the more philosophical question of the meaning of written language need not to be answered now.

Electronic text documents are needed furthermore. Modern Information Retrieval (IR) tools like NRT [1] are used to find information in large text document bases. To access the information, queries are formulated by users. Up to now, queries have been formulated by typing a message, related to the users' information need. Retrieval strategies, described in [2][3], are used to extract query related information from the document base. The question is why type when speaking is so convenient. It is clear that this results from the limitations of the current SR technology.

Current speech recognition systems are not perfect compared to humans capability in understanding speech. Many problems remain unsolved. The main problem of speech

recognition lies in the variability of speech. A message can be pronounced quite differently by varying many speaker-dependent parameters. This variability enables humans to understand a variety of different pronunciations of a given language but variability means on the other hand uncertainty which must be handled by the speech recognition system.

In the end of eighties two efficient approaches, (1) the statistical approach by using Hidden Markov Models (HMM), and (2) the connectionist approach by using artificial neural networks (ANN), were established. These techniques make modest and constrained speech understanding applications feasible. At present such constrained applications work satisfactorily in a limited context. If the context is natural then the constraints of the SR system are transparent to the user in such a way that the user does not necessarily recognise the weakness of the SR system. For instance, in an evaluation of an automatic train timetable information system accessible over telephone [4], it has been shown that most users were not aware that speech recognition does not work with 100% accuracy.

The question is where do the constraints come from? The constraints for a SR system come in general from the characteristics of the input language to be processed. The more restricted the input language the less is the uncertainty which must be handled by the SR system.

2.2 A Constrained Speech Recognition System

A Spoken Query Recogniser (SQR) constitutes a speech interface for a text-based IR system. It is a constrained SR application which works in a natural limited context. The constraints for a SQR come from the properties of the queries. Below are some general characteristics of queries stated with the benefits for a SQR. There is no particular IR technique assumed.

- Queries are normally short. They consist only of a few words. This is a benefit in favour of space and time complexity of applied SR algorithms. For instance, consider a typical query to an IR publications database: "Logic and Uncertainty". This spoken query lasts about 2 seconds. Using a standard sample frequency of 16 kHz for digitising that utterance gives a total of 32,000 samples. If the samples are represented by 16 bit integers then the query has a length of 62.5 kByte. The essential time is the delay between speech input and speech interface output. Current standard SR techniques can only begin to process when the utterance has been completed. Assuming a processing time of 3 seconds, which is reasonable from the user's point of view, and a standard Von Neuman computer with a machine command cycle time of about 20 ns then the SR algorithms have 150,000,000 commands available to process 62.5 kByte speech.

- The core of queries are content-bearing words. They contain often more than one syllable. Therefore in general they are easier to recognise [5]. Furthermore, content-bearing words have at least one strong syllable and strong syllables are easier to recognise than weak ones because the vowels are fully pronounced in strong syllables as opposed to weak ones. Consider the above used example query. The word "Lo-gic" has two syllables where the

first syllable is a strong one and the word "Un-cer-tain-ty" has even four syllables.

- Stop words like prepositions are the hardest to recognise. They are usually not considered in IR. Lee et. al. [6] reported that stop words are responsible for about 50% of the recognition errors which was established on the SPHINX SR system. For instance, stop words like "a, of, to" and "the" are so short that speech recognisers cannot separate them from content-bearing words.

- Finally, a query spoken by the user to a machine is generally better formulated in contrast to conversational speech because the dialog can be controlled by the machine. The benefit is that word and phoneme merging effects are smaller. This point is only true if the user is aware that he is speaking to a machine. For instance, if the user communicates via a telephone connection then he need not necessarily be aware that he is speaking to a machine. For instance, consider the query: "Information Retrieval". In a normal conversation an IR person would pronounce this phrase in one stream so that the final "n" of information and the initial "r" of retrieval would merge to one non distinguishable sound.

Thus, the impact of uncertainty in speech recognition can be minimised by exploiting the above restrictions. On the other hand, some characteristics of a query are not ideal for SR:

- Queries do not necessarily constitute a grammatically correct expression. Typically, it is a set of loosely coupled words. Therefore, syntactical information above the word level cannot be used to reduce uncertainty.

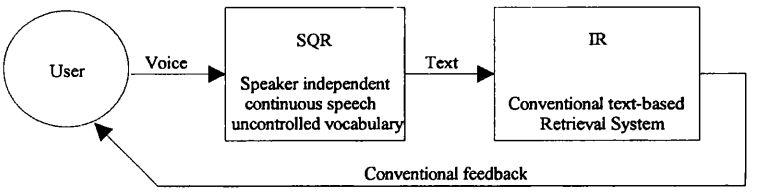
- The index vocabulary of practical IR applications is uncontrolled and can reach the size of the human mental lexicon. So, the vocabulary of a SQR, respectively its lexicon must also be uncontrolled.

- Typical IR applications are not restricted to an individual single user. Queries may be formulated by many different users. Therefore, the SR system must be speaker-independent.

- Queries are spoken fluently. This means a SQR must be able to process continuous speech. In general, artificial breaks between words provide a great advantage for speech recognisers. However it is totally inconvenient for the user community with the result that they will not accept this obstacle.

2.3 A Speech Interface for a Text-based IR System

The long-term goal is to develop a SQR which exploits not only the above mentioned constraints but which also addresses an unlimited vocabulary as well as speaker independence and continuously spoken speech. An unlimited vocabulary means that the vocabulary used by the IR application should be a subset of the lexicon of the SQR. It is known that this goal can only be approximated with the currently available techniques and may not be completed within a normal MSc research period. To be open for further developments and to be independent from a special technique, the framework described earlier in [7] will be applied. It gives the project a great flexibility and has some other interesting properties. The task of the SQR can be summarised as converting the spoken query into the corresponding orthographic transcription. The orthographic transcription is then applied to the IR system. The retrieval result is presented in the conventional manner as determined by the IR system. Figure 2.1 states the role of the SQR.



SQR: Constrained Speech Recognition System
IR: Limited Context

Figure 2.1: SQR, a speech interface for a conventional text-based IR System.

The project can be subdivided into 4 parts:

1. Building a suitable speech recognition engine using HMM technology;

2. Finding an optimal linguistic decoding strategies for obtaining keyword terms from a spoken query;

3. Integrating the speech interface with the information retrieval system;

4. Evaluation of the effectiveness and efficiency of information retrieval.

The major parts of the project are 1 and 2. The speech recognition engine which is also called the acoustic front-end will be the main concern of this thesis. The acoustic front-end is the lowest level of the SQR and all further parts rely on the acoustic front-end. The design of the acoustic front-end also determines how flexible the SQR will be in terms of the above mentioned requirements. The task of the acoustic front-end is essentially the mapping of

speech sounds to the corresponding linguistic units. The mapping function is realised by Hidden Markov Models. There are many decisions which have to be made in designing the acoustic front-end. Although the Hidden Markov Model technology provides a powerful tool for the acoustic front-end, there is still no generally accepted way how these techniques should be applied. There is still much scope for research.

Chapter 3

Feasibility Analysis

3.1 Introduction

The previous chapter described the goal of the SQR from the point of view of the requirements set by an interface for a text-based IR system. In this chapter, the project is considered from the speech recognition side. It will be determined whether it makes sense at all to build a speech interface for an text-based IR system by just using some basic technology. There must also be doubts as to whether the goals like uncontrolled vocabulary and speaker independence are not too high or even unrealistic for the current stage of SR technology. We first have a look at the current stage of related research especially in pure speech recognition research.

3.2 Related Work

Currently large industry labs and university labs are engaged in the area of Speech Recognition. Although Speech Recognition is not a solved problem in any fundamental sense, basic theory has been established which is sufficient for some specific commercial applications. The market for speech recognition applications is growing rapidly. In 1990, the US market for speech processing equipment was already estimated to exceed $1 billion per year with an increase of about 30% per year [8]. This boom has intensified speech recognition research and has led to high expectations. Nevertheless, a variety of original research and commercial applications has been produced.

3.2.1 State-of-the-Art of Speech Recognition

The examples for a speech interface for an IR system are applications with the primary goal of converting speech into the corresponding text. Therefore, such applications which come from the big speech recognition labs, must be investigated before addressing IR specific work. Table 3.1 states the properties of some of the most successful systems from university as well as industrial side.

System	Created	Speaker dependency	Speaking mode	Vocabulary size	Perplexity	Correct words %
Tangora	1985	dependent	isolated word	5000	160	96.9
INRS	1990	dependent	isolated word	86000	700	92.8
SPICOS	1988	dependent	continuous	900	74	93.0
SPHINX	1989	independent	continuous	1000	60;1000	94.7;73.6
DRAGON	1991	independent	continuous	1000	60	94.6
BYBLOS	1991	dependent	continuous	1000	60;1000	96.2;81.2
Decipher	1991	independent	continuous	1000	60;1000	95.2;82.4

Table 3.1: Performance of some speech recognition systems.

The perplexity is a measure for the complexity of the grammar which is used in the recognition task. All applications in Table 3.1 use a language model which has build-in prior knowledge of the language in question. The perplexity measures how many words on the average can follow a recognised word. For example, a perplexity of 60 of a 1000 word recogniser means that after a word has been recognised, a choice in recognising the successive word must be made between 60 words on the average. If the perplexity was 1000 then each word could follow each other word with equal likelihood. Despite the age of the systems, they are still up to date research projects and further improvements are expected.

Since 1992, the Advanced Research Project Agency (ARPA) has annually sponsored the evaluation of SR systems. In November 1994, fifteen research groups presented their speaker-independent continuous speech recognition systems. For the test, 10 male and 10 female speakers read approximately 15 sentences each from articles drawn at random from 237 million word corpus of North American business and general news. In the test, the systems were allowed to use any language model and as large vocabulary as practicable, and they ran with incremental speaker adaption. The top three systems' word-error scores were: Cambridge University Engineering Department (CUED) 7.2%, IBM T.J. Watson Laboratory 8.6%, and

CNRS-LIMSI from France 9.2% [9]. It has to be emphasised that the speech was read which is different from free spoken speech and is easier to recognise because of a better articulation and a slower speaking rate. However, despite the constraints of speaker adaption, the results are quite promising considering that this could not have been achieved a short time ago. It is also an interesting fact that the top three systems of the ARPA evaluation and of the systems given in Table 3.1 all use the Hidden Markov Model technology as the speech recognition engine. Although it should not lead to the conclusion that the second promising approach, the neural network technology, is ruled out. It is just a younger approach and so much effort has not been spent in it compared with the Hidden Markov Model technology.

DragonDictate

One of the most powerful commercial systems is DragonDictate from Dragon Systems. It is designed for document creation by voice and is offered in several versions which differ in vocabulary size and language. It is available for Unix platforms and for 486 PCs. It consists of an active vocabulary which is held in the RAM and a passive vocabulary which is on the hard disk. The access of the words in the active and passive vocabulary happens like the swapping or paging technique of operating systems. The latest version for American English operates with an active vocabulary of 60,000 words and an passive vocabulary of 60,000 words. It works with so called discrete speech and is speaker adaptive. This means a user must speak with a pause of 0.1s between the words and has to speak 45 minutes to the system for the speaker adaption stage before the system can be used. The main criterion in judging such a system is the time taken to produce a document of a certain size compared with the conventional type writing techniques. This time includes also the time to correct word errors. The general result of such an evaluation [10], on a version with 30,000 active and 50,000 passive words, was that DragonDictate can be used to input typical business text with an efficiency equivalent to the fastest alternative means. The error rate has been approximated to one word in three.

There are also other systems with similar performance but the systems described above are quite representative.

3.2.2 Speech Recognition in the Context of Information Retrieval

The Information Retrieval community has also recognised the potential of speech recognition. Information Retrieval has been discovered not only to be important for retrieving text documents but also for retrieving information represented in other media. There is a great demand for Multimedia Information Retrieval systems generated by the general Multimedia market. The IR research which is being done with speech media will be addressed in this chapter.

Multimedia retrieval requires a retrieval model which can coherently cope with different

media. Currently, research is concentrated on speech associated with text, although speech could be linked pictures. Figure 3.1 shows the possible scenarios of IR with speech and text. The diagram shows four major components: user, IR engines, speech and text document base, and speech to text, text to speech converter respectively. What is the difference between the external converter devices and the direct speech and text input arrows? In IR the full queries and documents are not processed but rather the representatives of queries and documents. Representatives of documents and queries are the so called index terms which constitute the index language. The index language can be in any form as long as they fulfil the basic IR principles: (1) suitable for a computer to use, (2) able to discriminate documents from each other, (3) able to describe topics precisely. The task of generating the index language belongs to the IR engine. It is applied to the query as well as to the document base. Now an IR system with both speech and text input as well as speech and text document bases requires a common single index language. This index language must fulfil the mentioned requirements and must be recognised both in speech and text. It is not an easy task to find such an index language but only then could we speak of a proper Multimedia IR engine. The converter devices could then disappear.

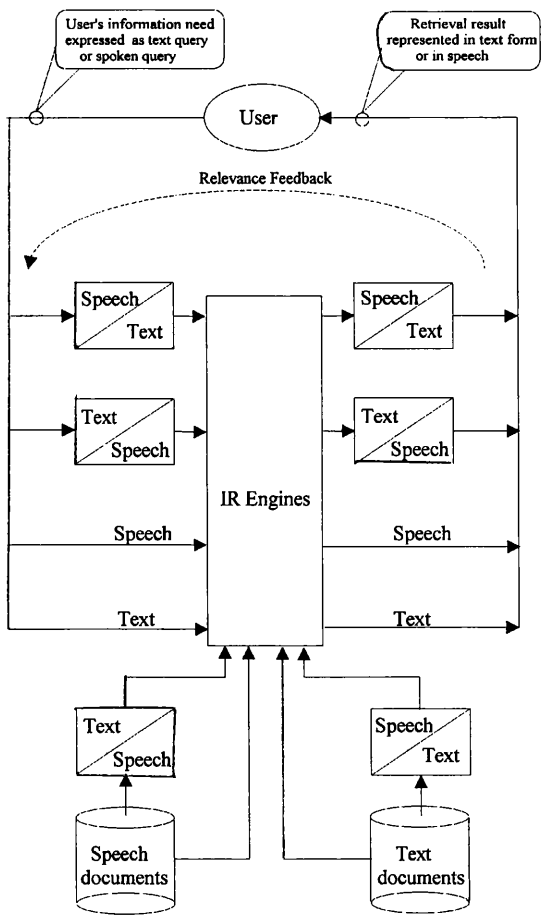


Figure 3.1: Possible scenarios of Multimedia Information Retrieval with speech and text.

Another possibility is to use IR engines which are especially tuned to either speech or text and convert speech into text or text into speech by external converters. Therefore, there would be several IR engines adjusted to the corresponding media with several appropriate index languages. It is certainly easier to design media specific index languages. For example, the proven conventional IR engines can be used for retrieving text documents from spoken queries. On the other hand, the properties of the index language can also help to make the recognition of speech and text easier as stated in section 2.2 by exploiting the properties of queries.

The output of Figure 3.1, usually a set of citations or document numbers, plays a minor role for the retrieval effectiveness. However, one note has to be made. The output is usually stated in text form but one could also imagine that a spoken output is required. This demand is, for example, for IR systems which can be accessed over telephone lines. If the output, e.g. document numbers, must be converted from text to speech and the converters work not 100% correctly, the output affects also the retrieval effectiveness especially when relevance feedback is used.

Now considering the Multimedia IR model of Figure 3.1, different work will be presented which uses either a media specific index language or a Multimedia index language. From users' view it does not matter whether there is a proper IR engine with a Multimedia Index language or whether there are several IR engines with external converter devices. The user judges the system only according to the outer behaviour which can be summarised as system's effectiveness and efficiency.

Pioneering work has been done in retrieval of speech documents by ETH Zürich. In 1992, Glavitsch & Schäuble [11] presented a retrieval model which retrieved speech documents. They devised an index language which is not only suitable for indexing speech documents but also for text documents. They called the index language a set of index features. Thus, they can use a single (Multimedia) IR engine for retrieving speech documents for text queries, or text documents for spoken queries due to the special set of index features. The index feature set was designed to be appropriate for speech recognition and to fulfil the mentioned IR principles. They discovered that such index features lie between words and phonemes and thus they used vowel consonant combinations as index features, similar to core syllables which are minimum combinations of vowels and consonants. These index features are modelled phonetically for identification in speech documents by a speech recognition component. The phonetic modelling is based on Hidden Markov Models. Text documents must be presented in the same phonetic transcription as used for the index features for retrieving it from speech. The index features in the text are identified by a parser. Speech data was hardly available for evaluation at that time. Therefore they evaluated the retrieval effectiveness of this index language by using text queries as well as text documents. The result was that the index language can compete with conventional index languages. The speech recognition component affects the retrieval performance additionally because of misrecognised index features. The SR component was evaluated with a single speaker, 13 word vocabulary, and isolated words. The recognition rate was high because of these constraints. This was an example of a Multimedia IR engine with a single index language. The speech recognition component was based on some constraints, also it must be considered that at that time, there was no toolkit

available for building a speech recognition component and recorded speech data, which is necessary for modelling index features, was also hardly available, so that they had to do a lot of basic work. Wechsler & Schäuble [12] are continuing the work. They have slightly changed the index feature set to tetragrams and have also based their work on phonetic Hidden Markov modelling. For the speech recognition component they have applied the Hidden Markov Model toolkit HTK [13] which has become available in the meantime.

Other work for indexing speech documents has been done by Wilcox & Bush at Xerox PARC [5]. Unlike ETH, they use the so called wordspotting technique which is also based on Hidden Markov modelling but with a different strategy. Wordspotting on continuous speech has been introduced by Rose & Paul [14]. The basic principle is to spot content-bearing words, called keywords, in a continuous stream of speech. This stream can be a query or a speech document. It is distinguished between non-keyword speech (spoken words that are not keywords) and keyword speech. Each keyword is modelled by an HMM which represents the most likely realisations (reference pattern) for each keyword. The non-keyword speech is modelled by just a single HMM. Wordspotting is done by calculating similarity scores between the speech from the speech document (or the query) and the keyword models and the non-keyword models. The computed scores and a decision rule are then used to decide whether a keyword has been detected or not, see Figure 3.2.

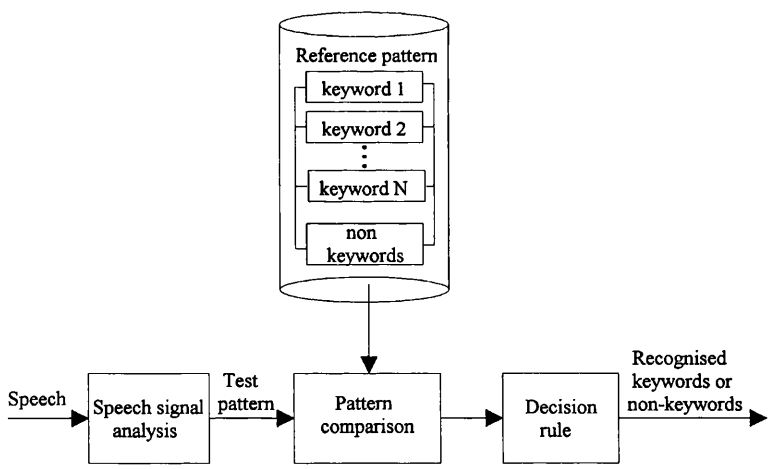


Figure 3.2: Wordspotting.

In [5], wordspotting is applied to speech documents and the queries are provided in text but the other way around is possible. So, they receive textual words from both the queries and the speech documents. These words are then passed to a conventional IR engine. That is why the wordspotting task corresponds to an external speech-to-text converter in harmony with Figure 3.1. However, the wordspotting technique uses the mentioned benefit of IR in the need of just recognising content-bearing keywords as opposed to the difficult stop words. Furthermore, there are some problems in modeling words with HMMs, which limits the number of keywords to a modest size. These problems are addressed in section 5.3.

A further application of wordspotting related to IR is currently under development at Cambridge University in association with Olivetti Research Limited [15]. The task is to retrieve video mail messages from a message archive by text queries. Retrieval is done by applying wordspotting, in the same manner as mentioned above, to the spoken part of the video mail message. The spotted words of all currently archived messages constitute the index vocabulary of a conventional text-based IR engine.

Finally, some work will be mentioned which has the same goal as addressed by this thesis. Kupiec et al. [16] have as their goal the retrieval of information from large unrestricted text corpora by voice. They use an HMM based phoneme recogniser. This recogniser converts the spoken query into phonemes. The resulting phoneme string is noisy. The phoneme string is matched against a phonetic described lexicon to obtain the words. They try to compensate the noise by so called "Semantic Co-Occurrence Filtering" which basically uses the context of a word to correct possible errors and to resolve ambiguity. Their system currently operates in a speaker-dependent, isolated word mode.

3.3 Conclusion

In has been shown that HMM technology is currently more successful and reliable than the Neural Network technology. Therefore this technology is a good choice for our approach. The widely used HTK toolkit is a further argument to go for an HMM approach. The SPHINX system and related work from Carnegie Mellon University serve as a good example for applying Hidden Markov Model technology: the progress of SPHINX has been well documented in several papers over the past decade.

Wordspotting seems to be a reasonable technique which exploits the advantage in recognition of just content-bearing words as opposed to stop words. However, there are two plausible main arguments against the use of wordspotting:

1. Significant information is thrown away by just spotting the keywords and ignoring other words. Even if the other words, like the stop words, are finally not used by the target application, e.g. the IR system, they can provide important information for correcting errors of the keywords. For example, syntactical and semantical relationships can be exploited.

2. The discrimination between keyword and non-keyword speech may converge to zero if the number of non-keywords increases compared to the number of keywords. The number of keywords can anyway not cover a large vocabulary because of some limits from the Hidden Markov Model technique which will be discussed in several places in this thesis. For example, the Video Mail Retrieval System from Cambridge [15] which uses wordspotting, accepts currently just 35 query words. This fact emphasises the weakness of wordspotting. There plans are to enhance wordspotting with techniques adopted from large vocabulary speech recognition systems [17].

The pure speech recognition research on large vocabulary speaker-independent continuous speech recognition systems is at a remarkable stage and it may be that soon speech interfaces become the norm. Especially a speech interface for an IR system, for which the requirements are much smaller than for a dictation system, will almost certainly be realised soon. In a way that a speech-to-text converter device, sketched in Figure 3.1, will be available. It is reasonable to expect that such a speech-to-text converter will be completely accommodated on single digital signal processor (DSP) chip so that the main CPU is relieved. For speech input an A/D device is needed anyway which demands a special hardware interface so that it is less problematical to extend this hardware interface by a DSP.

In conclusions. It does not make sense to go for a special index language just for the problem domain of text retrieval and using wordspotting is not feasible for large vocabulary. For example, Croft et. al. [18] applied as speech interface DragonDicate, which is a commercially available large vocabulary recogniser, to their text retrieval system. However, the situation is quite different in retrieval of speech documents because the task of converting a whole speech document to text is in general a greater problem as it is for a short query. Therefore, it is worth looking for a special index language for indexing speech documents.

All the systems described so far, have their individual strengths but it has also been shown that each system has its constraints. Thus speech recognition is still basic research and there is still demand for further research and still space for alternative approaches and views. Speech recognition itself, is a technique which involves the areas signal processing, pattern recognition, and linguistics. Working in the speech recognition area might also yield some kind of synergy effects for Information Retrieval. For example, the in SR widely used HMM technique may demonstrate alternative ways for handling uncertainty of other data than speech. Also the following restated assessment of the latest NSF workshop report [19] intensifies the decision in addressing such a project.

> *"Basic research, leading to new high risk approaches, and supporting educational and training activities, to ensure a steady flow of well-trained scientists to tackle these problems, ... is needed".*

Chapter 4

Speech Recognition

4.1 Introduction

The project goal has been defined in the previous chapter. One may have the impression that the project can be realised in a structured way without any further research. However, this view is false. Although many good techniques are now available, speech recognition is still a field of basic research. Compared to human capability of speech understanding, present recognition systems are far behind it. Humans can deal with unfamiliar accents, wrong grammar, unknown words, and background noise. Current SR systems cannot cope with such realistic conditions. That speech recognition is still fundamental research is also shown by the fact that two different approaches, the statistical and connectionist, are currently under development. At present it cannot be foreseen which approach will be the best. Thus, a speech recognition system cannot be completely built by using an existing toolkit as is possible for graphical applications. Some questions remain to be answered. A speech interface, satisfying users' need, will only be realisable by exploiting task constraints, e.g., provided by the IR task domain.

It must be understood why speech recognition is so difficult, before speech recognition techniques are introduced. But first some often used terms will be explained to prevent any confusion, and the main reasons are described why speech recognition is so hard, then basic techniques that try to deal with the problems are introduced. The Hidden Markov Model technique will be described in more detail because it is the basis for the acoustic front-end designed in this approach. Section 4.5 to section 4.7 show the complete process of statistical speech recognition in detail, starting from the conversion of the sound pressure wave and ending with the modelling of linguistic units. It shows where the huge parameter space which has to be set up comes from.

A recent published study [20] from the Center for the Study of Language and Information (CSLI) is an assessment of the state-of-the-art of speech recognition.

4.2 Important Terms

Linguistic Units

Linguistic units are entities bearing linguistic meaning. Linguistic units are, for example, phonemes, syllables, words, phrases, sentences, texts and also identifiers like "C" and "V" for denoting consonants and vowels respectively. Concatenation as a binary operator can be applied to linguistic units to form higher level linguistic units, for example, phonemes can be composed to form words.

Phonemes

Phonemes can be considered as the elementary sound units of speech. You can come across them, e.g., in the Oxford English Dictionary. These phonemes constitute the basic sound set to describe the pronunciation of the English language.

Phonemes will play an important role in this document, therefore it is appropriate to define what a phoneme actually is, and to clearly separate it from other, often confused, related terms. One must be precisely distinguish between phonemes, phones, allophones, phonetics, and phonemics.

The phoneme has been defined in several ways by phonologists. The so called functional approach has been generally accepted [21]. According to this definition, a "phoneme" is defined as the smallest sound unit which is still sufficient to distinguish linguistic meaning. For example, the phonemes called "/p/" and "/b/" distinguish the words "bin" and "pin". The phoneme conveys linguistic meaning. Each phoneme can be considered to be a label for a unique set of articulatory gestures. A phoneme can be regarded as the interface between the acoustic side (speech production, perception, transduction) and the linguistic side of speech recognition. The study of phonemes and their relationship in a language is called "phonemics". The actual sounds that are produced in speaking, without regard to their linguistic meaning, are called "phones". The study of the phones of a language is called "phonetics". The phoneme is rather a theoretical unit, used as transcription for the pronunciation of words. Phonemes cannot be reproduced exactly by the human. Therefore with each phoneme is a set of "allophones" associated that represents slight acoustic variations of the phoneme. Allophones represent the permissible freedom allowed within each language in producing a phoneme.

Most languages like English, German, have their special phoneme set. The phoneme set constitutes the phonemic alphabet. There are also phonetic alphabets, but the difference is that phonetic alphabets include diacritical markers like stress, and intonation information. However, the relevant literature does often not distinguish between phonemic and phonetic alphabets.

The phoneme is an important unit in SR because of its dual character to classify acoustic sounds, and to describe words. Nevertheless famous linguistics like Chomsky and Halle do

not support the phoneme as the basic linguistic unit. They stated about the phonemic representation [22]:

> *"We feel, however, that the existence of such a level has not been demonstrated and that there are strong reasons to doubt its existence".*

They describe sounds with a sophisticated binary feature matrix with features like "+vocalic", "-consonantal", "+stress", where +(-) means sound has (not) feature. This view, though, plays in today's SR research a minor role, but it should emphasis that the existence of phonemes is still only a working hypotheses.

4.3 Basic Problems of Speech Recognition

Many problems make speech recognition very difficult. The main difficulty is the variability of speech which causes many subproblems. This variability causes one particular sound with a particular meaning to have a variety of different acoustic realisations. But the acoustic realisation, namely the speech waveform, is the carrier of the information and the human listener can recognise speech only by processing the speech waveform. So a machine listener must ideally also be able to recognise speech just out of the speech waveform. (One could argue that human has in a face-to-face communication additional visual information available like lip movements, sign language, and gestures. This is true and actually some research is being done to exploit this kind of information. However such information is not always available and a human's perception is not dependent upon such information; an extreme example: telephone communication.) It is interesting to think about the reason of the variability. It means redundancy from a standpoint of the coding theory and redundancy can be used to correct errors. So if we regard an ideal communication system where each speech sound with a certain meaning (e.g. the sound for the word "a") has its unique acoustic realisation then the information can only be perceived by the listener when the acoustic realisation has not become noisy during transmission. However, the acoustic waveform always becomes noisy by several influences at different stages of the transmission channel as showed in Figure 4.1.

The variability already arises in the early stages of human's speech production system. A message which has been formulated for speaking, is encoded into a sequence of basic unique sounds, let us call them phonemes. Each language has its specific set of sounds. Stress markers, pitch accents, duration, and loudness are formed and associated with the phonemes during the sound coding task. These associated parameters are strongly dependent on the local context, the acquired speaking skills, and the mood of the speaker, so that phonemes can be realised quite differently. The acoustic waveform is produced in the human's vocal tract system where the larynx, glottis and vocal cords are the basic organs in the generation process. However, the vocal-tract system is not only designed for generating speech. It is also used for other functions like breathing and eating. For example, the main task of the vocal

cords is to avoid the entering of solid and liquid particles into the lung. This shared functionality might be the reason that the vocal-tract system is a highly non-linear voice production system as opposed to an acoustic tube for example. The shapes of the organs of the vocal tract are dependent on the individual and also on time (age). Also fundamental parameters of the vocal-tract system like the pitch frequency, which is the excitation signal of the vocal cords for voice sounds, are quite different between children, women, and men. The acoustic environment contributes a further remarkable part to the variability of speech which is known as the noise. The non-linearities of the microphone and the Analog/Digital converter cause further degradation of the speech waveform in a speech recognition system. This variability causes many problems for the speech recognition task. Some of the hardest problems for speech recognition will now be summarised in the following.

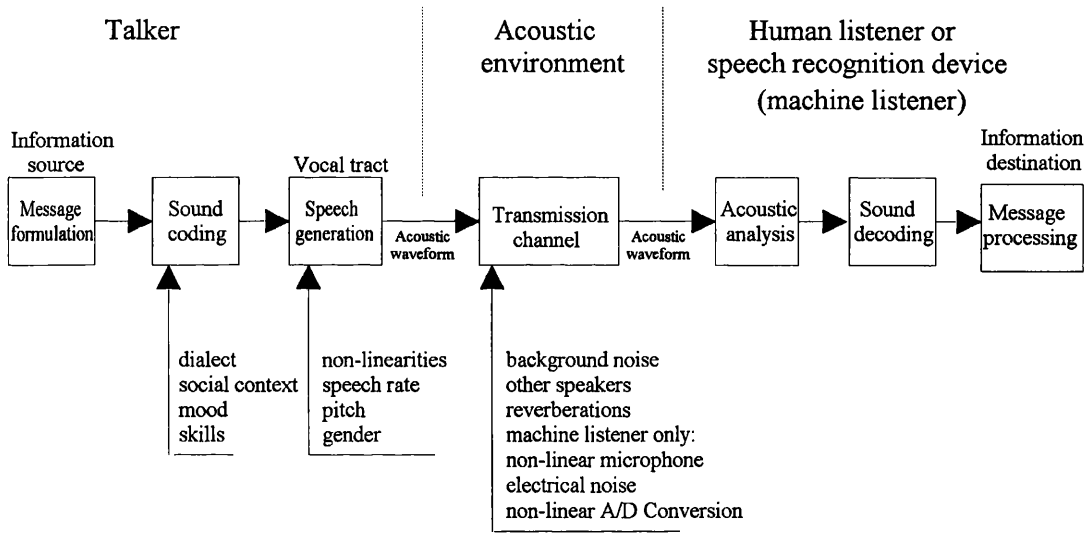


Figure 4.1: Source of variability of speech.

4.3.1 Boundary Problem

Figure 4.2 shows the speech waveform of the words "information" and "retrieval" spoken in isolation and spoken in a fluently manner. There is no separator, no silence between words in contrast to blanks in written text. The question is, where the exact word boundary is? The answer is, that there is no exact word boundary. The area between the two vertical lines marks the uncertainty region of the word boundary between the words "information" and "retrieval".

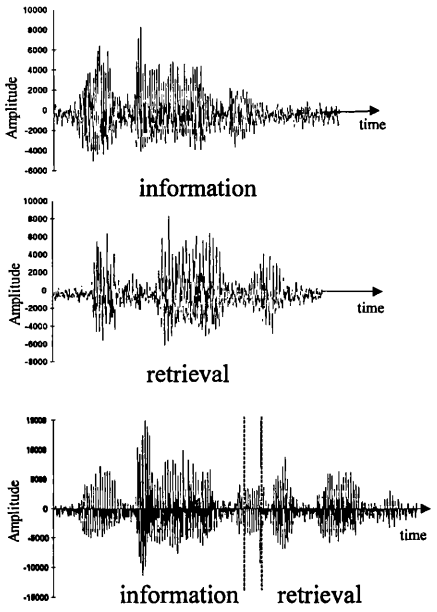


Figure 4.2: Boundary problem.

4.3.2 Coarticulation Effect

As explained in section 4.2, phonemes are practical units for speech recognition but there is one main problem involved, which makes the recognition of phonemes more difficult. This problem is the fact that adjacent phonemes influence each another. This happens because of the fact that when one phoneme is pronounced, the pronunciation of the next phoneme is prepared by the movement of the vocal tract. So it is quite different whether a word is spelled or it is fluently spoken. The phonemes influence each another within words as well as between word boundaries. In the latter case, it is additionally complicated by the boundary problem. Figure 4.3 shows the phoneme "/u:/" spoken alone and spoken in context of the word "to" ("/tu:/"). The second speech waveform shows that the phoneme "/u:/" is totally modified by the preceding phoneme "/t/".

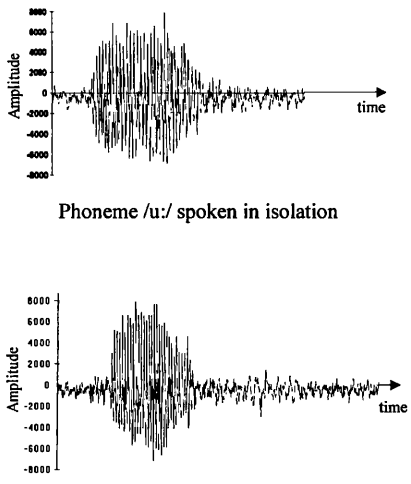


Figure 4.3: Coarticulation effect.

4.3.3 Basic Speech Sounds are Unique

The speech waveforms in Figure 4.4 show two german phonemes "/i/" spoken by the same speaker.

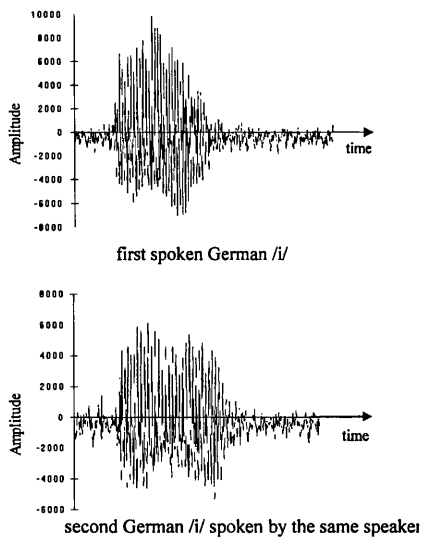


Figure 4.4: Two German "/i/" spoken by the same speaker.

The problem is, that the same phoneme cannot be repeated twice not even by the same speaker, without changing the physical representation. Although, phonemes serve to distinguish sounds, respectively utterances, their physical realisation varies according to the influence values stated in Figure 4.1. Therefore, a set of allophones is associated with each phoneme which represent slightly acoustic variations of the phoneme.

This scope is a great advantage to the human as it makes learning and understanding a language possible. But, this scope increases the uncertainty a speech recognition system has to deal with. This uncertainty is one of the main drawbacks in SR.

4.3.4 Energy of Vowels Versus Consonants

Vowels have a strong chance of detection because they possess a characteristic spectrum. They are excited by a periodical signal function and have therefore certain frequencies in their spectrum. Each vowel is characterized by its own set of frequencies. Vowels posses the most energy in their spectrum in contrast to consonants. On the other hand, consonants are a very complex group of sounds which are quite different in their generation and properties.

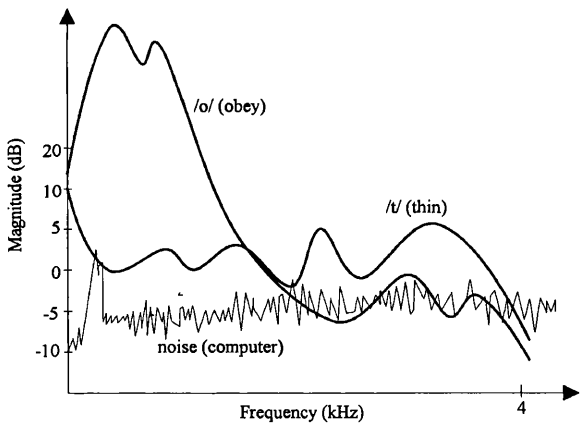


Figure 4.5: Energy of vowel versus consonant; speech understanding in function of frequency.

The magnitude spectrum (Figure 4.5 at the top) of the vowel "/o/" versus the consonant "/t/" emphasises the contrast in their energy content. This is the reason, why in general vowels are more easily recognised than consonants. In particular, unvoiced consonants like "/t/" have less energy in their spectrum than vowels. The low energy content of some speech sounds causes also another problem. That is the distinction between environmental noise, which is more or less always present e.g. the noisy computer fan, and the low energy sounds. The distance of the spectrum between the noise curve and the "/t/" curve is quite small in contrast

to the vowel curve (Figure 4.5 at the top). Human auditory perception is relatively insensitive to environmental noise. If the noise is four times higher than the speech [23], humans can still understand. However, an algorithm cannot simulate such a complex mechanism as the human ear and is dependent on the physical representation of speech. Speech energy should be at least 1000 times (30 dB) greater than noise energy. A proportion of 1,000,000 (60 dB) is ideal for good speech recognition.

Consonants are not easily recognised, however they bear major information as the following example illustrates:

Try to guess the string without consonants!

Term without consonants:

_a__ - _oi_ _e_

Now try to guess without vowels!

The same term without vowels:

b_ll - p__nt p_n

4.3.5 Ambiguity of Phonemic and Orthographic Transcription

A further problem is the ambiguity of phonemic pronunciation in respect to the written language. For example, the phonemic pronounced word "/tu:/" has three corresponding counterparts in the written language indicated by Figure 4.6.

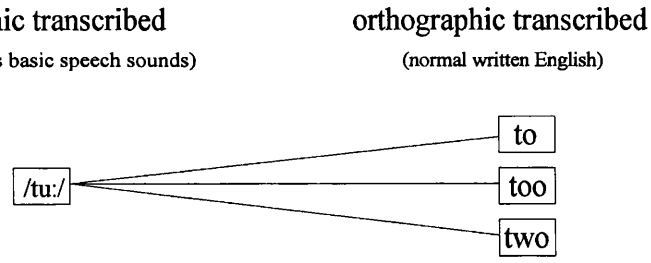


Figure 4.6: Ambiguity of phonemic and orthographic transcription.

This ambiguity can only be resolved on a higher linguistic level by syntax or semantic constraints.

These were some major problems in speech recognition. Another problem is, for example, detecting of prosodic features of language, which are characterized by intonation and stress. The typical English phrase "Isn't it". Is it actually a question or only a note which depends on the stress of this utterance?

The question is: What can be done to conquer these problems?

4.4 Classification of Speech Recognisers

A lot of modest constrained applications have been developed, for example, the Tangora system from IBM, DragonDictate from Dragon, or SPHINX from Carnegie Mellon University. The constraints make it possible to get around or to avoid some of the problems mentioned in the previous sections. Consequently, speech recognisers are classified according to the combinations outlined in Figure 4.7.

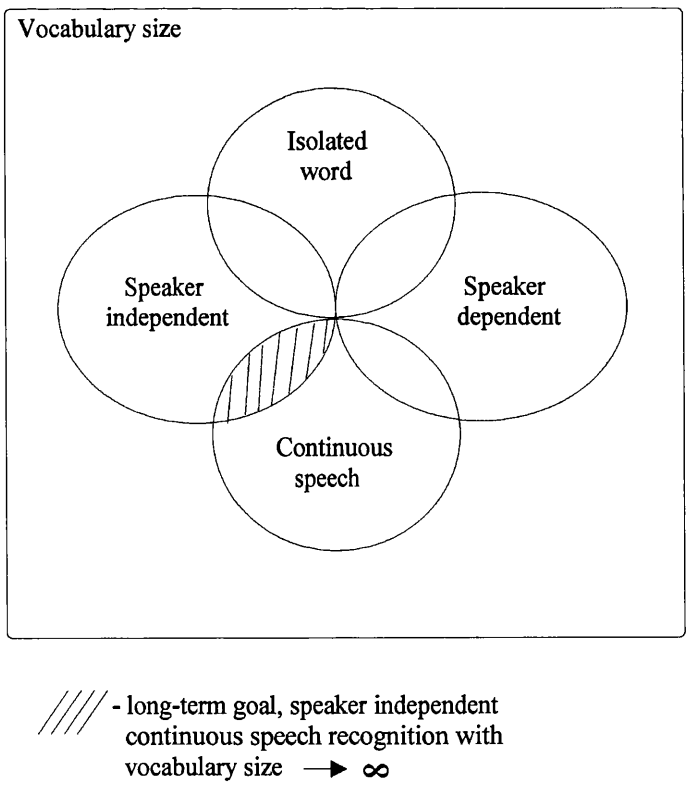


Figure 4.7: Classification of speech recognisers.

"Speaker-dependent" and "isolated word" systems with small vocabulary, e.g., 100 words,

are applications with the lowest complexity.

"Speaker-dependent" means that the system is only able to recognise speech from a particular speaker for who the system was trained. The system is then used specifically for recognising the speech of its trainer. The obvious disadvantage of a speaker-dependent system is the need to retrain the system each time it is to be used with a new speaker.

"Isolated word" means that words must be spoken with an artificial break, that means with about 100 ms silence between them. Such systems with 100 words show error rates below 1%.

The next are "speaker-independent and isolated word" systems with small vocabulary: Speaker-independent means that all speaker, speaking the language for which the SR system has been designed, are admitted, e.g., speaker-independent digit recognisers work with word error rates less than 1.5% [24].

"Continuous speech" means natural spoken speech. For example, the latest release of the Tangora system from IBM [24] which works speaker-dependent has a word error rate of 5.4% with a vocabulary size of 20,000 words (large vocabulary).

To date the most popular "speaker-independent continuous speech" system is SPHINX from Carnegie Mellon University. It was evaluated on 4200 sentences from 105 different speakers. The word error rate was 9% [6].

Speaker-independent continuous speech systems with large vocabulary of about 50,000-100,000 words are the current challenge of SR research. Speaker-independent continuous speech systems with unlimited vocabulary, which means that the vocabulary size is transparent to the user, are the vision for the near future.

Some systems offer the so called "speaker-adaptive" mode which means that the speech recogniser must be adjusted to the voice of the individual user in a pretraining session. The user usually has to read some displayed sentences, often the documentation of the tool, for example, 251 test sentences must be read before IBM's personal dictation system [25] can be employed. Speaker-adaptive systems are based on speaker-independent trained system, where new training data from individual speakers is used to improve the performance for the individual speakers. Speaker-adaption can also be performed online whilst the system is already in operation. Speaker-adaptive systems can be grouped into speaker-dependent and independent.

4.5 A General Speech Recognition System

Speech recognition is a fundamental classification problem. Segments of the acoustic speech waveform are mapped to classes with certain meanings corresponding to the desired target pattern. Classes can be any linguistic unit, e.g., phonemes, syllables words, or sentences. This pattern recognition problem can be tackled in various ways. However, all ways have to consider the special characteristics of the speech signal: Figure 4.4 shows that the raw speech waveform is an inappropriate representation for reference patterns. Therefore the raw speech waveform must be prepared by extracting those speech features which can be used to classify

linguistic units. The properties of the classes must be known before a classification algorithm can be applied. This is done by a training, learning or clustering step in which reference patterns, or reference models are derived from the same kind of speech features which are used for recognition. The classification step provides a transcription of the speech signal into linguistic units. Depending on the granularity of the unit, a language processing step must be applied to get the desired target units, e.g., phonemes must be connected into words. Thus a general speech recognition system works according to the steps outlined in Figure 4.8. It is needless to say that speech recognition is fully realised in digital computers either in special purpose digital signal processing (DSP) hardware or in general purpose computers. Therefore the steps in Figure 4.8 are implemented in software or directly in hardware.

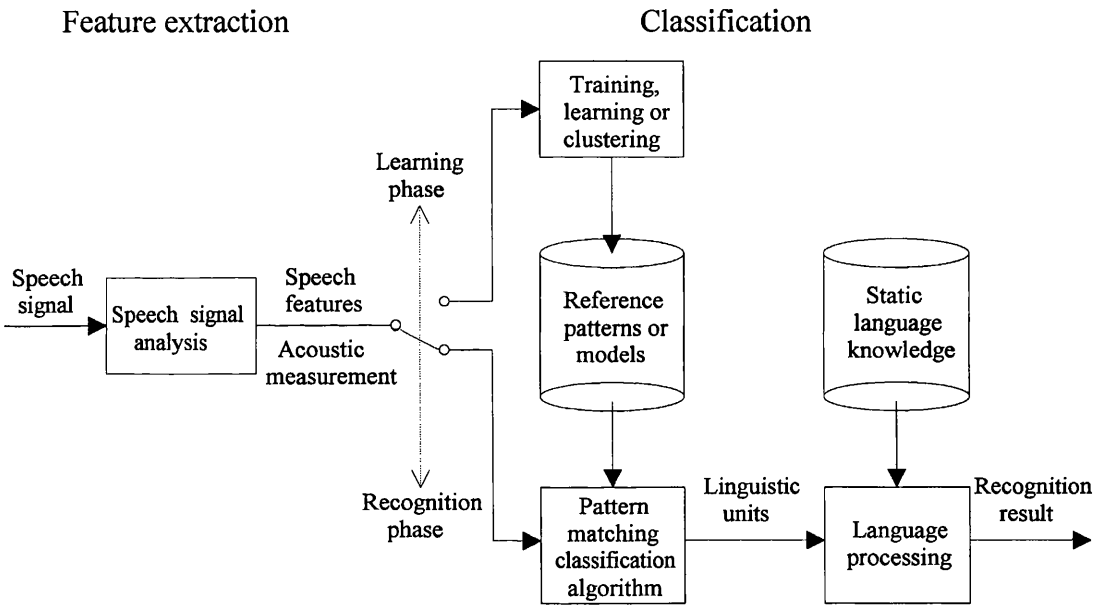


Figure 4.8: Basic steps of a general speech recognition system.

4.5.1 Feature Extraction

Since, the raw speech waveform is an inappropriate representation, the main task of the feature extraction is to extract features that characterise sounds. The speech waveform is nothing else than a signal over the independent variable time. The time dependent magnitude is after the digitising process a finite value which is proportional to the sound pressure of the sound wave. Such signals are best analysed with the tools of digital signal processing (DSP). The analyse can be done either in the time or the frequency domain because of the correspondence between the time and the frequency domain. The most common techniques

exploit properties of the human's auditory system and analyse the signal in the frequency domain. Ideally, the analysis technique should be insensitive to speaker and channel variability. The kind of extracted features also depends on the employed classification method.

4.5.2 Classification Step

The extracted features can be used in various ways but basically they must be matched to a reference pattern or a reference model. The reference patterns or models must be derived in a learning step. The learning step depends on the classification method being used. It can be done manually by reading the spectrogram of the speech signal and looking for speech features and appropriate classes, or it can be done automatically by supervised learning procedures. In supervised learning procedures, training samples of speech utterances (with each labeled as to its correct class) are used to estimate the parameters of a reference model or to cluster patterns. We can distinguish between four basic classification methods:

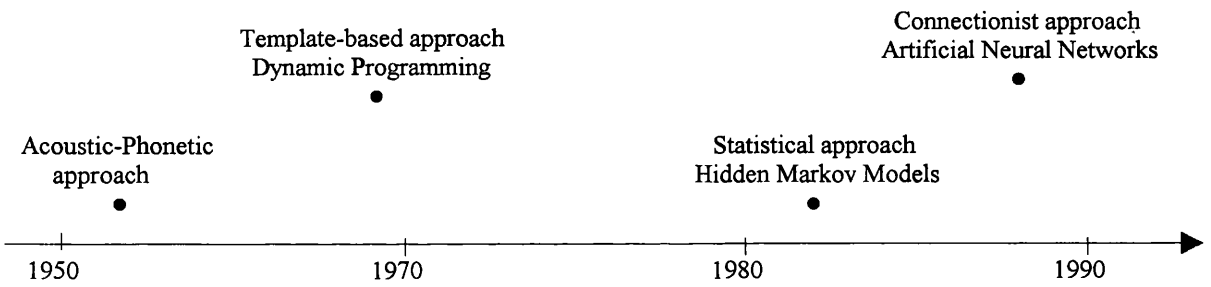


Figure 4.9: Speech recognition classification methods.

Acoustic-Phonetic Approach

One of the earliest attempts in automatic speech recognition was the Acoustic-Phonetic approach in the 1950s. It is based on the assumption that the phoneme is the basic linguistic unit which can be precisely described by speech features. The phonemes were studied intensively and classified according phonetic knowledge about the production of phonemes, place of articulation, manner of articulation, and type of excitation. This task belongs to the learning step according to Figure 4.8. It is the major responsibility of the feature extraction step to analyse the speech signal for those speech features (see Figure 4.10) that characterise the phoneme directly and precisely. Therefore the speech signal analysis is the core of the speech recogniser in this approach. A decision logic, e.g., the decision tree for the vowel "/i/" in Figure 4.10, is often used to classify the phonemes according to their features.

The main problem with this approach was that the available phonetic knowledge which

could be analysed was too inaccurate and there was also a general doubt about the quality of the phonetic knowledge. This approach plays only a minor role today. A further problem could be that the approach was based on phonemes as the basic linguistic unit. There might be a lower or a higher level unit, e.g., syllable which is more appropriate for such a method.

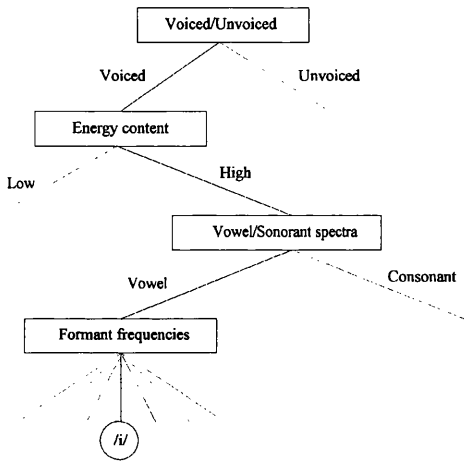


Figure 4.10: Acoustic-Phonetic classifier for the vowel phoneme "/i/".

Template-based Approach

The template-based approach is based on Dynamic Programming which was taken directly from Pattern Recognition research. It dominated Speech Recognition in the 70s and led to the first serious commercial applications.

The principle is simple: the test pattern is just matched against a set of reference patterns so called templates. All reference patterns together constitute the vocabulary. The patterns are a set of speech features representing a linguistic unit, e.g., a phoneme, or a word. The reference patterns are obtained in a training session by clustering similar segments of speech features into appropriate classes. The cluster centroid is then used as reference pattern. Using some distance measure, the pattern with the smallest distance is the best match.

One problem is, that speech utterances always have different lengths. For example, a "reference" pattern for a phoneme "/h/" consists of 12 sets of speech features (SF) and the corresponding "test" pattern of the phoneme "/h/" consists of 18 sets of speech features. The problem is that the length of the test pattern "/h/" is not known in advance. This belongs to the recognition task and addresses the boundary problem. The challenge is to find a time warping function which maps similar acoustic events (similar sets of spectral features) of the test pattern and the reference pattern, see Figure 4.11. This is done by "Dynamic Time Warping" algorithms which look for the minimum distances between all possible alignments of the test pattern against every reference pattern. The Human Interface Lab in Tokyo (NTT) was one of the first to use Dynamic Time Warping [26].

The advantage of this method is:

+ Simple algorithms.

And the disadvantages are:

- Number of limitations, e.g., storage of a sufficient number of templates and computational cost of search becomes inefficient.

- Using subword units, like phonemes as templates, is ruled out, because accounting for the coarticulation effect of phonemes is insufficient.

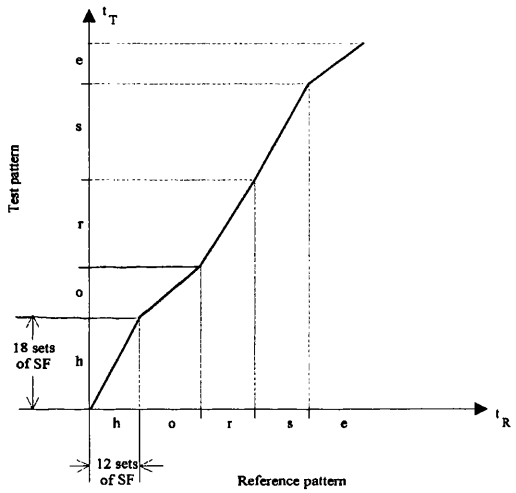


Figure 4.11: Dynamic Time Warping.

This technique is mainly used for small vocabulary systems: 100 words speaker-independent, 500 words speaker-dependent; but it has nearly been replaced by the statistical approach with Hidden Markov Models.

Statistical Approach

The weakness of the template-based approach was essentially that it could not deal sufficiently with the uncertain character of speech. A single reference pattern is used as representative for a linguistic unit, e.g., one set of speech features is used to represent a phoneme. Even if the reference pattern is derived by clustering methods which exploit statistical techniques, there is only one deterministic set of parameters which describes a linguist unit. However, a speech signal is basically a random process over time which cannot precisely described with deterministic parameters. The problems of speech recognition,

described in section 4.3, have shown the effects of the random process. The random process has ergodic properties and such processes can be characterised by statistical methods which has led to the statistical approach namely Hidden Markov Models. Baker and his colleagues at Carnegie Mellon University [27] and Jelinek and his colleagues at IBM [28][29] were the first to use Hidden Markov Models for Speech Recognition in the early seventies, however, the big breakthrough occurred in the mid eighties. The linguistic unit is represented by a probability distribution of its speech features as opposed to a single set. This distribution is accommodated in the HMM. So the HMM represents several reference patterns and depending on the kind of HMM it can represent even a infinite number of reference patterns. The problem of the comparison of patterns with different lengths as mentioned in the template-based approach is completely handled by the HMM because of its inherent property of self-normalising of time. The main advantages of this approach are:

1. The HMM can be trained from speech by an automatic supervised learning procedure.

2. There are powerful mathematical tools which allow efficient training and recognition when some restrictions are placed on the HMM.

3. HMMs are appropriate for realising statistical Regular Grammars because of the equivalence to Finite State Machines (FSMs).

The first two points are the main basis for HMM-based speech recognition. But, large vocabulary speech recognition is the present challenge. It requires sophisticated language models. A special kind of language modelling is directly supported by HMMs because of the equivalence to FSMs which itself are equivalent to Regular Grammars. For example, an HMM can model the linguistic unit phoneme. A phoneme is the interface to the speech waveform which will become clear in the following sections. A phoneme HMM can also be regarded as a terminal symbol of a Regular Grammar defining a Regular Language. A word can be regarded as an intermediate symbol of the Regular Grammar which can be built just by concatenating phonemes or actually HMMs. The word itself is then just one big composite HMM which can be analysed with the same methods as the phoneme HMMs.

The most important drawbacks of the HMM framework are:

1. First order Markov assumption. The consequence is that the speech features are modelled as independent acoustic phenomena which is never true.

2. The supervised learning procedure works as Maximum Likelihood Estimator (MLE). This means that the likelihood of the HMM of recognising a linguistic unit is maximised with the correct speech patterns and not minimised with the incorrect speech patterns. The consequence is a lack of discrimination power between correct speech and incorrect speech.

The first mentioned point can be relieved by some kind of tricks or extension of the HMM framework. The second drawback is also a serious issue. In section 5.11 it can be seen that phonemes, which sound similar, are responsible for typical recognition errors. The lack of the

discrimination power resulting from the MLE training procedure is certainly one reason. Therefore, finding new training procedures is an active research topic.

Despite the drawbacks, SR based on HMMs dominates today's speech research and development. My approach is to use HMMs and therefore I will describe them in more detail later in section 4.7.

Connectionist Approach

Another approach that was introduced at the end of the 80s, is the connectionist approach using Neural Networks. At present, the Hidden Markov Model technique is mainly used in commercial applications which indicates that this technique is more advanced. Recent research results [30], however, have shown that Neural Networks must be taken seriously and that more will be expected in the near future.

Neural Networks are used to model linguistic units in a way similar to HMMs. They can be used, for example, to map speech sounds to phonemes. Multilayer perceptrons or recurrent networks also known as Hopfield Network are the most common used models. These networks are trained by a supervised learning procedure like the HMMs. The training procedure minimises the distance to the correct class but also maximises the distance to the incorrect class in contrast to the MLE training for HMMs. Furthermore, the networks have no first order Markov limitation, so that the dependency of acoustic phenomenas can be modelled. On the other hand, training procedures for Neural Networks are considerably slower than for HMMs. A Neural Network is not necessarily equivalent to a Finite State Machine so that it is not always possible to concatenate, for example phoneme Neural Networks to word Neural Networks.

4.5.3 Language Processing

Depending on the granularity of the pattern, further steps are necessary to identify the desired target pattern. For example, the output of the pattern matching stage could be phonemes but words are desired. Knowledge about the language in question can be used to build the words out of the phonemes. A simple way to do this is by using a phonemic described lexicon. The lexicon provides static knowledge about the language. The language processing stage will be the most important part to tackle the great challenge of unlimited vocabulary of next generation's speech recognition systems.

4.6 Speech Signal Analysis

Before applying speech recognition techniques, the speech signal must be analysed. Hidden Markov Models, Neural Networks, or the reference patterns of the template-based approach have turned out to be inappropriate for modelling the raw speech waveforms.

The purpose of speech signal analysis is not only to extract linguistic information from the speech signal but also to discard any other information. A part of the speech signal is responsible only for the transmitting task and acts as an information carrier. The speech signal analysis task is also often called redundancy removal. However, this term must be used carefully because the nature of speech signal redundancy depends mainly on which part of its information will be used. For example, information which identifies the gender of the speaker is usually not important for this approach but may be for other applications. Furthermore, there is no evidence that the speech signal analysis techniques currently in use provide the optimal information for speech recognition. The only reason for relying on these techniques is that no better alternative has yet been discovered. The question still remains of whether these techniques are sufficient to tackle the complete speech recognition problem.

A view of the information theory should show the discrepancy between the speech signal and the information conveyed by it: taking a phoneme set with 50 phonemes, which is a realistic size for many languages, e.g., English, these phonemes can be coded with about 7 bit. Taking a speaking rate of 120 words/min, which is quite representative for colloquial speech, and taking 5 phonemes per word on average, gives an average data rate of 70 bit/s which is received by us when we recognise phonemes. Taking the speech signal, which is sampled (digitised) with 16 kHz and each sample is represented with 16 bit, gives a data rate of 64 kbit/s. A typical spectral analysis technique, usually applied to the speech signal, would reduce the data rate by a factor of 2. So, the difference between the data rate of the speech signal and the speech regarded as a sequence of phonemes received by the human is quite obvious. It must be considered, however, that the received phoneme sequence is pure information, unlike to the speech signal.

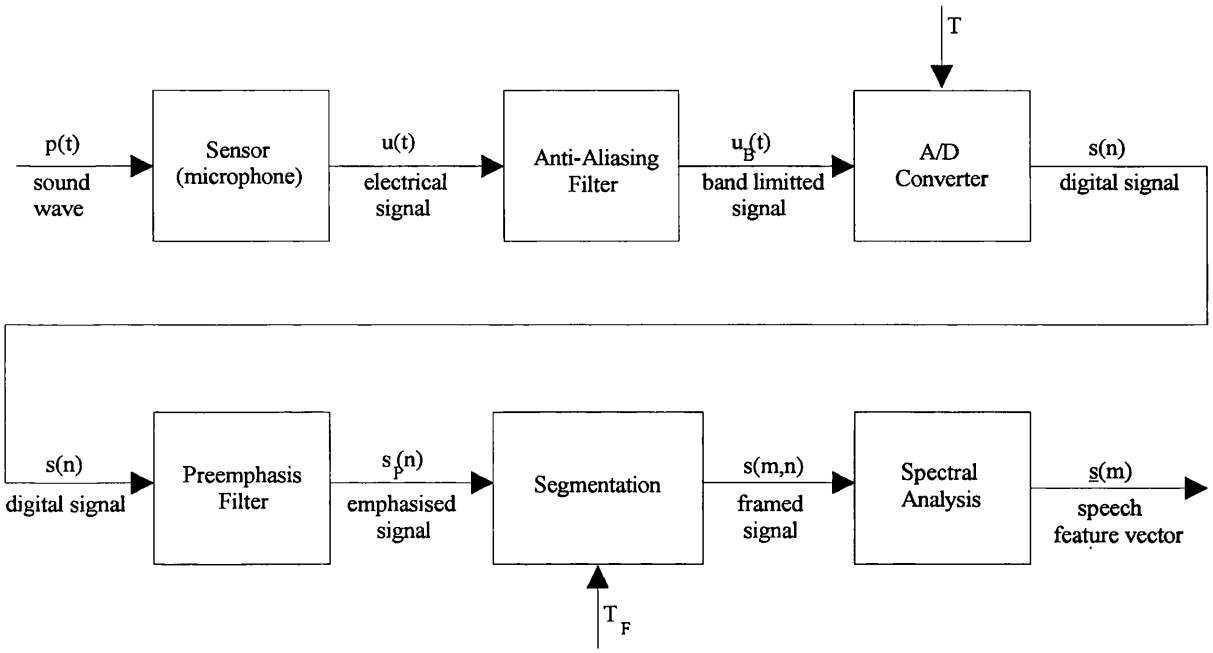


Figure 4.12: The sequence of operations in converting a sound pressure wave to speech features suitable for classification in speech recognition.

Speech analysis is based on the function of the human auditory system. It generates perceptually meaningful parameters. There are many different analysis techniques used in speech recognition, e.g., Fourier transforms and Linear Prediction. A good overview of speech signal analysis techniques with respect to SR is given in [31]. The two most important analysis techniques will be introduced in the following sections. Before these techniques can be applied to the signal, however, it must be digitised and prepared in several pre-processing steps.

These pre-processing steps are often regarded as a matter of course in SR systems because the components are often built-in in speech recognition hardware, e.g., the standard audio card in general purpose computers. However, these components have a significant effect on the quality of the speech signal, and thus of the speech recognition rate. The real recognition task, which is accomplished in the classification step, is computationally rather expensive, and correcting recognition errors is even more so. It is worthwhile spending some effort on the pre-processing step, which is relatively easy compared to dealing with speech recognition errors. The pre-processing steps are explained first. These steps contain already many important parameters of the speech recognition system. Figure 4.12 shows the steps of the process undergone by the speech as sound pressure wave, before it is supplied to the classification step.

4.6.1 Sensor

Today computers are usually equipped with a standard microphone. However, speech recognition performance is heavily dependent on the quality of the microphone used. Such a quality loss may cause high computational costs or even irreparable damage, so it is important to choose a high-quality microphone.

Speaking causes changes in air pressure. These changes constitute a sound pressure wave which conveys the speech through space. The sound pressure wave must be converted into a corresponding sequence of integers so that the speech can be electronically processed. This is done in several steps. The special characteristics of speech must be considered even at this early stage.

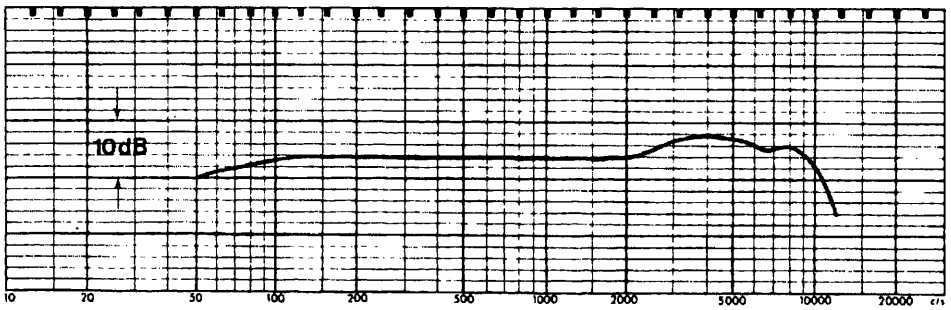


Figure 4.13: Transfer function of the close-talking microphone Sennheiser HMD 414.

The microphone is the sensor for the complete speech recognition system. It converts the sound pressure wave into a proportional electrical wave. The requirement for a microphone is that it must guarantee a constant transfer function between the sound pressure and the voltage in a given frequency range and sound pressure range. The challenge lies in the highly dynamic nature of the sound pressure, which covers over six powers of ten. The frequency range which can be perceived by a human is 16 Hz - 20 kHz. This range can be regarded as the maximum worthwhile to be considered. There are three main types of microphones used, the electrodynamic-, electrostatic-, and piezoelectric- microphones. The electrodynamic microphones offer the highest quality. Figure 4.13 shows the transfer function of a high quality electrodynamic close talk microphone which is often used in speech recognition experiments. The curve shows the relation between the microphone input, which is the sound pressure, and the microphone output which is the voltage, as a function of the frequency. The relatively constant transfer function in the range of 50 - 12,000 Hz can clearly be seen. The tolerance of 4 dB is relatively small considering the size of the dynamic range of speech which is about 40 dB.

The position of the microphone has a great influence on the quality of the speech output. Reverberations of the sound pressure wave due to the characteristics of the room also distort

the speech signal and thus have an effect on the speech recognition rate. Close-talking microphones, where the microphone is in a fixed position directly in front of the mouth, reduce that effect. Therefore, such microphones should be preferred to desktop microphones.

Another serious problem is the background noise which is nearly always present. The main sources are the computer, which is usually next to an SR system, the natural environment, e.g., other people's noise, and the electro-magnetic field caused by electrical mains hum. Noise-cancelling, particularly for speech recognition and speech transmission, is a growing research area. There are many methods which try to compensate for the noise part of the signal. Most methods are based on filter operations applied to the digitised speech signal. However, the microphone is the sensor which collects the noise. The microphone itself may also be used to reduce the noise. Pressure-gradient microphones are dynamic microphones especially designed for noise-cancelling. They work according to the principle that noise coming from a relatively longe distance is cancelled.
Close-talking microphones are also of this type. The Sennheiser HMD 414 is an example of such a special noise-cancelling microphone.

The sensor (the microphone) provides the speech signal in the form of an electrical wave which will be digitised.

4.6.2 Anti-Aliasing Filtering

The speech signal must be band-limited before it can be digitised due to the ambiguity of the sampling process. This means that the frequency range of the speech signal must be limited. The sampling theorem from Nyquist, which is the theoretical basis of the digitising process, declares that a time continuous signal with a spectrum of the width:

$$B = 0...f_g \tag{4.1}$$

can be completely described by periodically scanned samples, if the sample period fulfills:

$$T \leq \frac{1}{2B} \tag{4.2}$$

Thus, the sample frequency comes to:

$$f_t = \frac{1}{T} \tag{4.3}$$

and the spectrum of the speech signal must be in the range:

$$f_t \geq 2B \tag{4.4}$$

If Equation (4.4) is not satisfied, the frequency components of the speech signal become ambiguous, and therefore the sampled speech signal no longer represents the original signal. Figure 4.14 shows the effect of Aliasing. The spectrum of the speech signal is periodically repeated with the period of the sample frequency due to the sampling process (A/D conversion). This is a mathematical fact. However, the A/D converter has an inherent filter operation which cuts out only that part of the spectrum which is less than the half sample frequency. If the sample frequency is twice as much as the highest frequency component of the speech signal, that part corresponds to the original spectrum of the analog speech signal (see case 1 of Figure 4.14). However, if the sample frequency is less than twice as much as the highest frequency component of the speech signal, the spectra are overlapping (see case 2 of Figure 4.14) and the sampled signal becomes distorted.

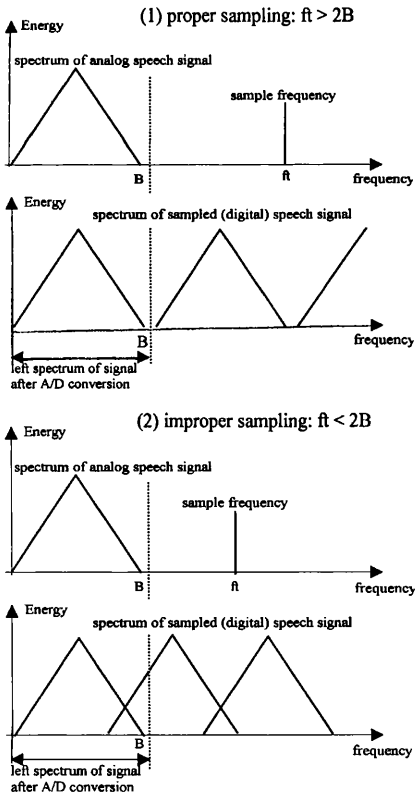


Figure 4.14: Aliasing effect of the sampling process.

This explains the need for the Anti-Aliasing Filter which limits the spectrum of the speech

signal to half or in practice even less, of the sample frequency, due to non-ideal filters. The filter operation can be formally expressed by:

$$
\begin{aligned}
U_B(f) &= U(f)\ H(f) \\
H(f) &= \begin{cases} 1, & for\ f < \dfrac{f_t}{2} \\ 0, & for\ f > \dfrac{f_t}{2} \end{cases} \\
U(f) &= \mathcal{F}\{u(t)\} \\
U_B(f) &= \mathcal{F}\{u_B(t)\}
\end{aligned}
\qquad \textbf{(4.5)}
$$

where the first equation of (4.5) denotes the filter operation in the frequency domain. The second equation is the filter equation for an ideal low-pass filter, although that equation can only be approximated. Practical filters have no rectangularly formed transfer function. The top is inflected with overshoots, the vertical edge at $f_t/2$ is a slope with undershoots towards H(f)=0. This causes additional distortions to the speech signal. Typical low-pass filters are the Cauer filter and Butterworth filter. The last two equations show the relation between the time and frequency domain of the two signals given by the well-known Fourier transform. It is clear from Equations (4.4) and (4.5) that the bandwidth of the speech signal which can further be used, depends directly on the sample frequency. The sample frequency can be regarded as one of the system parameters of the complete speech recognition system. A sample frequency that allows the use of the whole spectrum of the speech signal would be ideal. However, the following practical points must be considered:

1. A higher sample frequency causes a higher data rate and thus requires more computing power.

2. The sample frequency cannot be selected arbitrarily because it is a basic constant parameter of the A/D converter and must therefore be chosen according to the commercially available A/D converters (Hardware dependency).

3. Above a certain frequency, the speech signal cannot any longer be distinguished from the noise signal, which is almost always present, caused by electrical mains hum and non-ideal components (upper bound of frequency).

So, the question which needs to be answered is: which spectrum or which frequency range of the speech signal is important?

The output of this stage is a band-limited analog speech signal $u_B(t)$ which is directly passed to the A/D Converter.

4.6.3 Analog/Digital Conversion

The task of the A/D device is to convert the continuous speech signal into a signal which is discrete in time as well as in amplitude. The computer, the target platform for automatic speech recognition, can only handle sequences of numbers with finite precision. This is the simple reason for this twofold discretisation process. Discretisation, however, means quantisation, which means distortion or approximation of the original. The discretisation of the time and amplitude happens, in practice, in one step in the A/D device, but it is better to deal with it separately in order to understand the two processes and its consequences for the signal.

Quantisation of time is, in practice, not problematical because the sample theorem stated above says that the original signal can be reconstructed as long as Equation (4.2) is satisfied. The Anti-Aliasing Filter takes care that Equation (4.2) is always satisfied. Hence, discretisation in time is realised by sampling the signal with the period T, formally:

$$s'(n) = s'(nT) = \begin{cases} U_B(t) \quad for \ t = nT \quad n = 0,1,2,... \\ undefined \quad else \end{cases} \tag{4.6}$$

Note s'(nT) is a time discrete sequence with continuous amplitude. It is important for total understanding that s'(nT) is not zero between the samples but simply not defined. In the following only s'(n) will be used instead of s'(nT). This short-hand notation also has a theoretical justification because the speech signal, is after the sampling process, just a sequence where the integer n indexes the sample number. The sequence itself bears no further relation to the sample period. This means that the sample period must be stored at any place for devices which need the reference to the time again.

The quantisation process of the amplitude is more critical; s'(n) still has its original amplitude. The samples must be quantisised into numbers with finite precision. This causes distortions of the amplitude. Linear quantisation is the simplest way to represent a sample by R bits. It can formally be expressed by:

$$\Delta = \frac{2\ U_{AD,MAX}}{2^R}$$

$$s(n) = \begin{cases} 2^{R-1} - 1 & for\ \dfrac{s'(n)}{\Delta} > 2^{R-1} - 1 \\ k & for\ k - \dfrac{1}{2} < \dfrac{s'(n)}{\Delta} \leq k + \dfrac{1}{2},\ k = -2^{R-1}+1, .., -1, 0, 1, .., 2^{R-1}-1 \\ -2^{R-1} + 1 & for\ \dfrac{s'(n)}{\Delta} \leq -2^{R-1} + 1 \end{cases} \quad \textbf{(4.7)}$$

Equation (4.7) looks confusing but it demonstrates all the problems of linear quantisation. First, it must be remembered that the amplitude of the speech signal $u_B(t)$ represents a voltage. The quantisation task is nothing other than a mapping function of a given range of voltage to a limited number of code words. The number of available code words is given by 2^R. So the given voltage range can be expressed by 2^R code words. This causes a quantisation with a resolution of Δ in the linear quantisation case. Note that the quantisation of s'(n) by Δ causes a dimensionless signal s(n). The voltage range $\pm U_{AD,MAX}$ is a parameter of the A/D device and is, of course, finite. This causes the first problem with the speech signal. It must be guaranteed that the amplitude of the speech signal $u_B(t)$ is always in this range. If these bounds are crossed then the consequence is that the amplitude is clipped, and this means considerable distortion which directly affects the speech recognition quality. The first and the last term of Equation (4.7) show that s(n) is kept constant when the bounds are crossed. It can be conclude that the voltage range of the A/D converter should be chosen to be large enough. However, given the same number of bits R, the resolution Δ decreases when a greater voltage range is selected. However, this causes higher distortion, in particular for small amplitudes, and is therefore a basic problem for speech signals due to their highly dynamic amplitudes. Equation (4.7) is stated for the commonly used two-complement coding scheme. The term in the middle of Equation (4.7) maps the values lying in the quantisation interval to two-complement code words. It can be regarded as a rounding device which causes the samples of s(n) to be always uncertain by $\pm\ \Delta/2$. This is the reason why the speech signal is distorted by the amount $\pm\ \Delta/2$. The rounding error can formally be expressed by:

$$s(n) = s'(n) + e(n)$$

$$p(e) = \begin{cases} 1 & for\ -\dfrac{\Delta}{2} \leq e \leq \dfrac{\Delta}{2} \\ 0 & else \end{cases} \quad \textbf{(4.8)}$$

Therefore, the speech signal s(n) is always inflected by an error or noise signal e(n), where e(n) is a discrete random process with an assumed uniform distribution and the probability

density function p(e). Equation (4.8) shows that the relative error is high for small amplitudes. The extreme case is when the amplitudes are in the range of e(n). The distortion caused by e(n) is called quantisation noise. A measure to assess the quantisation noise is the Signal-to-Noise-Ratio SNR. The SNR is defined as the ratio between the average power of the speech signal and the average power of the noise signal:

$$SNR = 10\log \frac{E[s(n)^2]}{E[e(n)^2]}\ dB \qquad \textbf{(4.9)}$$

The average power must be considered as the expected value because both s(n) and e(n) are discrete random processes. It is clear that the higher the SNR, the less the distortion of the speech signal. Practicable SNR values are over 30 dB. However, other sources of distortion, like the microphone, must also be considered. It can easily be shown that, by assuming a Laplace distribution for the speech signal amplitudes, and by assuming the above stated uniform distribution for the noise signal, the following equation can be used as an approximation for the SNR:

$$SNR(R) = 6R - 7.2\ dB \qquad \textbf{(4.10)}$$

Equation (4.10) states the SNR caused by the quantisation noise of an R-bit A/D converter. Equation (4.10) must be considered carefully. It is based on the assumption that the speech signal uses the full dynamic range of the AD converter. If, for example, only half the range is used then R must be replaced by R-1 for calculating the SNR. The conclusion is that the dynamic range should be fully exploited to increase the SNR. However, this cannot be done arbitrarily because there must be a safety margin to avoid clipping the amplitudes. Table 4.1 shows the SNR for different R-bit A/D converters using linear quantisation.

R (bit)	8	10	12	14	16
SNR (dB)	40,8	52,8	64,8	76,8	88,8

Table 4.1: SNR for different R.

Table 4.1 can be used to provide practical guidelines for the assessment of linear A/D converters. The values give only an approximate view because of the random processing character of speech and its highly dynamic nature. Subjective hearing tests [32] have shown that to obtain an SNR of 40 dB, 12 bit is required. Quantisation noise is only one of many noise sources distorting the speech signal, so a 16 bit A/D converter is recommended for speech recognition.

The final output of the A/D converter is a sequence s(n) which is discrete in time and amplitude.

4.6.4 Preemphasis Filtering

This is a simple high-pass filter operation which is described by the equation:

$$s_p(n) = s(n) - K \times s(n-1) \qquad \textbf{(4.11)}$$

where K is a constant, and typical values for K are in the range of [0.95, 1.0]. The higher frequency components of the speech signal are increasingly raised by this filter equation (see Figure 4.15). There seems to be no existing publication which evaluates the speech recognition rate using this filter. Therefore, the significance of this parameter must be well understood. In the literature, there are two different reasons given for recommending the use of this filter. The first reason is that this filter compensates for the spectral contributions of the larynx and the lips, so that, after applying it, the signal consists only of the contributions of the vocal tract. This is based on the idea of a speech production model with the vocal tract realised as a technical filter. This filter is excited by an excitation signal modelling the air pressure wave coming from the lungs. Many speech signal analysis techniques are based on finding the parameter set describing the vocal tract filter and the excitation signal. The removal of other contributions from the speech signal leads to more precise vocal tract parameters. The second reason is that it has been argued [24] that the speech signal becomes less sensitive to finite precision effects by applying Equation (4.11).

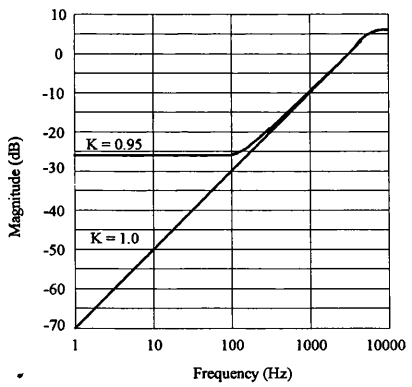


Figure 4.15: Frequency response of the preemphasis filter for typical values of K.

4.6.5 Segmentation

The speech signal is basically a random process over time, where the sound pressure magnitude is the random variable. It cannot be fully described by deterministic parameters like harmonic waves. However, the signal varies only slightly in intervals of 10 - 50 ms. This property is the basis for most spectral analysis techniques. It can be stated that the speech signal is quasi periodic in certain intervals. In other words, the speech signal is divided into segments of equal duration, typically in the range of 10 - 50 ms. The segments are successively cut out of the speech signal. This task is called windowing because there are several techniques for cutting segments out by so-called windows. The simplest window is a rectangular window. This means that only the unchanged speech samples of the segment will be used as input for the later spectral analysis stage. This has the disadvantage of adding additional higher frequency components to the speech signal because of the abrupt cut-off on the segment border. This distortion can be avoided by using smoother windows in which the samples near the segment border have a lower weight than the samples in the middle of the segment. Smoother windows are thus prefered in speech recognition [32]. The so-called Hamming Window is a smoothing window which is commonly used in speech recognition. Using a Hamming Window, the segment of the speech signal is multiplied by the function,

$$w(n) = \begin{cases} 0.54 - 0.46 \cos(\frac{2\pi n}{N-1}) & for\ 0 \leq n \leq N-1 \\ 0 & else \end{cases} \qquad \textbf{(4.12)}$$

where the N is the segment duration expressed in the number of samples.

However, it is not known when the interval where the speech signal is regarded as quasi periodic is beginning to change. This is the reason, why in practice, overlapping segments are used. To obtain overlapping segments, the whole speech signal $s_p(n)$ is divided into frames of equal duration, say T_F. The speech signal can now also be regarded as a sequence of frames, say s(m) as opposed to a sequence of samples. Then a segment of duration T_w is taken at every frame. Overlapping happens for $T_F < T_w$. The difference between the segment and frame duration determines the averaging between the segments. Figure 4.16 illustrates the segmentation of the speech signal.

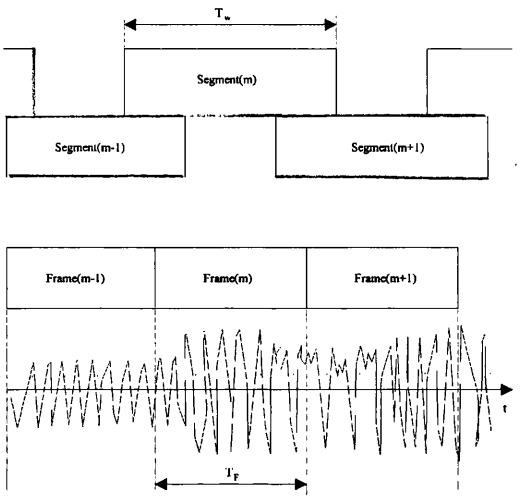


Figure 4.16: Framing and segmentation of the speech signal.

It can be summarised that the speech signal is divided into segments of duration T_w and the segments are passed to the spectral analysis stage with a period, the frame period, of T_F. T_w and T_F are further parameters of the speech recognition system.

4.6.6 Spectral Analysis

The previous steps have prepared the speech signal for spectral analysis. This is the real feature extraction part. The aim is to extract those parameters which are linguistically meaningful and able to distinguish sounds. Formally speaking, the spectral analysis is nothing more than a transformation:

$$\underline{s}(m) = T[s(m)] \tag{4.13}$$

where the sequence of frames s(m) is transformed into another sequence of $\underline{s}$(m) according to the operator T. $\underline{s}$ is usually a vector of parameters. There are many different types of transformations. The transformation is typically based on a model of the human auditory system or the human speech production system. A representative technique which takes the human auditory system and speech production system as an example is introduced in the following section. This technique is also one of the most popular analysis techniques currently used in speech recognition.

4.6.6.1 Cepstral Analysis Based on Fourier Transform Filter Bank

The basis for cepstral analysis is the speech production model according to Figure 4.17. An excitation signal produced by the air pressure from the lungs and controlled by the glottis (opening of the vocal cords) is filtered by a filter which models the vocal tract. The filtered excitation signal is the final speech signal. The filter parameters, which need not be static, correspond to the shape of the vocal tract and the place of articulation, i.e. nasal or oral. The excitation signal is determined by the glottis and the fundamental frequency. The fundamental frequency is a relatively fixed parameter for each individual speaker and basically controls the speaker's pitch. The glottis is responsible for causing the broad class of sounds, e.g. voiced (glottis only slightly opened) and unvoiced (glottis fully opened) sounds.

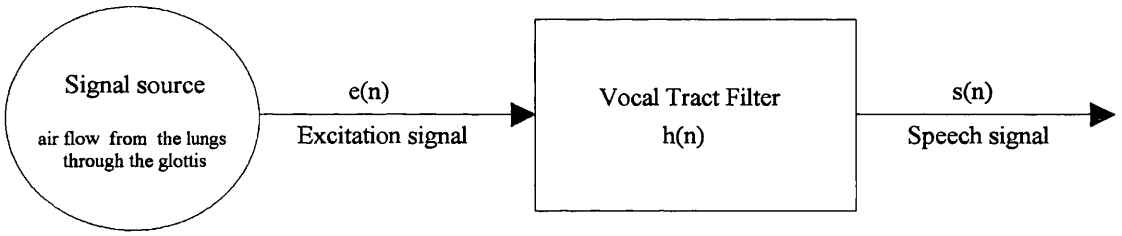


Figure 4.17: Speech production model as the basis for cepstral analysis.

The essential task is now to find the filter parameters and the excitation signal. This is the aim of the cepstral analysis. A filter can be described by its impulse response h(n) according to the Signal and System Theory. An impulse response is the output signal of a filter that is excited by the so-called "unit sample sequence", which is a signal with an amplitude of one at n=0 and an amplitude of zero otherwise. An important property of the impulse response is that it can be used to calculate the filter output signal for arbitrary input signals, or vice versa, according to,

$$s(n) = e(n) \otimes h(n) \tag{4.14}$$

where the operator $\otimes$ denotes convolution and leads to an integral to evaluate. However, another interesting property of the Signal and System Theory is the fact that Equation (4.14) can also be solved in the frequency domain simply by transforming the components of Equation (4.14) into the frequency domain. The relation between the time and frequency domain is for time discrete signals given by Equation (4.15),

$$X(f) = \sum_{n=-\infty}^{\infty} x(n)\ e^{-j2\pi fn}$$
$$x(n) = \int_{-\pi}^{\pi} X(f)\ e^{j2\pi fn}\ df \qquad \textbf{(4.15)}$$

which is the Discrete-Time Fourier Transform. Equation (4.15) applied to e(n) and h(n) then simplifies Equation (4.14) to,

$$S(f) = E(f) * H(f) \qquad \textbf{(4.16)}$$

where the capital letters denote the transformation into the frequency domain. So the convolution has been exchanged by a multiplication. H(f) is also known as the "transfer function" of the filter. The speech signal s(n) or, after the transformation into the frequency domain, the spectrum S(f) is what is actually available. If H(f) and E(f) can be separated then the model of Figure 4.17 is fully determined. Using logs yields:

$$\log(S(f)) = \log(E(f)) + \log(H(f)) \qquad \textbf{(4.17)}$$

The transformation into the frequency domain may be inverted, so that it can be transformed back into the time domain. But, instead of transforming S(f), the logarithms are transformed back to:

$$s`(n) = e`(n) + h`(n) \qquad \textbf{(4.18)}$$

The components are now no longer the same as the components in model of Figure 4.17, since log calculations have been used. The law of correspondence between the time and frequency transformations states that the superimposition remains, so that e`(n) and h`(n) correspond to E(f) and H(f). Thus, s`(n) is the superimposition of a representative of the excitation signal and a representative of the vocal tract filter.

Further filtering could be carried out for a total separation of the excitation signal and the vocal tract but this step is often omitted. s`(n) is called the cepstrum of s(n). Cepstrum is simply the word spec-trum changed since s`(n) looks like the spectrum of the vocal tract filter and the excitation signal, although, it is in the time domain and not the frequency domain.

The cepstral analysis can be carried out in several ways. One commonly used method is described in the following section. First, the speech signal s(m,n) is transformed into the frequency domain, which means that, for each speech segment m containing N samples, the spectrum is calculated by,

$$S(m,k) = \sum_{n=0}^{N-1} s(m,n)\, e^{\frac{-j2\pi nk}{N}} \qquad for\ k = 0,1 \ldots N-1 \tag{4.19}$$

Equation (4.19) is the Discrete Fourier Transform (DFT). It represents a segment of N speech samples by N frequency components in the frequency domain. Equation (4.19) requires N^2 complex operations. The well-known Fast Fourier Transform (FFT) is used to calculate the spectrum of a speech segment. The result of the FFT is exactly the same as that of the DFT but it requires only $N + N^2/2$ operations. The index k states that S(m,k) is a dimensionless sequence. However, without going deeper into DSP, it should be noted that the samples of S(m,k) are directly related to real frequencies according to,

$$f = k\,\frac{f_t}{N} \tag{4.20}$$

where f_t is the sample frequency. The DFT spectrum of a speech segment is often called the short term spectrum of speech because it spans only a frame. This spectrum could now be used directly to take the logs according to Equation (4.17) and then to transform it back into the time domain to receive the final cepstrum. However, two significant properties of the human auditory system should be considered at this point.

1) It is known that the perceived pitch of a tone is connected to its physical frequency. However, it has been shown that the relation between the physical frequency of a tone and the perceived pitch of a tone is non-linear. Steven and Volkmann (1940) have established experimentally a scale for the perceived pitch versus the physical frequency of tones. The scale transforms the physical frequency into a linear scale of perceived frequency (perceived pitch). The perceived pitch has the unit mel. Figure 4.18 shows the relation. For example, a tone with 1000 mel has a frequency of 1 kHz, and a tone with 2000 mel is perceived to be twice as high as a tone with 1000 mel although it has a frequency of 3 kHz.

The following approximation is used to transform the frequencies calculated by Equation (4.19) into mel frequency components (perceived pitch):

$$Mel(f) = 2595\ \log_{10}\left(1 + \frac{f}{700}\right) \tag{4.21}$$

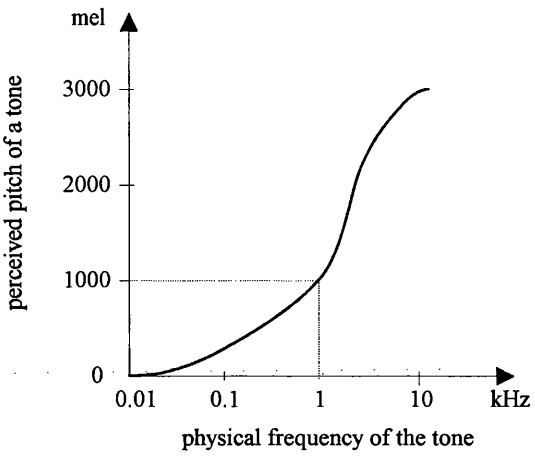


Figure 4.18: Relation between perceived pitch and physical frequency.

2) It has been shown experimentally that frequencies of a sound in a certain range cannot be individually distinguished. However, if one of the frequency components falls outside this range, it can be individually distinguished. This frequency range is called critical bandwidth. There are also established tables which give the critical bandwidth with respect to the so-called center frequency, which is the frequency component in the middle of the critical bandwidth. For example, a frequently-used table is based on the mel scale where the center frequencies are arranged according to the scale shown in Figure 4.18. Below 1 kHz the critical bandwidth is 100 Hz and above 1 kHz the bandwidth grows exponentially. It is non-overlapping.

Let us return to the frequency components delivered by the DFT of Equation (4.19). Both of above stated properties of the human auditory system can be applied in combination by weighting the spectrum with the filter function outlined in Figure 4.19. Each triangle realises a filter of a critical bandwidth. Each individual filter is laid out according to the mel perception scale.

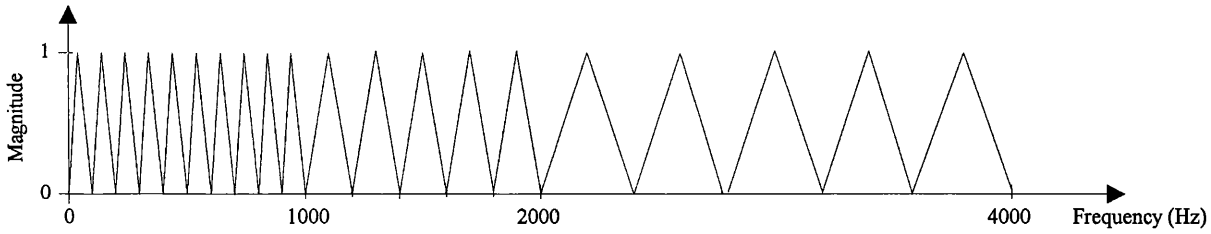


Figure 4.19: Filter bank of triangular critical bandwidth filters spaced along the mel scale, considering a speech bandwidth of 4000 Hz (sample frequency > 8000 Hz).

The final cepstrum signal is given by:

$$s`(m,n) = \frac{1}{N}\sum_{n=0}^{N-1} \log_{10}|S_w(m,k)|\, e^{\frac{-j2\pi nk}{N}} \qquad for\ n = 0,1...N-1$$

$$s`(m,n) = \frac{2}{N}\sum_{n=0}^{N-1} S_w(m,k)\cos(\frac{2\pi nk}{N}) \qquad for\ n = 0,1...N-1 \qquad \textbf{(4.22)}$$

where $S_w(m,k)$ is the weighted spectrum according to the applied filter function. s`(m,n) are referred to as the cepstrum coefficients. The second equation is a simplified version which is normally used in implementation. The cepstrum coefficients are the output of the speech signal analysis stage. They are used directly in the classification step, e.g., as parameters of the Hidden Markov Model.

Normally, additional analysis is carried out on the speech signal s(m), e.g., the energy calculation:

$$E(m) = \sum_{n=0}^{N-1} s(m,n)^2 \qquad \textbf{(4.23)}$$

The energy helps to distinguish between voiced (high energy) and unvoiced (low energy) sounds. The final output of the speech signal analysis is a set of different parameters, representing a speech frame m. These parameters are summed up conveniently in a vector, say $\underline{s}$, where the dimension is the number of parameters, N+1 parameters in the case of the cepstral analysis and energy analysis as stated above. This vector is calculated for each frame m. So the output of the speech signal analysis stage is the vector signal $\underline{s}(m)$. This signal will be referred to hereafter as the "speech features" or as "acoustic measurement".

4.7 The Hidden Markov Model

4.7.1 A Stochastic Model for Speech Recognition

The use of Hidden Markov Models in speech recognition is the result of some independent work of Jelinek and his colleagues at IBM [28][29] and Baker and his colleagues at Carnegie Mellon University [27]. As mentioned previously, the HMM can deal with the variability of the speech signal unlike the template-based approach. Furthermore, mathematical tools have been devised to make the application of the HMM feasible. A good introduction to Hidden Markov Models can be found in [33][34].

An HMM can be regarded as a kind of knowledge base for the variability of speech sounds, but it is not an inherent feature of the Hidden Markov Model. It can represent an infinite number of speech patterns of a particular linguistic unit, whereas in the template-based approach only a single template speech pattern is used for the representation of a linguistic unit. This statement explains the power of the HMM, considering that there is no unique template speech pattern for a given linguistic unit.

The model consists of states and transitions (see Figure 4.20). The circles are the states and the arrows are the state transitions. To each state is attached a probability distribution over speech features. This function models the distribution of the acoustic measurements of a speech sound and thus represents their variability. To each transition is attached a transition probability. This transition probability models the time flow of a speech sound and therefore represents its variability.

The Hidden Markov Model can serve as a probabilistic generator of sequences of speech features. The model can be traversed through several paths, and along each path a random sequence of speech features is generated. Mathematically speaking, the HMM can be considered as a random process. Analogously, the HMM can also serve as an acceptor of a given sequence of speech features. The given acoustic measurement must be aligned to each possible stochastic sequence. If several sequences are possible, the likelihood which can be calculated across the sequence can be used as the decision criterion for accepting the right sequence.

Figure 4.20 shows a model for a phoneme. The typical spectrum of a phoneme as a function of time can be divided into three subunits (beginning, middle, and end part) and is sketched at the bottom of Figure 4.20. Each part can be represented by the HMM stated above. An HMM which models a phoneme does not have a unique representation of a phoneme but rather a collection of possible variations. Therefore, an HMM is ideal for modelling allophonic variations of a phoneme. A phoneme HMM approximates the abstract unit phoneme in the same way as a human does, by allowing a certain range in modelling/articulating a phoneme. In the template-based approach only a single reference model is used to model a phoneme. However, remembering that a phoneme is an abstact ideal unit, a human cannot produce a complete unique phoneme. The phoneme HMM has been used only as an example. An HMM can also be used to model other linguistic units like

syllables, words or even whole sentences.

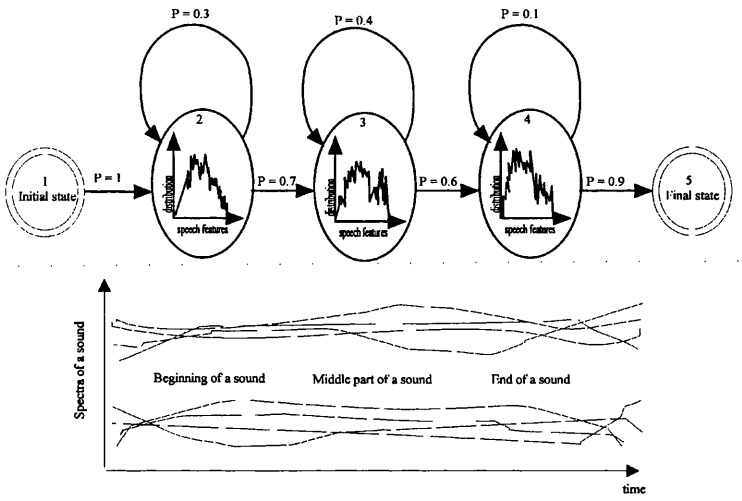


Figure 4.20: Hidden Markov Model for a phoneme.

A Hidden Markov Model is nothing other than a non-deterministic Finite State Machine (FST), where several stochastic transitions out of a state cause non-determinism. FSTs are well known in the computing science community, e.g., through [35]. Such machines can be described by the tuple,

$$M = (Q, \{a_{ij}\}, b_j) \qquad 1 \leq i, j \leq Q \tag{4.24}$$

where Q is the number of finite states, including the initial and final states. $\{a_{ij}\}$ is the transition matrix defining the transition probability from state i to j. Finally, b_j is the observation probability distribution of state j. The observation function can be realised as a discrete probability function or as a continuous probability density function. In the discrete case, the speech features are quantisised to a finite set by the so-called vector quantisation technique. b_j is then the probability distribution of the discrete speech features. In the continuous case, the speech features are not quantisised. Therefore, they are directly modelled by a continuous observation probability density function (pdf). In the continuous case $b_j(\underline{s})$ states the probability density of observing the speech feature $\underline{s}$ at state j. Only the continuous case will be pursued, so the discrete case will not be mentioned again, in the context of this thesis. A commonly used pdf is:

$$b_j(\underline{s}) = N(\underline{s} | \mu, \Sigma) \tag{4.25}$$

It is common to define an initial transition vector for the start transitions into the HMM. Another method is to introduce non-emitting initial and final states. This means that these states have no speech feature probability density function attached. This definition will be assumed hereafter, so that state 1 is an initial non-emitting state and state Q is a final non-emitting state. The matrix elements a_{1j} and a_{iQ} $(1 \leq i,j \leq Q)$ define the transitions out of the initial state and the final state respectively. This definition holds an advantage for the concatenation of HMMs because these non-emitting states serve as glue states.

The transition probability matrix and the number of states define the topology of the Hidden Markov Model. Basically, an HMM can have any form of a stochastic Finite State Machine and transitions from any state to any other state are possible. The number of states is also flexible. However, the HMM serves in speech recognition as a model of a linguistic unit. This is the reason why the HMM structure is selected according to the linguistic unit to be modelled. The transition matrix of the phoneme HMM of Figure 4.20 has the following form:

$$\{a_{ij}\} = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.3 & 0.7 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.4 & 0.6 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.1 & 0.9 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix} \quad \textbf{(4.26)}$$

This model is also called a left-to-right model because the allowed transitions are from left to right only.

The HMM has now been introduced but nothing has yet been said about how it can be applied to speech recognition, or how it gets the transition probabilities and the observation probability density functions. These topics will be addressed in the next sections. First, the factors which make HMM-based speech recognition possible will be examined.

4.7.2 The Basic HMM Equation

The acoustic measurements are the result of the speech signal analysis stage. The corresponding linguistic units are what is wanted. How are they related? Let $\underline{s}(m|w)$ denote the acoustic measurement of a spoken linguistic unit w, where w can be a phoneme or a word etc. However, w is unknown to the device which takes the acoustic measurement, so it actually measures $\underline{s}(m)$. The power of the HMM technique, with respect to SR, results from applying Bayes' rule:

$$P(w|\underline{s}(m)) = \frac{P(\underline{s}(m)|w)\ P(w)}{P(\underline{s}(m))} \qquad \textbf{(4.27)}$$

Considering at first only the left-hand side of Equation (4.27), which is the probability that w has been spoken when s has been measured. Recognition could now be performed by finding the value of w from a given vocabulary of size N which has most probably caused s:

$$\underset{i}{argmax}\{\ P(w_i|\underline{s}(m))\ \}\quad for\ i\ = 1,\ldots,N \qquad \textbf{(4.28)}$$

A vocabulary could be, for example, a phonemic alphabet where w would denote a phoneme, or a vocabulary of words where w would denote a word.

However, to evaluate Equation (4.28), the left-hand side of Equation (4.27) must be given for all w over the given vocabulary. This is not directly possible since it is a posterior probability, because only the acoustic measurement is available and that is the latter event. The right-hand side of Equation (4.27) makes it possible to calculate this posterior probability. Therefore P(s(m)|w) is needed. This is the probability of measuring the speech feature s when the linguistic unit w has been spoken. Actually P(s(m)|w) is in Bayes' language a likelihood, which is a quantity related to the probability. It has already been stated above that a linguistic unit can be modelled by an HMM. Now an HMM can be used to approximate P(s(m)|w) by a supervised learning procedure:

$$P(\underline{s}(m)|w) \approx P(\underline{s}(m)|M) \qquad \textbf{(4.29)}$$

This supervised learning procedure needs, for each linguistic unit being modelled by an HMM, examples of training speech in the form of acoustic measurements. Depending on the recognition task, more or less data is required. For example, for speaker-independent models much more speech is required than for speaker-dependent models. Given a set of G speech examples of a linguistic unit w, the actual HMM for the linguistic unit w is trained according to:

$$M_{ML} = \underset{M}{argmax}\{\ P(\underline{s}_r(m)|M)\ \}\quad for\ r\ = 1,\ldots,G \qquad \textbf{(4.30)}$$

This is the standard Maximum Likelihood Criterion which states that the likelihood of the Hidden Markov Model M is maximised for the given input. Maximisation of the HMM likelihood is actually achieved by updating the HMM parameters.

Thus replacing P(s(m)|w) in Equation (4.27) by P(s(m)|M), where M means "HMM for linguistic unit w", solves a part of Equation (4.27). It is this approximation by HMMs which

makes speech recognition feasible. Consequently, Equation (4.27) can be rewritten as:

$$P(w|\underline{s}(m)) = \frac{P(\underline{s}(m)|M)\ P(w)}{P(\underline{s}(m))} \qquad \textbf{(4.31)}$$

P(w) states the prior probability of generating the linguistic unit. It is prior knowledge that can be added to influence Equation (4.27). For example, a statistic for the language in question over the occurrence of linguistic units could be used to estimated P(w). This probability reflects knowledge about the language and therefore it is common to say that P(w) represents the Language Model. Analogously, it is common to say that P(s(m)|w) represents the Acoustic Model. P(s(m)) need not be regarded in evaluating the left-hand side of Equation (4.27), because it plays no role in the final decision rule. It can be regarded as normalising the probability for the allowed range. Since P(w) is independent of s(m) and P(s(m)) is a constant, recognition can be performed by using the corresponding HMM:

$$\underset{i}{argmax}\{\ P(w_i|\underline{s}(m))\ \} = \underset{i}{argmax}\{\ P(\underline{s}(m)|M_i)\ P(w_i)\} \quad for\ i = 1,\ldots,N \qquad \textbf{(5.1)}$$

Summarising Bayes' rule: The estimated likelihood approximates the probability that a certain linguistic unit has been produced by a certain acoustic measurement. This likelihood can be modelled by HMMs. However, what is actually needed is the posterior probability that a certain linguistic unit (formed in a human's mind) has produced a certain acoustic measurement (articulated by human's vocal tract). Fortunately, this conversion is provided by Bayes' rule. Bayes' rule involves an additional property for speech recognition. This is how the two components P(s(m)|w) and P(w) can be interpreted. P(s(m)|w), which is provided by the HMM, can be regarded as the Acoustic Model, because it is concerned with the measurement and interpretation of the speech signal. This component can be compared with the human ear to some extent. P(w) is called the Language Model because it provides space for incorporating human knowledge about language. Language knowledge can be categorised into morphology, syntax, semantics, and pragmatics. The Language Model represents a human's cognitive structure.

It could be argued that there is no need for a Language Model. This would be true if the likelihood P(s(m)|w) were strong but this is in general not the case, depending on the linguistic unit which will be modelled by the HMM. P(s(m)|w) is a probability distribution which states the uncertainty or weakness of Acoustic Models. A sophisticated Language Model as support for the Acoustic Model is essential for complex recognition tasks.

The Language Model must play a major role in speech recognition taking into consideration the language knowledge that we have in our minds, and that we are able to understand very noisy, degraded speech. The Acoustic Model is merely the trigger for the Language Model.

4.7.3 Speech Recognition with Hidden Markov Models

Bayes' rule has shown that recognition is feasible if the likelihood P(s(m)|w) is given for each unit of the vocabulary in question, assuming that the probabilities are available. For recognition, an algorithm is required which actually calculates P(s(m)|w) out of the HMM. The maximum likelihood decision rule of Equation (5.1) can then be used to decide which HMM is the match.

Basically the acoustic measurement, which is the signal s(m), must be aligned to all possible state sequences of an HMM. The probability is calculated along all allowed paths. Valid paths are such ones which start from an initial state and end at a final state and which have the exact length of the acoustic measurement s(m). The marked region of Figure 4.21 shows all possible paths of the HMM of Figure 4.20 for an acoustic measurement s(m) consisting of 9 speech frames.

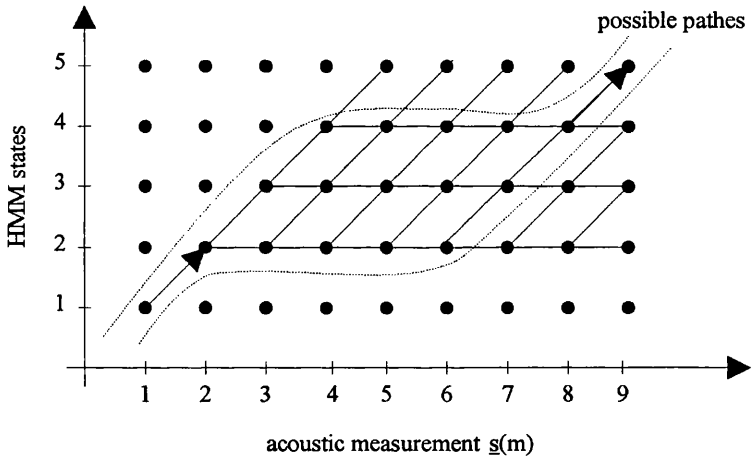


Figure 4.21: Alignment of acoustic measurements with HMM state sequences.

Assuming that only one path q is valid, then P(s(m)|w) = P(s(m), q|w). This probability can be calculated by taking the product of all transition probabilities a_{ij} multiplied with the speech feature observation probabilities $b_j(s)$ along the path, assuming statistical independence, which is stated by the first order Markov assumption. In general, however, several paths are valid. Therefore,

$$P(\underline{s}(m)|M) = \sum_{all\ valid\ q} P(\underline{s}(m), q|M) \qquad \textbf{(4.32)}$$

where the sum is justified because the different paths are mutually exclusive events. The calculation of Equation (4.32) requires $O(2NQ^N)$ floating point operations, where N is the length of the acoustic measurement signal s(m) in frames and Q is the number of HMM

states. For example, the duration of a phoneme is about 0.4s, so working with a frame period of 10ms, there are already 40 frames. And if a 5-state HMM, like that in Figure 4.20, is used there are then $80*5^{40} \approx 7*10^{29}$ floating point operations required. This amount is infeasible.

Luckily, there are two algorithms which allow a relatively efficient calculation of P(s(m)|M). The first algorithm was introduced by Baum [36] and is called the Forward-Backward algorithm. This algorithm provides the same magnitude as Equation (4.32) but requires only $O(Q^2N)$ operations. Another efficient algorithm comes from Coding Theory and is called the Viterbi algorithm [37]. It is slightly more efficient than the Forward-Backward algorithm. However, the quantity is different from Equation (4.32) because only the probability along the best path, rather than any valid path, is considered. A good introduction to the Viterbi Algorithm is given in [38]. The Viterbi algorithm is generally used as the HMM recognition algorithm because of its simplicity and so will be explained in more detail. The Forward-Backward algorithm is also used as part of the training procedure and is therefore explained in the next section.

The Viterbi algorithm is a breadth-first algorithm which means that all hypotheses (possible paths) are pursued in parallel. This class of algorithms works according to Bellman's optimality principle, which states that if several hypotheses arrive at a local decision point (an HMM state) only the best will survive (the path with the highest probability). Let $\Phi_j(k)$ denote the maximum likelihood of observing the speech features s(1) to s(k) and being in HMM state j at speech frame k. This likelihood can be calculated by the recursion:

$$\Phi_j(k) = \max_{1 \le i \le Q} [\Phi_i(k-1) a_{ij}] \; b_j\{\underline{s}(k)\} \quad for \; 1<k \le N, \; 1<j<Q \tag{4.33}$$

Initialisation is given by:

$$\begin{aligned} \Phi_j(1) &= a_{1j} b_j\{\underline{s}(1)\} \quad for \; 1<j<Q \\ \Phi_1(1) &= 1 \end{aligned} \tag{4.34}$$

Note that state 1 and state Q are non-emitting states defined in section 4.7.1. The algorithm terminates when the full length N of the speech signal has been aligned, where the paths (hypotheses) end in a final state:

$$\Phi_Q(N) = \max_{1 \le i \le Q} [\Phi_i(N) a_{iQ}] \tag{4.35}$$

This is the likelihood which is used in the decision rule Equation (5.1) for P(s(m)|M). Equation (4.33) states Bellman's principle and provides thus not exactly the same quantity given by Equation (4.32). The maximum likelihood paths, ending at all states i at speech frame k-1 (remember k is actually a time unit), are extended to all possible successor states j

by aligning the next speech frame k, but only the most likely path (max operator) is pursued in the case of several transitions into state j. The maximum likelihood path in state j is updated with the probability of observing the speech feature $\underline{s}(k)$ in state j. However, the max operator should not lead to the conclusion that at a certain speech frame k only one valid path exists. This is not the case. There are possible hypotheses for "each" intermediate state j for which the speech features $\underline{s}(1)$ to $\underline{s}(k)$ could have been aligned. This suggests the pruning of unlikely paths to save computing power. In practice, the logarithm from Equation (4.33) is used to avoid numeric underflow situations.

4.7.4 Training of Hidden Markov Models

So far, the initialisation of the HMMs has not been explained. There remains the question of how the states get their probability distributions and how the transitions get their stochastic behavior. This task is called HMM training. Once the HMM structure has been defined, i.e. the number of states and the valid transitions, training can be performed. The training algorithm works as the Maximum Likelihood Estimator according to Equation (4.30). Basically, the solution of,

$$\frac{\partial}{\partial M} P(\underline{s}(m)|M) = 0 \qquad \textbf{(4.36)}$$

is required, where M stands for the parameters of the HMM. However, there is no direct solution for Equation (4.36). Baum [36] and his colleagues devised an algorithm which solves Equation (4.36) not directly, however it provides certain stationary point solutions. This algorithm is called the Baum-Welch reestimation algorithm. It is introduced in the following section. First, some terms must be defined.

Let $\alpha_j(k)$ be the joint probability of observing speech features $\underline{s}(1)$ to $\underline{s}(k)$ and being in state j at speech frame k:

$$\alpha_j(k) = P(\underline{s}(1), \ldots, \underline{s}(k), q(k) = j|M) \qquad \textbf{(4.37)}$$

where q(k) = j means the path of the sequence of speech features 1 to k, starting at the initial state and ending at state j. This probability can be calculated by the recursion:

$$\alpha_j(k) = \left[\sum_{i=2}^{Q-1} \alpha_i(k-1)\, a_{ij}\right] b_j\{\underline{s}(k)\} \quad for\ 1<k\leq N,\ 1<j<Q \qquad \textbf{(4.38)}$$

The quantity $\alpha_j(k)$ is called the forward probability, because Equation (4.38) calculates the probability by moving forward in the sequence. Initialisation is given by:

$$\begin{aligned} \alpha_j(1) &= a_{1j}\, b_j\{\underline{s}(1)\} \quad for\ 1<j<Q \\ \alpha_1(1) &= 1 \end{aligned} \tag{4.39}$$

The algorithm terminates when the full length N of the acoustic measurement has been aligned, so that the paths (hypotheses) end in a final state:

$$\alpha_Q(N) = \sum_{i=2}^{Q-1} \{\alpha_i(N) a_{iQ}\} \tag{4.40}$$

The comparison of the Equations (4.38)-(4.40) with the Equations (4.33)-(4.35) of the Viterbi algorithm shows that the only difference is the max and the Σ operator. Actually, the forward probability of Equation (4.40) corresponds to the total likelihood of Equation (4.32). The result is:

$$\alpha_Q(N) = P(\underline{s}(m)|M) \tag{4.41}$$

The conclusion is that the forward probability of Equation (4.41) can also be used in the decision rule of Equation (5.1). Therefore the Equations (4.38)-(4.40) state the second alternative recognition algorithm. From the Probability Theory point of view, the forward probability is more appealing, the reason being that there are several paths through the HMM but there is only one sequence of speech features given which can be aligned (at least in classical physics). Several paths, only one of which can be chosen, means, in terms of Probability Theory, mutually exclusive events which in turn means adding the probabilities of the events. Summing the probabilities of all possible paths is more appealing than pursuing only the most likely path by the Viterbi algorithm. Nevertheless, it has been shown that the Viterbi algorithm works in practice anyway, and is in fact even faster than forward probability calculation.

Analogously to the forward probability, a backward probability $\beta_j(k)$ can be calculated by working backwards beginning at the end of the sequence. It is the probability of observing the speech features $\underline{s}(k+1)$ to $\underline{s}(N)$, given state j at speech frame k,

$$\beta_j(k) = P(\underline{s}(k+1),\ldots,\underline{s}(N)|q(k)=j,M) \tag{4.42}$$

where q(k) = j means the path of the sequence of speech features from k to N, starting from state j and ending at the final state Q. This probability can be calculated by the recursion:

$$\beta_i(k) = \sum_{j=2}^{Q-1} [a_{ij}\ b_j\{\underline{s}(k+1)\}\ \beta_j(k+1)] \quad for\ 1 \le k < N,\ 1 < j < Q \tag{4.43}$$

Initialisation is given by:

$$\beta_i(N) = a_{iQ} \quad for\ 1 < i < Q \tag{4.44}$$

The algorithm terminates when the full length N of the speech signal has been aligned, when the paths (hypotheses) end in an initial state:

$$\beta_1(1) = \sum_{j=2}^{Q-1} a_{1j} b_j\{\underline{s}(1)\}\beta_j(1) \tag{4.45}$$

The backward probability given by Equation (4.45) also corresponds to the total likelihood of Equation (4.32). So:

$$\beta_1(1) = P(\underline{s}(1),\ldots,\underline{s}(N)|q(1) = 1, M) = P(\underline{s}(m)|M) \tag{4.46}$$

Thus, the backward algorithm can also be used for recognition.

Finally, estimates are needed for the transition probabilities a_{ij} and the observation probability density function $b_j(\underline{s})$. From the nature of Probability Theory, it seems plausible that the estimates,

$$a_{ij}' = \frac{E(transitions\ from\ state\ i\ to\ state\ j)}{E(transitions\ from\ state\ i)} = \frac{\xi_{ij}}{\nu_i} \tag{4.47}$$

$$b_j'(\underline{s}) = \frac{E(times\ in\ state\ j\ and\ observing\ speech\ feature\ \underline{s})}{E(times\ in\ state\ j)}$$

can be assigned, where E(...) means "expected number of". Given some training data, the question arises of how this frequency information can be counted. This information cannot be observed directly because of the "Hidden" Markov Model but it can be obtained by the forward and backward probabilities:

$$\xi_{ij} = \sum_{k=1}^{N-1} \alpha_i(k)\, a_{ij}\, b_j\{\underline{s}(k+1)\}\beta_j(k+1) \quad for\ 1<i,j<Q$$

$$\nu_i = \sum_{k=1}^{N} \alpha_i(k)\beta_i(k) \quad for\ 1<i<Q \qquad \textbf{(4.48)}$$

Given a set of G training examples, the transition probabilities are estimated by:

$$a_{ij}' = \frac{\sum_{r=1}^{G} \xi_{ij}\{\underline{s}_r(m)\}}{\sum_{r=1}^{G} \nu_i\{\underline{s}_r(m)\}} \quad for\ 1<i,j<Q \qquad \textbf{(4.49)}$$

The estimates for the transitions from the non-emitting initial state are given by:

$$a_{1j}' = \frac{1}{G}\sum_{r=1}^{G} \frac{\nu_j\{\underline{s}_r(1)\}}{P(\underline{s}_r(m)|M)} \quad for\ 1<j<Q \qquad \textbf{(4.50)}$$

And the transitions into the non-emitting final state:

$$a_{iQ}' = \frac{\sum_{r=1}^{G} \nu_i\{\underline{s}_r(N)\}}{\sum_{r=1}^{G}\sum_{k=1}^{N} \nu_i\{\underline{s}_r(k)\}} \quad for\ 1<i<Q \qquad \textbf{(4.51)}$$

The estimates for the observation probability density function $b_j(\underline{s})$ are more sophisticated because $b_j(\underline{s})$ is, according to the definition of Equation (4.25), a Gaussian probability density function. The parameters of $b_j(\underline{s})$ which must be estimated are the mean μ and the covariance Σ. Note, μ is a vector with the same dimension as the speech feature vector $\underline{s}$. For each component of the speech feature vector there exists a separate mean. Correspondingly, Σ is a squared matrix. The quantity,

$$\zeta_j = \sum_{k=1}^{N} \alpha_j(k)\beta_j(k)\ \underline{s}(k) \quad for\ 1<j<Q \qquad \textbf{(4.52)}$$

gives information about the observation of speech feature $\underline{s}(k)$ in state j. Hence, given a set of G training examples, the mean of the probability density function of state j can be calculated by:

$$\mu_j' = \frac{\sum_{r=1}^{G} \zeta_j\{\underline{s}_r(m)\}}{\sum_{r=1}^{G} \nu_j\{\underline{s}_r(m)\}} \quad for\ 1<j<Q \tag{4.53}$$

Correspondingly, the quantity,

$$\vartheta_j = \sum_{k=1}^{N} \alpha_j(k)\beta_j(k)\ \{\underline{s}(k) - \mu_j\}\ \{\underline{s}(k) - \mu_j\}^T \quad for\ 1<j<Q \tag{4.54}$$

gives information about the variance of the speech feature $\underline{s}(k)$ in state j. Hence, given a set of G training examples, the covariance of the probability density function of state j can be calculated by:

$$\Sigma_j' = \frac{\sum_{r=1}^{G} \vartheta_j\{\underline{s}_r(m)\}}{\sum_{r=1}^{G} \nu_j\{\underline{s}_r(m)\}} \quad for\ 1<j<Q \tag{4.55}$$

The estimates for the transition probabilities denoted by a_{ij}' and the estimates for the probability density function denoted by $b_j'(\underline{s})$ can be calculated by the Equations (4.48)-(4.55). What has not been mentioned so far is the implicit assumption that the equations rely on the already initialised parameters a_{ij} and $b_j(\underline{s})$ because they are needed for the calculation of the forward and backward probabilities. This assumption constitutes the reestimation procedure. Given a Hidden Markov Model,

$$M = (Q, \{a_{ij}\}, b_j) \qquad 1 \le i, j \le Q \tag{4.56}$$

where the parameters have already been initialised in some way, this model M can be used to reestimate new parameters by the Equations (4.48)-(4.55). The result is a reestimated HMM with new parameters:

$$M' = (Q, \{a_{ij}'\}, b_j') \qquad 1 \leq i,j \leq Q \tag{4.57}$$

Baum and his colleagues have proven that the initial model M defines either a critical point of the likelihood P(s(m)|M), which means M = M', or the likelihood P(s(m)|M') of model M' is greater than the likelihood P(s(m)|M) of model M. Thus, if the HMM is initially not in a critical point then using M' in place of M maximises the likelihood P(s|M) iteratively, for a given set of training speech. Furthermore, Baum has shown that the iteration converges to a local maximum. So it can be terminated by using the convergence criterion:

$$P(s(m)|M') - P(s(m)|M) > \rho \tag{4.58}$$

where ρ is a small threshold. In general, a global maximum would be preferred; however, the algorithm leads in most cases only to a local maximum and cannot progress beyond that maximum since a local maximum can be regarded as a critical point. The probability of finding a global maximum is very small because the likelihood function P(s(m)|M) spans a high- dimensional space with many local maxima. The algorithm converges to the nearest maximum relative to the initial HMM parameters. Critical points and maxima of the likelihood P(s(m)|M) are indicated in Figure 4.22.

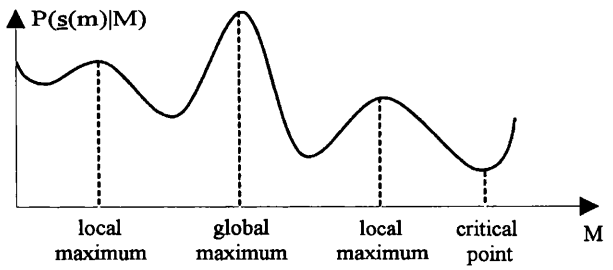


Figure 4.22: Local maxima and critical point problem of Baum-Welsch reestimation.

The critical point is not a particular problem. If a critical point is caused by the initial parameters, only the initial values have to be changed by adding a small amount of noise. The initial values of the parameters must come from somewhere. They can be guessed or assigned in some more intelligent way, for example, by an uniform segmentation procedure where some training speech examples are uniformly segmented and assigned to the states.

A major drawback of the training algorithm results from the Maximum Likelihood Estimation. It is argued [39] that HMMs suffer from poor discrimination power because of the assumption that they are trained to maximise P(s(m)|M) for the correct model rather than trained to minimise P(s(m)|M) for the incorrect models. Nevertheless, in reality it has been shown that the correct application of the Baum-Welsch reestimation procedure assigns useful likelihoods to Hidden Markov Models. It is obvious that the amount of training speech is an

essential factor for the usefulness of the HMM in speech recognition.

4.7.5 Speech Recognition based on Subword Units

In Equation (4.31), the HMM training and recognition algorithms were stated for training and recognition of single linguistic units, for example, phonemes or words. Recognition of single words is sensible; however, recognition of single phonemes is uninteresting. The purpose of phonemes is to model higher level linguistic units like words, phrases, sentences etc. The extension of the training and recognition algorithms to handle that is straightforward. It can be fully illustrated in Equation (4.31), with word recognition based on phonemes. Let $w_{w,x}$ denote a word which will be modelled. $w_{w,x}$ can be decomposed into a sequence of phonemes, say $w_{p,1}$, $w_{p,2}$, ..., $w_{p,u}$. For each phoneme there exists a trained HMM; let them be denoted by $M_{p,1}$, $M_{p,2}$, ..., $M_{p,u}$. The concatenation of the phoneme strings to form a word can also be regarded as an intersection of events in the sense of joint probabilities. Replacing w in Equation (4.31), the symbol for a single unit, by a sequence of linguistic units results in:

$$P(w_{p,1}, w_{p,2}, ..., w_{p,u} | \underline{s}(m)) = \frac{P(\underline{s}(m) | M_{p,1}, M_{p,2}, ..., M_{p,u})\ P(w_{p,1}, w_{p,2}, ..., w_{p,u})}{P(\underline{s}(m))} \quad \textbf{(4.59)}$$

The joint probabilities can be resolved:

$$P(w_{p,1}, w_{p,2}, ..., w_{p,u} | \underline{s}(m)) = \frac{P(s | M_{p,1}, M_{p,2}, ..., M_{p,u})}{P(\underline{s}(m))} \quad \textbf{(4.60)}$$

$$\times\ P(w_{p,1})\ P(w_{p,2} | w_{p,1})\ ...\ P(w_{p,u} | w_{p,1}, w_{p,2}, ..., w_{p,u-1})$$

The first term of Equation (4.60) shows how the individual phoneme HMMs are handled. They are treated simply as concatenated HMMs. The concatenation of HMMs is also an HMM. This becomes apparent by considering that the HMM is a Finite State Machine with initial and final states, which are the connector states. Thus the recognition algorithm is applied to the concatenated HMM in exactly the same way as to a single phoneme HMM. Concatenation of phoneme HMMs can also be exploited for the training task. It is possible for training speech, which is in form of spoken sentences, to build such concatenations of HMMs according to the phonemic transcription of the sentences. The concatenated HMMs are then trained on the complete sentence. There are two great advantages of this training method. The phoneme HMMs are trained in context, which means that coarticulation is considered, and the time boundaries of the individual phonemes of the training speech need

not to be known. Section 5.9 will explain this training method in more detail. Thus, speech recognition based on subword units can be handled with the same framework as that used for single unit recognition, no additional tools are needed.

Words modelled by phonemes have only been used as an example; it works in exactly the same way for every other linguistic unit which can be modelled by HMMs and which can be concatenated to higher level linguistic units. The higher level unit can be used to model still further levels of linguist units. Therefore, it is possible to model sentences with an intermediate unit word and a terminal unit phoneme.

The second term of Equation (4.60), the Language Model, also looks straightforward but estimating the conditional probabilities of Equation (4.60) is only feasible in a limited way by using constraints. In fact, finding reliable estimates for these conditionals is a major issue in large vocabulary recognition systems. As stated, the Language Model represents a priori knowledge about the language, for example, the probability that a certain word follows all previously recognised words. Such a probability can be obtained by using statistics over larger text corpora, but the greater the history being considered, the less training data for getting reliable estimates is available. This is the reason why typically only one, two or three previous words are considered in prediction of the expected word. If no previous words are considered, all terms behind the conditional stroke of the Language Model term disappear. This means independence between the linguist units. These kinds of constraints are typical in all types of systems based on Bayes' framework, for example the Binary Independence Retrieval Model in Information Retrieval, which is based on an independence assumption. Pearl noted the problem of representing what he calls probabilistic knowledge [40]:

> *"In a sparsely connected world like ours, it is fairly clear that probabilistic knowledge, in both man and machine, should not be represented as entries of a giant joint distribution table, but rather by a network of low order probabilistic relationships between small clusters of semantically related propositions".*

The application of language modelling will be explained in chapter 7.

4.8 Conclusion

This chapter has introduced speech recognition theory. Some practical examples of what makes speech recognition so difficult have been given. These problems direct speech recognition research. For example, the pure speech waveform, represented in the time domain, cannot be directly applied to speech recognition because it does not show reliable features which can be used for classification. Therefore, a feature extraction step must be applied to gain sensible speech features. The feature extraction step can also be regarded as the acoustic measurement device which outputs acoustic measurements of the speech signal. The stochastic character of the speech signal has led to the statistical and connectionist approaches. The statistical approach is based on Hidden Markov Models. Hidden Markov Modelling can be

regarded as a generalisation of the template-based approach which overcomes its weaknesses. It has been shown that the power and elegance of the statistical approach derives not only from the Hidden Markov Model but from the whole Bayes' framework in which the Hidden Markov Model is embedded. This framework provides additional useful properties for speech recognition. It splits speech recognition into acoustic modelling and language modelling. Furthermore, modelling of subword units shares with Bayes' framework and the Hidden Markov Model framework, the feature that statistical speech recognition can be generalised for each linguistic unit. Finally, sections 4.5 up to 4.7 explained the complete process of speech recognition in detail and emphasised the enormous parameter space which accumulates together through the different processes. It also shows the multidisciplinary character of speech recognition research, which spans the Signal and System Theory, Probability Theory and Linguistics. Speech recognition relies on the optimal application of all these areas. For example, the best Hidden Markov Models cannot correct fundamental errors in the acoustic measurement step.

Chapter 5

Design of the Acoustic Front-End

5.1 Introduction

The speech recognition techniques have been discussed in the previous chapter. To realise an acoustic front-end, those algorithms must be implemented. This is, in principle, not a problem, as many classic papers about these algorithms are available, but it would take several man-years to build a reliable and flexible product. In this chapter a practical design of the acoustic front-end is developed by using Hidden Markov Models and the toolkit HTK [13].

The phrase "acoustic front-end" will become clear if the choice of subword units has been introduced because the output of the acoustic front-end is not in word form but in the form of linguistic units below word level. The acoustic front-end can be regarded as the interface to the machine's cognition system. The phrase "machine's cognition system" is used to refer to what should follow the acoustic front-end.

5.2 Hidden Markov Model Toolkit

Fortunately, the basic HMM technology has already been implemented in the form of a toolkit called HTK [13]. It was developed by Entropic Research Laboratory, which is a commercial signal processing software supplier, in association with the Speech Group at Cambridge University Engineering Department.

HTK is a collection of HMM algorithms necessary for training and recognition. It includes the main HMM research which was carried out during the years 1989 to 1993. HTK provides C source code as well as executables of the algorithms in Unix command line style. It runs on 32 bit machines. Table 5.1 presents an overview of the most important HTK tools.

Name	Function
HCode	Speech signal analysis, spectral feature extraction
HERest	Baum-Welsch reestimation, training algorithm for embedded model training
HHEd	Batch mode HMM editor for model refinement
HInit	Algorithm for HMM initialisation
HRest	Baum-Welsch reestimation for single model training
HResults	Evaluation of recognition result
HLEd	Script-based stream editor for editing label files
HVite	Viterbi recognition algorithm

Table 5.1: Summary of the most important HTK tools.

Before HTK and the acoustic front-end are explained in more detail, it is important to note the following. Once you have decided to use the HMM technique for speech recognition, the HTK toolkit will save you a lot of work. As discussed earlier, a number of decisions need to be made. For example, the fundamental configuration of the HMMs themselves and the training and recognition strategies are not specified by the toolkit at all. Instead, HTK gives you enormous freedom to implement your own strategy. The choice of strategy and decisions will be discussed in the following sections.

5.3 Choice of Subword Unit

Speech sounds have to be mapped to lexical or phonological representations in some way. This is the task of the HMM, as mentioned before. It maps a segment of a speech waveform to a corresponding linguistic unit, e.g., to a phoneme, syllable, word or sentence. The HMM acts as a stochastic mapping function. For each linguistic unit a special mapping function, a special HMM, is needed. Thus, an HMM represents an instance of a linguistic unit. The main question is

which linguistic unit should be used?

This question remains unanswered [34]. Even in the recent study [20] which confirms the feasibility of the big SR project verbmobil, there is no evidence that the proper speech unit has been found:

"Despite the great amount of work that has been done on speech recognition, especially over the past decade, we believe that it is still far from clear what the size of the segment should be that a system should aim to recognize".

Some of the following general points help to restrict the choice.

1. Trainability of Linguistic Units

The HMM must be trained from existing speech. For each instance of a linguistic unit, a large amount of corresponding speech must be provided. For example, assuming only a medium-sized vocabulary of 1000 words, about 10 examples per word and speaker, with a total of about 500 different speakers, are required for training for speaker-independent recognition. It is almost impossible to record 5,000,000 spoken words for economic reasons. In addition, each word (including also derivatives and compounds of lexical words) must be modelled by a HMM, unlike IR, where only the word stems are used. A study [41] of the size of a human's mental lexicon, using college students, has shown that the average college student knows 58,000 lexical words (dictionary entries) and 96,000 derivatives and compounds.

The conclusion from this is that the training problem restricts the choice to a unit smaller than a word because it is impossible to provide enough training material for each word. However, the trainability problem should not lead to the opinion that words are, in general, an improper linguistic unit. Small vocabulary applications, using words as the basic linguistics unit, work reliably and have high recognition rates.

2. Language Universality of Linguistic Units

In order not to be restricted to a certain vocabulary, the linguistic unit must have a universal language character. A linguistic unit has a universal language character when new words can be created by composing linguistic units according to some universal language rules. For example, in written language, the normal letters are universal language units because new words can be created simply by concatenating the letters according to some rule. In speech, the counterparts to the letters of the written language are the phonemes.

In English, there are only about 50 of them. Words can be built simply by concatenating the phonemes according to a phonemic lexicon. A large number of training examples can also be provided. For example, 100 average sentences contain about 4000 phonemes. However, their disadvantage is that they are highly sensitive to the coarticulation effect, see section 4.3.2. This means that adjacent phonemes influence each another. Actually, for each phoneme there exists a set of different variations dependent on the context in which the phonemes occur. However, this context dependency can be modelled to some extent by the individual phoneme HMM.

The syllable also has a universal language character according to [42]. The number of syllables in English is about 4000, which is on the limit of trainability. However, there is currently no suitable speech database available which provides syllabified speech for training.

In addition to the recorded speech, the speech sounds must be labeled with the corresponding syllable description and with exact time boundaries of the syllables. This fact rules out the use of speech databases designed for other linguistic units.

It is an interesting question to which representation the speech sounds are mapped in a human's perception. Intuitively, it would be phonemes but Cognitive Scientists [43] argue:

> *"Speech is more than just a sequence of phonemes. There must be larger Gestalt-like units that come into play either at the decision level or at an earlier processing state."*

For example, they [43] desribe some experiments which show that syllables are more quickly perceived/detected as phonemes. Therefore it can be supposed that the syllable is the first basic linguistic unit onto which the sounds are mapped. The phonemes are then extracted from the syllable.

To demonstrate the role of syllables in human perception, you can do a simple experiment: try to segment the word "uncertainty", firstly in phonemes without regard to a particular phonemic alphabet and just segmenting, secondly in syllables. Most people will find it difficult to segment the word "uncertainty" into phonemes but they will have less trouble in determining the syllables of the word "uncertainty". The fact that the syllabification task is easier for us certainly arises not only from knowledge of hyphenation rules but also from the fact that we speak in segments of syllables.

The properties of the syllable as the basic linguistic unit in speech perception will be taken into consideration and therefore a way of exploiting syllable characteristics will be kept in mind. However, the main decision has been made to use

"phonemes as the basic linguistic unit"

which will be modelled by the HMM because of their flexibility and their trainability, although it is known that the phoneme is an ideal unit rather than a practical one. Furthermore, the HMM also has its weakness which arises from the underlying first-order Markov assumption, i.e., that successive speech sounds are treated as statistically independent. Therefore, some phoneme recognition errors must be expected. These errors cause a noisy phonemic transcription of the spoken query. As stated in chapter 2, a query can be regarded as a sequence of loosely coupled words, which are not necessarily syntactically related to each another. Therefore, the goal is to extract words from the noisy phoneme sequence. Being aware that the phoneme sequence is noisy, the question is

how much noise is acceptable?

so that words can still be extracted. This question must be investigated more thoroughly before proceeding with the acoustic front-end, in order to provide a basis for the decision to use phonemes or experiment with phonemes.

To address this problem, the recognition task is regarded from the point of Coding Theory. In general, if a message without redundancy is transmitted, possible errors in the message

cannot be corrected. However, if a redundant message is transmitted, possible errors could be corrected or at least detected.

Languages are very redundant. According to Shannon [44], written English has a redundancy of approximately 50%. This means that 50% of a message is enough to decode the content of the whole message. It is plausible that this is also true, to some extent, for a phonemic described language. However, what is redundancy of language? It is human knowledge about language, which can be subdivided into knowledge about syntax, semantics and pragmatics. Thus, errors could be reconstructed by using human knowledge about language.

A simple experiment was carried out which might give a sense of the redundancy which results from the syntax of a language. In a 100,000 word phonemic described lexicon, all words with 6 phonemes were picked out. Six phonemes correspond to the average length of words. Then, in every word, each phoneme was substituted by another phoneme successively. The consequence was that words became equal. The diagram in Figure 5.1 shows how many words became equal for each phoneme pair. The maximum was the replacement of the phoneme "IH" by "AH" (see ARPAbet Appendix A), which produced 500 equal words. The average was 11 equal words. In most cases of replacement, surprisingly no equal words were produced, see diagram. This good result of relatively few words becoming equal is due to the enormous gap between possible words with 6 phonemes (6^{48} words) and the real English words (about 20,000 words).

The question is not whether we should use human knowledge, but how we can apply it. The acoustic front-end, which maps the physical representation of speech to linguistic units, is the interface to the machine's cognition system. Further steps to reach the final word stage must be based upon the chosen linguistic units. The choice of the linguistic units is also reflected in the following further major decisions about the acoustic front-end.

Finally, it must be stated that it may seem self-evident that phonemes should be used, but this is not the case. Phonemes are certainly a good choice considering the overall project goal. There will be problems involved, however, with regard to their atomic character and the expected high error rate. Furthermore, such an approach requires the use of multidisciplinary resources to cope with these problems. For a short-term approach with a limited vocabulary, say 500 words, the word as the basic linguistic unit would be the better choice: the expected error rate would be small and words can be used directly as output of the speech recognition system. However, the word training problem which was mentioned previously must be taken seriously. The vocabulary size could not be expanded. A 500 word speech recogniser is not appropriate for a text-based IR system. In general, IR is used for information-seeking in large information spaces, and, therefore, in a text document space the vocabulary is much higher than 500 words. For example, in the IR tool NRT, which is a retrieval tool for retrieving news from a newspaper collection, the index vocabulary is more than 100,000 words.

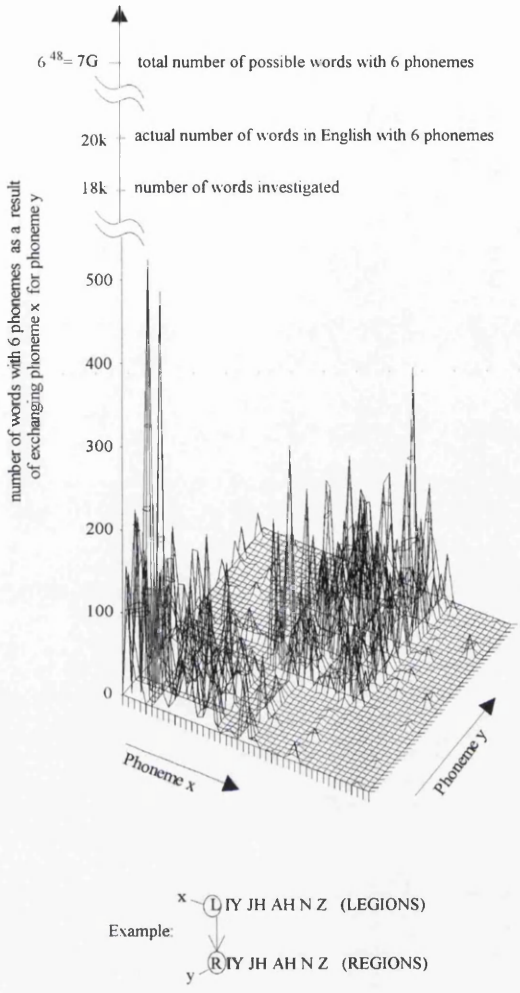


Figure 5.1: Redundancy or gap between what is possible and what is actually used of the phonemic codes, in the context of the English language.

5.4 Choice of Phonemic Alphabet

Phonemes are organised in alphabets like the counterpart of the written language. However, there remains the question of which phonemic alphabet should be used. The alphabets of phonemes for English sounds used in SR are based on the International Phonetic Alphabet (IPA). They differ mainly in the total number of phonemes. The bigger the alphabet is, the more precise the description of spoken English. It is also plausible that a bigger alphabet causes more recognition errors. Conversely, a smaller alphabet causes less recognition errors, but the description of spoken English is more general. This generality introduces additional ambiguity in decoding the words. Hence, it is a trade-off between specificity and generality.

The speech database necessary for HMM training limits the choice. As mentioned, the speech must be labeled with phonemes. Therefore, the phonemic alphabet must be derivable from that used in the speech database. I have decided to go along with Lee & Hon [45]. Their set is based on 48 phonemes; however, they finally used a set with just 39 phonemes by merging similar phonemes to conform to the CMU/MIT standard. Lee & Hon reported that the merging of the phonemes {el,l}, {en,n}, {sh,zh}, {ao,aa}, {ih,ix} and {ah,ax} led to only small improvements, but the merging of the closures was necessary for good performance. To get a more precise description, the above mentioned merging which caused only small improvements, has not been taken oven. In TIMIT, eight closures may be distinguished. These eight closures have been merged into the three closure groups "cl, epi, vcl". Lee & Hon merged all closures to the silence group. The silence symbol is not a phoneme in the correct sense but plays an important role in modeling silence. It will also be used and, in the following, it will be treated like a phoneme. It can also be debated whether closures are phonemes or phonetic symbols denoting allophonic variants, but the three closure groups used are also just treated as phonemes. Hence, the phonemic alphabet used in this approach consists of 48 phonemes. This alphabet and that used by Lee & Hon are based on the ARPAbet which has been established as the overall standard for speech research in the United States. It is also similar to the IPA. Appendix A describes these phonemic alphabets and presents conversion tables. The specified set of 48 phonemes used here is denoted below as the p48 set.

5.5 Choice of Hidden Markov Model

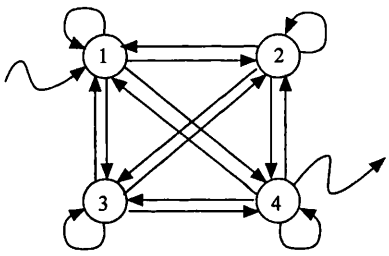
As mentioned in section 4.7, an HMM can take the form of any stochastic Finite State Machine. What sort of HMM can be realised by HTK? Basically any Finite State Machine can be realised. The observation probability function $b_j(\underline{s})$ is limited to a Gaussian probability density function (pdf), but this pdf itself can be adjusted in several ways. Thus, the speech features are modelled by a continuous pdf. The number of states and the way the states are connected by the transitions determine the structure of the HMM. In HTK any structure can be realised. In general, there is no special recommendation as to what the best structure is. Some reference work can be used and experiments are required. Quoting Rabiner et. al. [46]:

> *"There appears to be no good theoretical way to choose the number of states needed for a word model, since the states need not be physically related to any single observable phenomenon."*

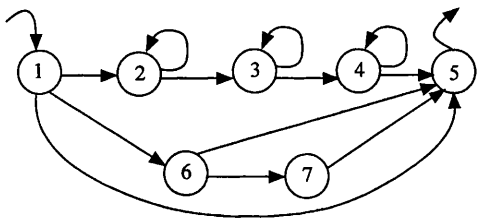
The transitions, which are probabilities, and the output probability density function must be estimated by the training stage. It is plausible that the more transitions, and states, and the more complex the output probability function, the more training data is required to satisfy the parameter space. On the other hand, an HMM describes a linguistic unit, in our case a phoneme. Therefore, it also seems plausible that the more sophisticated and specific an HMM,

the better is the approximation of the HMM to the corresponding linguistic unit. There is a trade-off between trainability and specificity. Training data, as well as the computing time to estimate the HMM parameters, will certainly be a limited resource. Roughly speaking, the number of free HMM parameters which have to be estimated (trained) increases by the order square in relation to the number of transition states. Therefore, complex structures should be avoided.

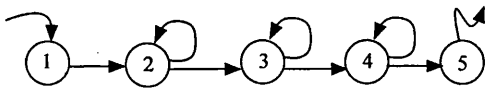
An HMM for a phoneme can be designed according to the Figure 5.2.(a), or like (b) which is used in the SPHINX system [6], or like (c). Experimental evidence suggests that states represent identifiable acoustic phenomena. Therefore, the number of states is often chosen to correspond roughly to the expected number of acoustic phenomena in the speech unit (phoneme). As stated in Figure 4.20, a phoneme can be roughly divided into three subunits: the beginning, middle and end parts. These subunits are ordered according to their time. Using these subunits as acoustic phenomenas, a structure with 3 states, connected from left to right, can be assigned to an HMM.



(a) Ergodic HMM with 5 states



(b) Phoneme HMM used in the SPHINX system



(c) Phoneme HMM used in the Spoken Query Recogniz

Figure 5.2: HMM structures.

Five alternative structures have been used for experimentation. The recognition results should help to decide which of them will be used in the future. The tested HMM structures are sketched in Figure 5.3. The time and space complexity increases from model 1 to model 5. The left-to-right character of the HMMs seems to be a reasonable choice, considering the time flow of phonemes. Models 2-5 allow skipping of intermediate states that is why these models model better the variance of phoneme duration. The two states 1 and 5 are special

states which have no attached output probability density function. These states serve to connect several HMMs to an HMM network. They can be regarded as glue states.

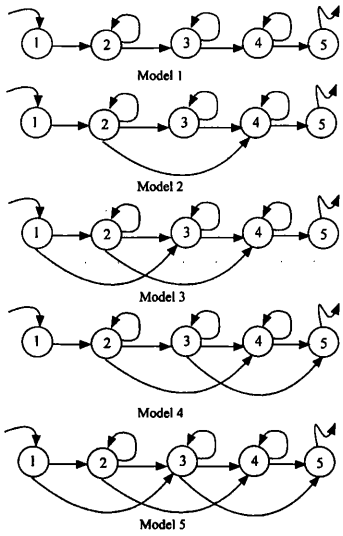


Figure 5.3: HMM structures used in this approach.

Intuitively, one might expect that the more complex models represent the phoneme better because there are more skip transitions which might reflect the different durations of the individual phonemes in a better way. Table 5.2 shows the evaluation of the five models, using the TIMIT database (see section 5.8), by recognition rate. The recognition rate is defined as the ratio between the number of correct recognised phonemes and the total number of phonemes occurring in the test set. The relative recognition rate is the ratio between the recognition rate of model x and the recognition rate of model 1 scaled by 100%.

Model number	1	2	3	4	5
Relative recognition rate in %	100	99.49	99.90	99.68	97.35

Table 5.2: Relative recognition rate of different phoneme HMM structures.

Surprisingly, the simplest model seems to be the best. Even if the individual figures could not be taken so seriously, there is at least no clear discrepancy which would favour a particular model. As previously mentioned, the time and space complexity of model 1 is the lowest, hence it will be used for further work. However, this evaluation holds only for this training set. As stated above, the availability of more training data allows more sophisticated HMMs and these results may not hold for other training sets.

5.6 Speech Signal Analysis Specification

Although the speech signal analysis can be carried out using the HTK tool HCode, the type of speech signal analysis is not fixed. HCode offers the most common speech signal analysis techniques, allowing a total of 54 different techniques. It is also possible to bypass HCode and use other techniques. The specification of the speech signal analysis technique for this approach will be addressed in this section and will be informed by reference to the literature and the results of some experiments.

The Sample Frequency

As mentioned earlier, the sample frequency is one of the basic parameters of an SR system. Experimental equipment that allows experiments with variable sample frequencies would be ideal. However, such equipment is not available, and therefore only general purpose equipment can be used for carrying out the experiments. The audio interface of a Sun Sparc 10 offers the sample frequencies of 8 kHz, 16 kHz, 32 kHz and 44.2 kHz. The frequency to be used for the experiments must now be decided. This decision has to be made in advance in order to be able to develop the audio interface. The important frequency range of the speech signal must be taken into account. The points noted in section 4.6.2 should help to make the decision. For speech recognition an important frequency range is which (1) distinguishes sounds, and which (2) is perceived by the human. It is obvious that (1) is contained in (2). The following points should help to indicate the right sample frequency.

For example, telephony uses a bandwidth of 300 Hz to 3.4 kHz which is internationally accepted. This range was established by measuring the quality of syllable understanding. Figure 5.4 shows the contribution of the frequency components of a speech signal to syllable comprehension, and demonstrates that the range between 500 Hz and 4 kHz is important. Everybody who is familiar with the quality of telephone speech knows that the quality is limited but it is still good enough for understanding. This bandwidth also denotes the minimum for speech recognition although the higher, the better. Thus, the sample frequency should be at least 8 kHz, twice as much as the bandwidth in accordance with Nyquist's sample theorem.

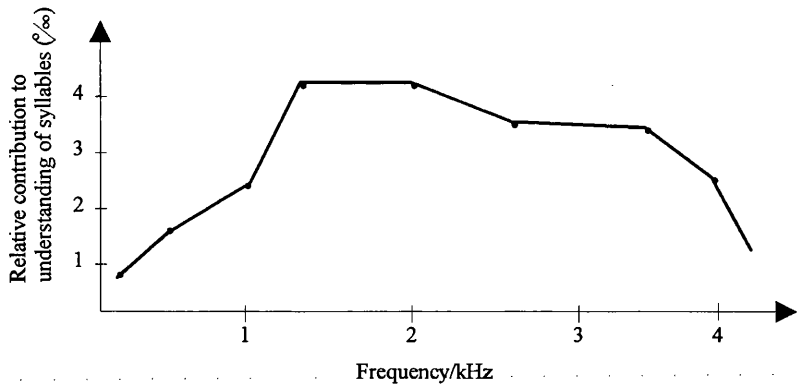


Figure 5.4: The contribution of the frequency components of a speech signal to the understanding of syllables.

The fundamental frequency, which is the major frequency component responsible for differentiating speakers, is far below 300 Hz (50-250 Hz for men and 120-500 Hz for women). However, in telephony it is still possible to differentiate between speakers, even if this frequency is not actually present. This is a phenomenon of the human auditory system, as the perceived frequency, the pitch, is different from the frequency components of the signal. The formant frequencies, which characterise vowels, are in the range of 500 Hz to 3500 Hz. They can be quite well recognised in the spectrum of the signal. Consonants have no such marked characteristic frequencies. Their range is unsharp and is similar to a noise spectrum and their frequencies range up to about 8 kHz.

Therefore, a sample frequency of 8 kHz is adequate for telephony, and it might be concluded that it is also adequate for speech recognition. However, some consonant sounds are likely to be suppressed, so that a sample frequency of 16 kHz might be the better choice. Above 8 kHz there are no outstanding frequency components which contribute to sound characterisation. The influence of noise becomes stronger above 8 kHz, so a sample frequency of 32 kHz would be too high. Thus, both 8 and 16 kHz frequencies will be used for experiments to determine the influence of the sample frequency on speech recognition.

The Speech Signal Analysis Technique

The decision about which speech features to use is fundamental. Very little comprehensive data exists on comparative analyses of signal modeling in speech recognition. Convincing comparative data on large speaker-independent continuous speech recognition tasks is simply not currently available [31]. Nevertheless, two trends have emerged. The first is the use of Fast-Fourier-Transform (FFT) based parameters. The second is the use of Linear Predictive Coefficient (LPC) based parameters. The output of these fundamental analysis techniques is transformed by a further signal processing stage which yields the final parameters. An overview of the speech parameters used in many well-known SR-tasks is given in [31]. The mel-frequency cepstral coefficients (MFCC), introduced in section 4.6.6, are the most common

coefficients derived from FFT. Cepstral coefficients (LPCC) are also the most common coefficients derived from LPC. The cepstral coefficients, derived by either FFT or LPC, have the same final meaning since they are both methods of the cepstrum analysis, although the way in which they have been generated is different. An evaluation of monosyllabic word recognition [47] has shown that the FFT derived cepstral coefficients outperform the LPC derived ones, especially in noisy environments.

Both the MFCC and the LPCC parameters are calculated for segments of speech, as stated in section 4.6.5. Often, segment durations in the range of 20 - 30 ms are used [31]. The Hamming window is typically used to weight the samples of a segment. The frame period, which controls the overlapping of the segments, is typically in the range of 10 - 20 ms.

For the MFCCs, the analysis order, say P, determines how many critical bandwidth filters are used as shown in Figure 4.19. The number of output parameters used, say M, determines how many coefficients are calculated per segment. The number of output parameters M is unrelated to the analysis order, but the analysis order should be, typically, twice as large [13]. The same is also true for the LPCCs. A minimum of M is required to guarantee that enough information is transported. The experiment in [47] shows that for MFCCs at least six coefficients succeed in capturing most of the relevant information and the importance of higher order coefficients appears to be speaker-dependent. In the SPHINX system, they used 12 LPCCs.

It has been shown [48] that incorporating energy brings further improvements in recognition. The energy is just a squared sum over the speech samples of a window. Furthermore, it has also been shown in [6] and [49] that the inclusion of spectral derivatives (in time) improved the performance of speech recognisers significantly. This is realised by using the first and second order derivatives of the MFCCs, the LPCCs, and the energy coefficient, which is only the difference of two successive coefficients divided by their spanned time. These derivatives are called delta coefficients and delta-delta coefficients respectively.

As a result, two types of configurations will be used for the experiments. The recognition results should then help to decide the final parameter set to be used in the future. Table 5.3 represents the common parameters which have been applied for both analysis techniques. Table 5.4 shows the parameter set for the FFT-based analysis technique and Table 5.5 the set for the LPC-based analysis technique.

Preemphasis coefficient	0.97
Segmentation window	Hamming
Window duration	30 ms
Frame duration	10 ms
Sample frequency	16 kHz

Table 5.3: Common used parameters independent of the analysis technique.

Analysis technique	FFT derived mel-frequency cepstral coefficients (MFCC)
Number of coefficients M	12
Analysis order P	2×12
Energy	appending normalised log energy
Derivatives	delta and delta-delta coefficients

Table 5.4: Configuration 1: FFT based analysis technique.

Analysis technique	LPC derived cepstral coefficients (LPCC)
Number of coefficients M	12
Analysis order P	2×12
Energy	appending normalised log energy
Derivatives	delta and delta-delta coefficients

Table 5.5: Configuration 2: LPC based analysis technique.

The experimental results given in Table 5.6, evaluated on the TIMIT speech database (see section 5.8) and using the phoneme recognition rate as the measure, show that MFCCs work better, at least for this task. Therefore, only configuration 1 will be considered further.

Analysis technique	MFCC	LPCC
Relative recognition rate in %	100	96.43

Table 5.6: Relative recognition rate of different analysis techniques.

Thus, 30 ms of speech will be described by a parameter vector of dimension 39 according to configuration 1, see Figure 5.5. This parameter vector will be recalculated with the frame period of 20 ms.

MFCC	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	log Energy
$\frac{d\,MFCC}{dt}$	$\dot{C}_0$	$\dot{C}_1$	$\dot{C}_2$	$\dot{C}_3$	$\dot{C}_4$	$\dot{C}_5$	$\dot{C}_6$	$\dot{C}_7$	$\dot{C}_8$	$\dot{C}_9$	$\dot{C}_{10}$	$\dot{C}_{11}$	$\dot{}$ log Energy
$\frac{d^2 MFCC}{dt^2}$	$\ddot{C}_0$	$\ddot{C}_1$	$\ddot{C}_2$	$\ddot{C}_3$	$\ddot{C}_4$	$\ddot{C}_5$	$\ddot{C}_6$	$\ddot{C}_7$	$\ddot{C}_8$	$\ddot{C}_9$	$\ddot{C}_{10}$	$\ddot{C}_{11}$	$\ddot{}$ log Energy

Figure 5.5: Parameter vector as a result of the speech signal analysis used in this approach.

An interesting point is the data reduction which happens in this analysis. For every 20 ms a vector with 39 parameters is calculated. One parameter is represented by 4 Byte in HTK. So every 20 ms 156 Bytes are used from the speech signal. Conversely, 20 ms of 16 kHz sampled raw speech waveform represented by 2 Byte samples, corresponds to 320 Byte. Therefore, data reduction is about 2:1.

5.7 Specification of the Probability Density Function

As previously discussed, the speech feature parameters are represented not directly by the HMM but in the form of probability density functions about the parameters. This is the observation probability density function attached to every HMM state. The following multivariate continuous density Gaussian mixture function (pdf) will be used,

$$b_j(\underline{s}) = \sum_{m=1}^{M} c_{jm} \frac{1}{\sqrt{(2\pi)^{39}\ |\Sigma|}} e^{-\frac{1}{2}(\underline{s}-\mu)'\Sigma^{-1}(\underline{s}-\mu)} \qquad (5.1)$$

where M is the number of mixtures, c_{jm} is the weight of mixture m in state j, μ denotes the mean vector, $|\Sigma|$ is the covariance matrix and b_j denotes the pdf for state j. The mean vector will represent the mean value of each of the 39 parameters. The 39 × 39 covariance matrix represents the correlation between the parameters and the variance of the parameters. The mean, the covariance and the mixture weight are the parameters of Equation (5.1) which will be determined by HMM training. They are reestimated from analysed training speech data represented by the vector derived from Figure 5.5. However, the pdf is not completely specified. The mixture weight c_{jm} and the number of mixtures M must be specified, along with the type of covariance matrix. By using a value of 1 for M, the pdf results in a unimodal distribution which is less complex in time and space. Does a multimodal distribution make

sense? It has been argued [50] that the essential advantage of the mixture densities is that several maxima in the density function can be modelled, which may correspond to different acoustic realisations of the same phoneme due to the coarticulation effect. The use of phonemes and the coarticulation effect make experiments using several mixtures necessary.

Instead of using a full covariance matrix, it is possible just to use a diagonal matrix. A diagonal covariance matrix assumes zero correlation between the parameters and represents merely their variance. Of course, this assumption is never true. This points out the fact that the speech signal analysis techniques rely on the correlation of the parameters. However, the benefit of the diagonal matrix is that the computation of b_j reduces to just a sum of products of Gaussians, whereas for a full covariance matrix the computation of b_j requires a matrix multiplication. In [51] it has been shown that it is possible to model (case 1) a K-dimensional correlated random process by (case 2) a mixture of M uncorrelated, K-dimensional Gaussian processes. In conclusion, the number of mixtures and the choice of a full or a diagonal matrix cannot be decided independently. In general, the more parameters, the more training data will be needed to estimate reliable parameters. It can be assumed that the training data will be too little rather than too much. Thus case 1 or 2 is chosen to favour the minimum amount of parameters. Using the formulas from [51],

$$P_c = \frac{K(K+3)}{2}$$
$$P_u = M(2K+1) \qquad \textbf{(5.2)}$$

where P_c, and P_u denote number of parameters for case 1 and case 2 respectively, the decision depends on the relation:

$$P_u \leq P_c$$
$$M \leq \frac{K(K+3)}{2(2K+1)} \qquad \textbf{(5.3)}$$

The dimension K of the Gaussian has already been specified by the feature parameter vector and is 39. The Relation (5.3) amounts to M=10 with K=39. Hence, as long as $M < 10$, case 2 will be the better choice under the above criterion. However, it is not clear which M should be assigned. In other experiments [49] and [52], values between 2 and 9 were used. Although this work does not exactly match the context of these experiments (different subword units), it could be supposed to be in a similar vein. The best way is to run experiments with different numbers of mixtures. Normally such experiments involve high training costs in terms of time, but, luckily, HTK supports this well. From an arbitrary stage of trained HMMs, it is possible to change some parameters, like the number of mixtures. The HMM can be changed dynamically during the training. Despite this advantage, a certain retraining cost remains.

Equation (5.1) is still not fully specified because the initialisation values of the parameters

that will be trained have not yet been determined. In a study [51] about properties of such pdfs used for HMMs, it is stated:

> *"The most important conclusion from our experiments is that it is absolutely mandatory to have a good initial guess of the means of the density functions to obtain good HMMs."*

A good guess can be generated automatically and can be done by HTK. This is often called initial training or bootstrapping of the HMMs. The initialisation process and the experiments to find the optimal number of mixtures are presented in section 5.9.

5.8 Training Speech

Training data is an important resource for HMM-based speech recognition. The parameters of the HMM have to be estimated from existing speech. The amount of training data dictates the speech recognition rate. In general, the more training data, the higher the recognition rate. However, the recognition rate as a function of training data converges to an asymptote as it does for any statistical learning systems with static parameters. There will be, at some point, a break-even point where more training no longer improves the recognition rate. On the other hand, the more complex the HMM, the more training data is needed to satisfy the HMM parameter space. The complexity of the HMM is dependent on the linguistic unit to be modelled. There is hardly any information available about the optimum amount of training data for a particular task. Lee [53] did some experiments on the amount of training data in the SPHINX system. He gives some general figures for the minimum amount of training data for different linguistic units for speaker-independent training, see Table 5.7. These are only rough recommendations for the minimums. The upper bound may be expected to considerably exceed the recommendations. Current large vocabulary HMM speech recognition systems [54] based on triphones, which are basically phoneme models extended with a predecessor phoneme and successor phoneme, use over 35,000 sentences of speech for HMM training. Similarly, for pure phoneme models, a figure in the range of 1,000 - 10,000 sentences also holds. The question is where we could get the speech data from. Such an amount cannot be recorded by one person. Furthermore, recorded speech only is not sufficient since each phoneme which occurs in the recorded speech has to be labeled by its name with exact time boundaries, since the training algorithm has to know that a certain piece of speech corresponds to a certain phoneme. The phonemes are mapped to the appropriate counterparts of the speech waveform.

Number of training sentences	Linguistic unit
100	Phonetic models
1,000	Phonetic models with simple context (triphones)
10, 000	Phone models with more context (e.g. stress, syllable/ word position)
100, 000	Longer (e.g syllable) models; Rough speaker cluster (e.g. gender) models.
1,000,000	Even longer (e.g. morph, word) models; Detailed speaker cluster (e.g. dialect) models

Table 5.7: Minimum number of sentences for speaker-independent training for different linguist units.

In section 4.3 the boundary problem and the coarticulation effect, which make fully automatic labeling as yet impossible, have been discussed. Labeling is achieved manually by expert spectrogram readers and this is very hard work. Fortunately, such labeled speech databases are available commercially. The Linguistic Data Consortium [55], located at the University of Pennsylvania, sells speech databases. This is, at the moment, the best source of speech data.

For this approach, a speech database called TIMIT [56] was purchased. TIMIT contains speech from 630 speakers of American English and is based on spoken sentences. Ten sentences spoken by each of the 630 speakers have been recorded. The speakers were selected to be representative of 7 different geographical dialect regions of the U.S. A speaker's dialect region was defined as the geographical area of the U.S. where he or she lived during his or her childhood years (age 2 to 10). The dialect regions are outlined in a map in appendix C. There was also an eighth speaker group, consisting of speakers who had moved around a lot during their childhood. 70% the speakers were male and 30% female. In total, there were 6300 sentences and 6100 distinct words. On first impression 6300 sentences may sound a large number but the duration of 6300 spoken sentences is just 6 hours. This figure is poor, considering how many sentences humans hear when learning a language.

The speech database is organised in a hierarchical file structure sketched in Figure 5.6. It is ordered according to the dialect region and speakers. Each speaker is identified by a speaker code, for example "mjc0" or "fjk0", where the first letter designates the gender (f=female, m=male). For every sentence, there is a file containing the raw speech waveform data, a file which contains the corresponding labels and a file which contains only the normal text of the sentence. The sentences are identified by a sentence code, e.g., "sx634".

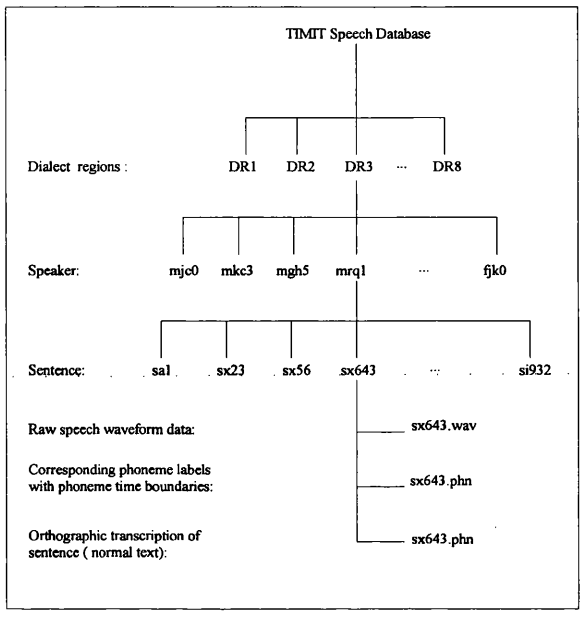


Figure 5.6: Organisation of the TIMIT speech database.

The TIMIT database has been chosen because:

1. It is ideal for phoneme-based speech recognition, since every spoken sentence is labeled with the corresponding phonemes.

2. It is ideal for speaker-independent speech recognition because of the mixture of speakers, covering a broad range of English speech.

3. It is ideal for comparisons because TIMIT was also used in other speech recognition experiments [45][57], so there is reference work available to provide a benchmark.

The TIMIT speech database is used for training, as well as for the evaluation of the acoustic front-end. Therefore, the speech database has to be divided into two subsets, a training set and a test set. 80% of the TIMIT speech database is used for training and 20% is used for evaluation. The speakers who are present in the training set, are not present in the test set, see Figure 5.7.

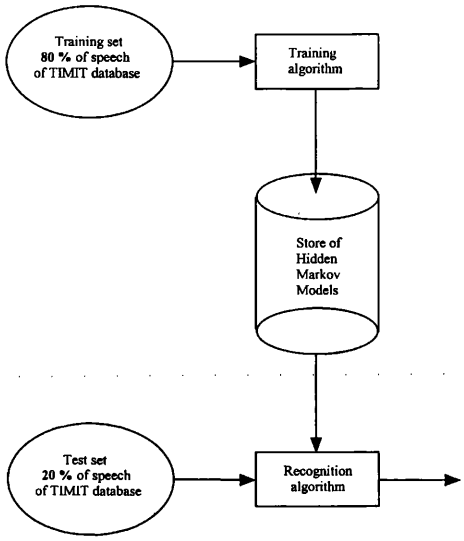


Figure 5.7: Overview of the training and test configuration for a phoneme-based recognition system.

There are three types of sentences distinguished in the TIMIT database. They are marked by the prefixes "sa", "sx", and "si" in their filenames. The "sx" sentences are designed to be phonemically compact. The objective was to provide a good coverage of pairs of phonemes, with extra occurrence of phonetic contexts thought to be either difficult or of particular interest. There are a total of 450 different "sx" sentences contained in TIMIT, where only 7 speakers spoke the same sentence. The "si" sentences were designed to be phonemically diverse. They were selected from existing sources like the Brown Corpus. The selection criteria maximised the variety of allophonic contexts found in texts. There are 1890 different "si" sentences contained in TIMIT, where each of the 630 speakers spoke 3 out of the 1890 sentences. The "sa" sentences were intended to expose dialectal variants of the speakers. There are only two "sa" sentences but they were spoken by all 630 speakers. The "sa" sentences will be excluded from the training and the recognition task because the phonemes which occur in the "sa" sentences would be favoured over those which do not occur. Table 5.8 gives an overview of the training/test subdivision of the speech data.

	Training Set	Test Set	Total
Total Sentences	3696	1008	4704
sx Sentences	2310	504	2814
si Sentences	1386	504	1890
Distinct Text	2340	1717	623
Distinct Words	6099	4891	2371
Distinct Phonemes	45	45	45
Total Speakers	462	168	630
Male Speakers	326	112	438
Female Speakers	136	56	192

Table 5.8: Training and test subdivision of the TIMIT database.

5.9 HMM Training

The introduced specifications of the HMM, the TIMIT speech database, and the HTK toolkit are the starting point for the training of the Hidden Markov Models.

The whole specification of the HMM, which has been introduced in the previous sections, can be accommodated in an ASCII file. The training begins with a single Hidden Markov Model prototype, which is the starting point for all phonemes. It could be argued at this point that different phoneme classes, like consonants and vowels, and different HMM structures should be used to model the differences between these classes. This is certainly not wrong, but, to simplify the first training experiments, one HMM prototype is used for all phonemes. This generalisation may not be very significant, considering that the individual probability density functions of the HMM states provide a huge space for the specialisations of the phonemes. Figure 5.8 shows what the prototype description of the HMM looks like. For example, the identifier "<MFCC_E_D_A>" specifies the meaning of the parameter vector or the speech signal analysis technique. "<DIAGC>" specifies that only a diagonal covariance matrix will be used. The structure shows clearly the prototype pdf attached to each state. The initial values of the mean and variance do not play a role at this stage; they are initialised properly by a special procedure. At the bottom, the transition matrix is given. Although the transitions will also be reestimated at the training stage, their initial values are important because they specify the HMM topology. The value a_{ij} in row i, column j states the initial transition between state i and j. If a_{ij} is set to zero, this denotes no transition between state i and j.

```
<BeginHMM>
<NumStates> 5 <VecSize> 39
<MFCC_E_D_A> <NULLD> <DIAGC>

<State> 2
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0

<State> 3
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0

<State> 4
<Mean> 39
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
<Variance> 39
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0

<TransP> 5
  0.000e+0 1.000e+0 0.000e+0 0.000e+0 0.000e+0
  0.000e+0 5.000e-1 5.000e-1 0.000e+0 0.000e+0
  0.000e+0 0.000e+0 5.000e-1 5.000e-1 0.000e+0
  0.000e+0 0.000e+0 0.000e+0 5.000e-1 5.000e-1
  0.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0
<EndHMM>
```

Figure 5.8: HMM definition according to HTK standard.

Figure 5.9 gives an overview of the arrangement of the HTK tools for the complete HMM training session. The rectangles represent the HTK modules, and the circles mark input and output data. The training task is subdivided into three stages: the initialisation stage, single model training, and embedded training combined with model refinement. These three stages are performed by the HTK tools HInit, HRest, and HERest. The subdivision of the training task will become clear later on. The complete training task requires managing an immense number of files with different content. Furthermore, training will take some time; it may take a couple of months. Good organisation is required to handle the training task. It must be automated as much as possible; however, partial results must always be saved to check the progress of the training. An organised file structure, which reflects the organisation of the

whole training, recognition and evaluation environment, is given in appendix B.

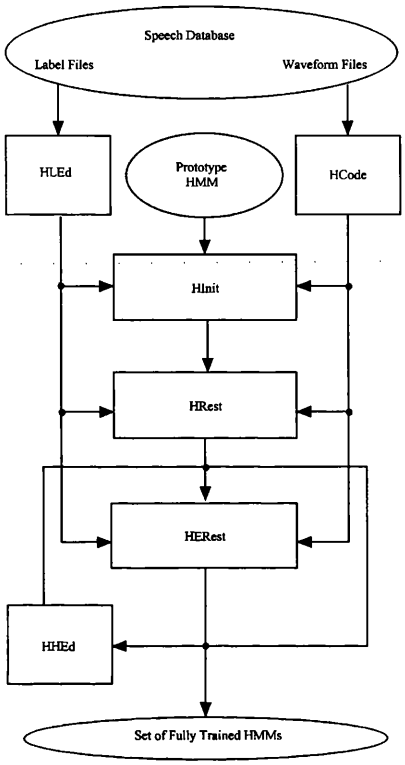


Figure 5.9: Architecture for HMM training.

Speech Signal Analysis Stage

The first step is the speech signal analysis of the speech waveform. It is convenient to perform the speech signal analysis stage in advance for each speech utterance in the TIMIT database, for both the training as well as the test part, because only the analysed speech is needed after this point. This can be done by the module HCode. This module performs the speech signal analysis fully automatically. The type of analyses defined in section 5.6 can be specified by command line options of HCode. HCode operates in a file-based way taking the raw speech waveform file as input and storing the speech features in another file. It is best to devise a straightforward way of organising the whole training session. HTK is totally file-based. All inputs and outputs must be provided in files and there are many different kinds of files to manage. Therefore, it is convenient to use the same extensions for the same files and to use scripts to automate the training session. Figure 5.10 outlines the speech signal analysis stage where the "*.mfcc" files contain the analysed speech. The extension "mfcc" has been chosen to denote the applied speech signal analysis technique.

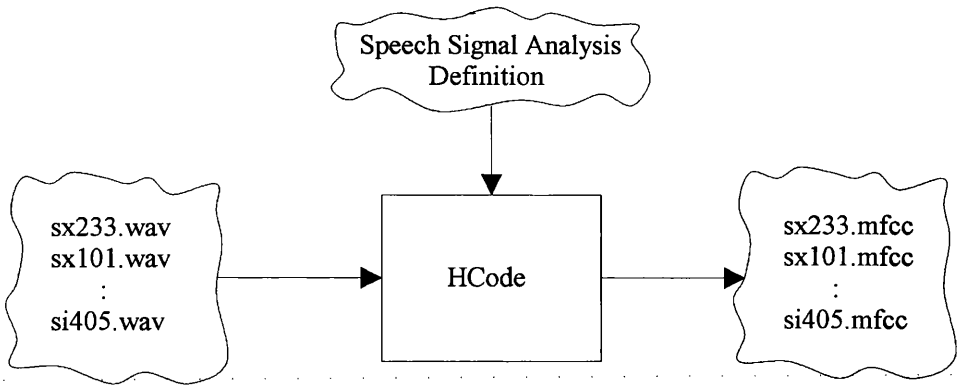


Figure 5.10: Speech signal analysis with HTK tool HCode.

The raw speech waveform is not needed for the rest of the speech recognition task. Therefore, in the following, the file containing the speech features will be referred to simply as the speech file.

Preparation of the Phonemic Transcription

As previously mentioned, HMM training is a supervised learning procedure. Therefore, in addition to the recorded speech utterances, the corresponding phonemic transcription has to be supplied. The phonemic transcription employed in TIMIT contains some additional phonetic transcriptions. Remember, phonetic transcription describes the pronunciation, more precisely than is possible by phonemes by using symbols denoting stress, the type of articulation or general allophones. The p48 phoneme set will be used, see section 5.4. TIMIT uses a total of 64 phonemic and phonetic symbols. Therefore, the TIMIT transcriptions of the utterances must be converted into the reduced p48 phoneme set. This can be done either by a conventional script editor or, more conveniently, by using the HTK stream label editor HLEd. Figure 5.11 shows the conversion of a TIMIT transcription into a p48 transcription.

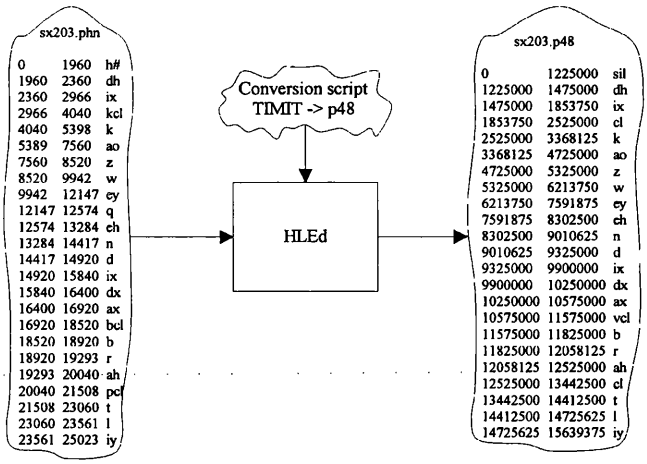


Figure 5.11: Conversion of TIMIT phonemic alphabet into p48 set with HLEd.

The phoneme conversion for the utterance "The causeway ended abruptly" is given as an example. The numbers in front of the phonemes mark the start and end time of the individual phonemes. TIMIT uses the sample number as the time unit, but HTK requires absolute times in units of 100 ns, so the times are converted too. In addition, the phonetic transcription describes phonemes in more detail. To remove the phonetic symbol, it must be merged with the succeeding phonetic-described phoneme, for example "bcl,b" is merged to "b". For each speech waveform file there is also a phonemic transcription, called a label file. Correspondingly, phoneme conversion is also carried out over the whole TIMIT database in advance.

It may look as if all HTK tools are optimised to support the subword unit phoneme. It only seems so because the phoneme is being used in this approach and thus all explanations are based on phonemes. If words or syllables were used, it would seem as if they were the basis. There is no preferential treatment of phonemes by HTK; this is entirely the decision of the HTK user.

Initialisation

As stated in section 5.7, it is absolutely essential to have a good initial guess for the HMM parameters before starting the real reestimation procedure, in particular for the parameters of the pdf of Equation (5.1). The transition probabilities have already been initialised manually by the HMM topology specification.

The necessary initialisation can be achieved by the tool HInit. It takes the speech files and the corresponding label files as input. The question is how much data is needed for the initialisation. In general, the literature suggests that for this bootstrapping operation only a small part is needed, unlike the final reestimation procedure. However, the question is what a small part is. It is plausible that there is a minimum amount which must be provided.

However, there might also be an upper limit, above which the initialisation would cause bad initialisation values. Studying the initialisation algorithm led to the conclusion that there is no upper bound. Therefore, the whole training set will be used for the initialisation task. Thus, 3696 speech files and 3696 label files, corresponding to the 3696 training sentences, have to be processed by HInit. By using a script for automatisation, this task can be completed in one run. It takes about 4 hours on an average loaded Sun/Sparc10 workstation.

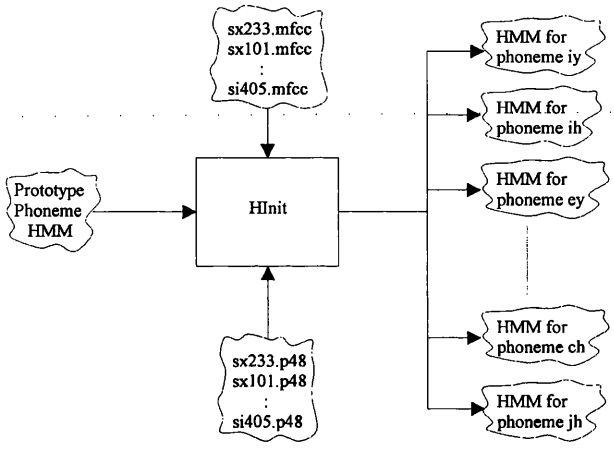


Figure 5.12: HMM initialisation with HInit.

HInit initialises each HMM separately. For example, initialisation for the HMM of phoneme "iy" works as follows. All speech segments of phoneme "iy" in all 4704 training files are cut out by using the time information provided by the label files. The prototype HMM is taken and the speech segments of phoneme "iy" are equally divided amongst the HMM states. Then simple averages are used to initialise the mean and the covariance of the pdf from Equation (5.1). In the next step, each speech segment of the phoneme "iy" is aligned to its corresponding maximum likelihood state sequence by using the Viterbi algorithm [37]. The means and covariances are reestimated. The Viterbi alignment and the estimation process are repeated until the initial value estimates converge. The convergence can be controlled by setting the number of estimation cycles and a convergence limit as an option of HInit. Twelve estimation cycles have been used. Figure 5.13 shows the convergence of the average logarithmic probability of phoneme "iy". That is "log $P(\underline{s}_{iy,i}(m)|M)$" averaged over all speech segments i...N for phoneme "iy". The convergence limit is measured as the relative change of the average logarithmic probability.

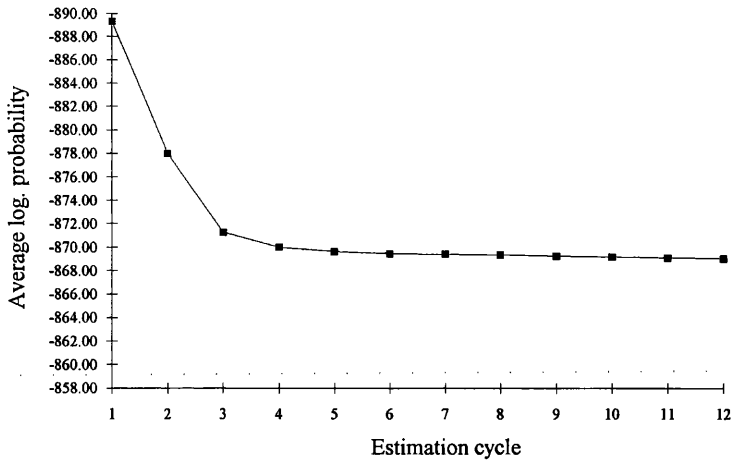


Figure 5.13: Convergence of the HMM initialisation shown by the average log probability of phoneme "iy".

The average probability could have been calculated without taking the logarithms, which might have been more intuitive. However, all speech training and recognition algorithms work with logarithmic probabilities, usually in base 10. Logs are used to avoid underflow situations caused by handling very small probabilities during the calculations. For example, consider the figure 10^{-869}, which is the average probability after 12 cycles taken from Figure 5.13. For comparison, the smallest value of the commonly used single-precision floating-point standard IEEE-754 is about 10^{-38}. The logarithmic probability measure is, therefore, necessary for proper assessment.

The output of the initialisation stage is one single initialised HMM for each phoneme of the p48 set.

Single Model Training

The initialised HMMs are the input for the first real training stage. This stage is performed by the HTK tool HRest. It takes the speech files and the corresponding label files as input.

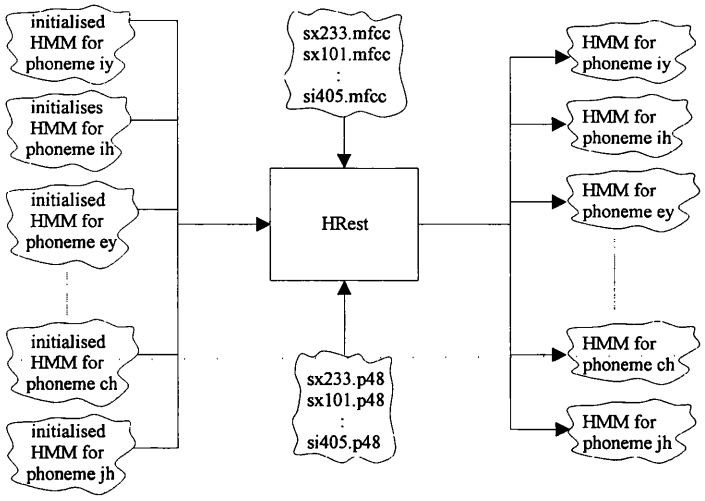


Figure 5.14: HMM training with HRest.

The question is how much data is needed for the first training stage. To answer this question an important point has to be made. HRest trains the HMMs as single models; this may seem to be obvious, but is different from the final training stage, which will be explained below. Single model training needs separated speech corresponding to the models. This means that, for example, the HMM of phoneme "iy" must be trained exactly with the speech segment of phoneme "iy", which conforms to the supervised learning paradigm. Direct separated phoneme speech is not available in the TIMIT database because whole sentences are used. However, the sentences are transcribed by the phonemes and by the start and end times of the phonemes. This time information can now be used to cut out the speech segments of the corresponding phonemes. The question now arises why a final training stage is needed at all. In general, the really expensive task in transcribing speech is the labeling of the time boundaries, since it cannot be done automatically. This task must be done by expert spectrogram readers. On the other hand, providing just the phonemic transcription can be automated by an appropriate translation algorithm. Now the final training stage does not need the time information of the phonemes, but only the phonemic transcription. Thus, the HMMs can be initialised and trained with a small amount of expensive training data. The final training can then follow with a larger amount of cheaper training data.

In this approach only the TIMIT database is used for training and TIMIT is fully transcribed with phonemic transcription as well as time information, so that it could be argued that the final training stage is unnecessary. This is true for the case of training word HMMs for an isolated word speech recogniser, since words are less sensitive to the coarticulation effect and, therefore, it makes sense to train them as single models. Conversely, there is no doubt that phonemes are highly sensitive to the coarticulation effect. The advantage of the final training stage is that it takes into consideration the context in which the phonemes occur so that the phoneme HMM becomes context sensitive to some extent. This sensitivity is important because words ought eventually to be recognised where phonemes occur always in context. The coarticulation effect is a serious problem. Phoneme HMMs are certainly not the

best models for modelling the coarticulation effect but they must be regarded as a compromise between the available training data and the flexibility of vocabulary independence. Returning to the question of the amount of training data needed for HRest, the whole training set will be used for the same reasons as during the initialisation stage.

HRest works as follows. For example, training for the HMM of phoneme "iy": all speech segments of phoneme "iy" in all 4704 training files are cut out by using the time information provided by the label files. Then the initialised HMM of phoneme "iy" is taken as the starting model for the Baum-Welch reestimation algorithm [36]. The training algorithm calculates the new HMM estimates, which are the transition probabilities, and the mean and covariance of each state according to Equation (5.1). The algorithm calculates the new parameters for all speech segments in one pass. The HMM of phoneme "iy" is finally updated with the new estimates. The Baum-Welsch reestimation is repeated with the same speech data, but based on the updated HMM, until convergence is reached. The convergence can be controlled by setting the number of estimation cycles and a convergence limit as an option of HRest. Twenty estimation cycles are used. Figure 5.15 shows the convergence of the average logarithmic probability of phoneme "iy". That is "log $P(\underline{s}_{iy,i}(m)|M)$" averaged over all speech segments i...N for phoneme "iy". The convergence limit is measured as the relative change of the average logarithmic probability.

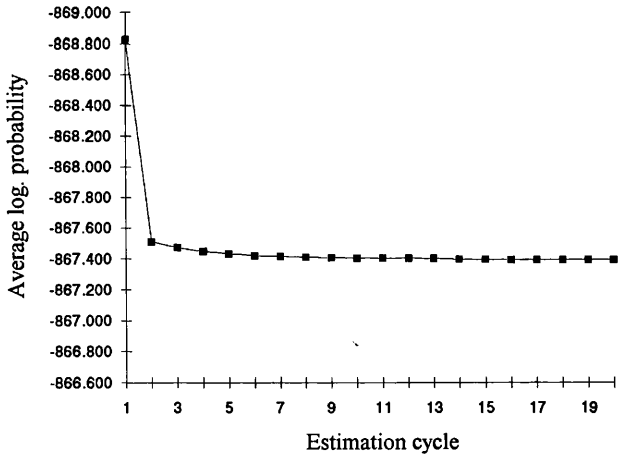


Figure 5.15: Convergence of the single model Baum-Welsch reestimation showed by the average log probability of phoneme "iy".

It can clearly be seen that the algorithm converges quickly. It is not guaranteed, however, that it is within a global maximum.

Embedded Training

The final training, performed by HERest, is mainly responsible for the recognition results. Like HRest, HERest uses the Baum-Welsch reestimation procedure for training. However, the main difference is that models are not trained as single entities, but are connected to a composite model according to the phonemic transcription provided by the label file. Figure 5.16 shows a concatenated HMM for the utterance "The causeway ended".

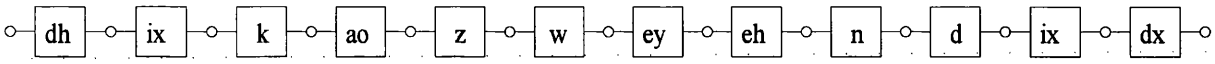


Figure 5.16: Concatenated HMM, which is composed of single phoneme HMMs corresponding to the phonemic transcription of the utterance "The causeway ended".

This is simply the concatenation of the single phoneme HMMs. Concatenation of HMMs is in principle possible. This can be proven if the HMM is simply regarded as a non deterministic Finite State Machine. In HTK, concatenation is realised by a unique initial and a unique final state. They have no acoustic observation density functions attached, i.e., the pdf of Equation (5.1). Consequently, they can be used as glue states to connect HMMs. The concatenated HMM is regarded as one large HMM, which is then used for training in place of the single phoneme HMMs. The phonemes are trained in the context of their neighbourhood phonemes. That is the reason why the training stage is called embedded training. Furthermore, the coarticulation effect is taken into account by this kind of training because phonemes are trained/embedded in their natural context. HERest works as follows. First, a concatenated HMM is constructed for each utterance of the training speech. Then the Baum-Welsch reestimation procedure is applied to calculate the forward and backward probabilities for each utterance and the corresponding concatenated HMM. The transition probabilities and the acoustic observation density functions are not updated after each utterance, but after all utterances have been processed. This is possible by storing partial results which record the state occupation of each HMM state and the transitions out of each state. These partial results are updated after each processed utterance. Each phoneme is updated with the new estimated parameters after all utterances have been processed. HERest performs only a single iteration and therefore has no convergence criterion unlike HRest. Embedded training, however, is usually not finished in one pass. Therefore, HERest must be performed several times using the same speech data but the new updated HMMs from the previous run. HERest outputs the overall logarithmic probability $P(\underline{s}(m)|M)$, which is averaged over all phoneme models and training utterances. The probability is also normalised by the duration of a speech frame because different utterances always have different length. This probability can be used to observe the convergence of the estimated HMMs. Figure 5.17 shows the convergence of the embedded training after 18 estimation cycles.

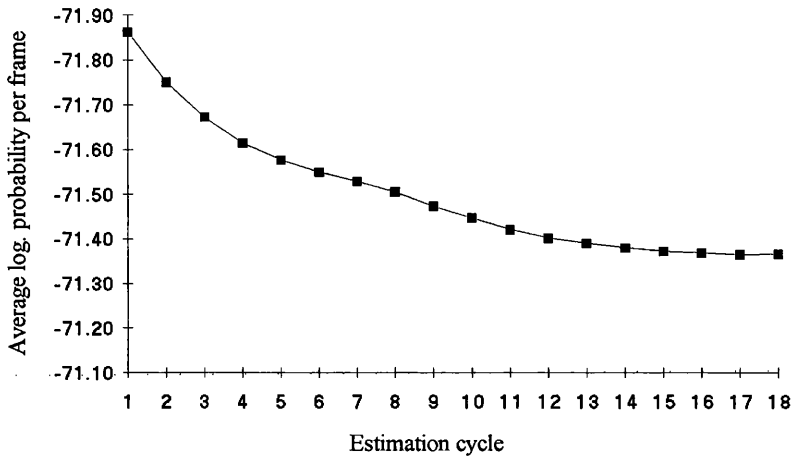


Figure 5.17: Convergence of the embedded training shown by the log probability per frame average over all phoneme HMMs.

Model Refinement

The embedded training is not yet finished because only one mixture of the pdf of Equation (5.1) has been used so far. In section 5.7, it was stated that experiments with several mixtures must be carried out. It has further been demonstrated that, as long as the number of mixtures is roughly below 10, using a diagonal matrix for the covariance of the pdf of Equation (5.1) is justified. Diagonal matrixes have been used so far under the assumption that there would be too little training data to train HMMs with full covariance matrixes satisfactorily. Increasing the number of mixture components requires more training data. However, the optimum number of mixtures can be found by experiments using a key feature of HTK called model refinement. Model refinement works as follows. The trained HMMs from the embedded training stage are the input for the first refinement step. The script-based HMM editor HHEd allows the HMM specification of a given HMM to be changed. It is possible to change the number of mixtures of each acoustic observation density function at each state for a given HMM. For example, a change for an HMM from one mixture to two mixtures is realised as follows by HHEd: The mixture weights are halved so that the sum of the pdf of Equation (5.1) is always one, the original mean and covariance simply being cloned. The two identical mean vectors are then made different by adding 0.2 standard deviation to one and subtracting the same amount from the other. An HMM with an extended number of mixtures is then the starting point for further embedded training. An embedded training run, as explained above, must follow since HHEd is not a training algorithm. There would be no effect without a following training stage. After the HMMs have been retrained, a new mixture update can follow from two to three mixtures and so on. What convergence criterion should be used as an indication that the optimum number of mixtures has been found? An easy convergence criterion is simply to take the recognition result as the measure. This measure has been used in the way that all phoneme HMMs have been evaluated over the complete test set after each

incrementation of the number mixtures. For the embedded training runs after each mixture update, only 5 estimation cycles, as recommended in 13, have been performed. Figure 5.18 shows the recognition rate dependent on the number of mixtures. The recognition rate is always evaluated after the embedded training run. The recognition rate is the percentage of correctly recognised phonemes.

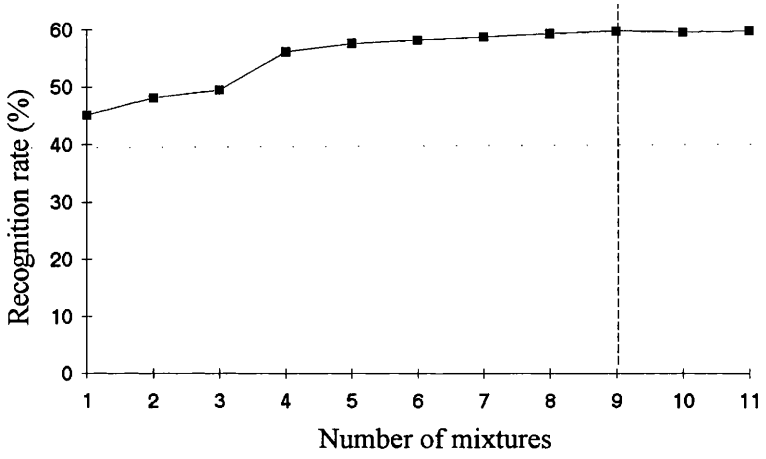


Figure 5.18: The recognition rate which is dependent on the number of mixtures.

The jump between 3 and 4 mixtures is very noticeable. After four mixtures, there is only a slight improvement. The recognition rate seems to converge after 9 mixtures and thus 9 mixtures will be used as the optimum number of mixtures from this point. Of course, it must be kept in mind that this experiment is conditioned on the used training data. In addition, the use of diagonal matrixes, seems theoretically, as discussed in section 5.7, to be justified.

An experiment like this takes considerable computing time. Embedded training and model refinement can, theoretically, be performed fully automatically by setting up suitable scripts but the computer platforms used must be guaranteed stable. The training process must be observed not only to control the convergence of various stages but also to control the estimated HMM parameters. There are some critical cases which must be checked; for example that the estimated variances do not get too small. The problem of variances which are too small is that unseen data (speech data not covered by the training set) will not be recognised. HTK allows a threshold to be set for the covariance and this is done after the fourth mixture. The threshold is estimated as 10% of the average over all covariances of all states of all phoneme HMMs.

This is the complete training stage for the training set of the TIMIT database. The result is 48 phoneme HMMs and one silence HMM, with a recognition rate of nearly 60%.

5.10 HMM Recognition

The training stage yields 48 phoneme models and an additional silence model. This set of HMMs can be regarded as the interface to the "machine's cognition system" or just as the acoustic front-end. The recognition rate has already been mentioned because it has been used for several control experiments for the training task. Recognition and training, however, are completely independent tasks. Thus, training can be done in advance, and the trained HMMs can afterwards be integrated into several speech recognition applications. This point is important, considering the time- consuming nature of training. Figure 5.19 shows the arrangement of HTK tools necessary for recognition, HVite being the recognition engine.

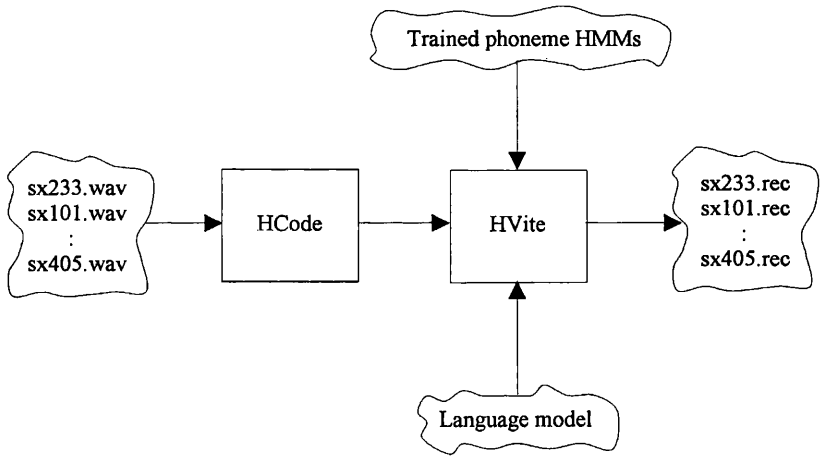


Figure 5.19: HTK architecture for recognition.

The set of trained HMMs is used by HVite for the recognition task. The HMMs can be regarded as the interface to the training task, which is the only remaining link. The speech waveform files from the defined test set of the TIMIT database are used for evaluation. Arbitrary recorded utterances can be used as input. They must only be accommodated in files, since all HTK tools operate with file input and output.

Speech Signal Analysis Stage

HCode works in the same manner as in the training stage, and performs the speech signal analysis. The whole database, the training and the test set, have been converted at one time and consequently this step need not discussed further. However, it is interesting to consider whether exactly the same speech signal analysis must be performed as during the training stage. The answer is that only the speech feature vector which is represented in the HMM

must be the same. This means that the parameters listed in Table 5.9 must be identical to the definitions used in the training task.

Analysis technique	FFT derived mel-frequency cepstral coefficients (MFCC)
Number of coefficients M	12
Energy	appending normalised log energy
Derivatives	delta and delta-delta coefficient

Table 5.9: Parameters which must be identical to the training stage.

Only these parameters are specified in the HMM, which is the only link to the training task. The parameters listed in Table 5.10, however, are not part of the HMM specification and thus can be changed.

Analysis order P	2×12
Preemphasis coefficient	0.97
Segmentation window	Hamming
Window duration	30 ms
Frame duration	10 ms
Sample frequency	16 kHz

Table 5.10: Parameters which can be changed.

This fact can be exploited for some experiments in the recognition task. For example, it can be experimented with the impact of lower sampled test data. It is quite a practical situation that the speech to be recognised is lower sampled. For the evaluation of the TIMIT test set, however, exactly the same speech signal analysis definitions have been used, so that the speech input to the recognition engine HVite is of exactly the same format as for the three training tools HInit, HRest, and HERest.

Language Model Interface

Another important interface of HVite is the Language Model. The Language Model specifies how the HMMs are arranged by a regular grammar. It can be used to build higher level linguistic units based on lower level linguistic units where the lowest level linguistic unit must

end up in an HMM. For example, phoneme HMMs can be concatenated to form words. The concatenated HMMs can then be regarded again as an HMM according to the equivalence to Finite State Machines. The HMMs constitute the terminal symbols of a regular grammar. The original HMM name, for example the phoneme label, or another name can be used as an identifier. Nonterminal symbols can be made up from arbitrary character strings. The regular grammar is specified by rewriting rules using the extended Backus Naur form. The grammar specification is simply written in an ASCII file. Figure 5.20 shows a grammar specifying the Language Model of the ten digits.

```
$NINE = n ay n;
$EIGHT = ey t;
$SEVEN = s eh v ax n;
$SIX = s ih k s;
$FIVE = f ay v;
$FOUR = f ao r;
$THREE = th r iy;
$TWO = t uw;
$ONE = w ah n;
$ZERO = z ih r ow;

($ZERO | $ONE | $TWO | $THREE | $FOUR | $FIVE | $SIX | $SEVEN | $EIGHT | $NINE)
```

Figure 5.20: Language model for the ten digits according to the HTK syntax.

Normally, a parser module parses the grammar and substitutes each nonterminal symbol by the corresponding terminals which are actually the HMMs. In Figure 5.20, the ten digit names are the nonterminals and the phonemes are the terminals. One large finite state network is built up according to the syntax of the last line of Figure 5.20. This line is also called the network. This means that the ten digits are arranged in parallel. Another arrangement, for example, could be, just by ignoring the "|" symbol, a sequence of digits from zero to nine. That would mean that the recogniser would only be able to recognise the digits "0123456789". A recogniser using the network of Figure 5.20 can recognise single digits. A loop construction can be used to extend the network of Figure 5.20 for recognition of arbitrary digit strings, simply by putting the whole line in angular brackets. The complete network of Figure 5.20 is basically treated as a single HMM. The HMM is really only a Finite State Machine, which is a finite state network. Actually, powerful recognition tasks can be realised with the syntax provided. Figure 5.20 also shows another important feature, the flexibility of phonemes. Theoretically, arbitrary tasks can be realised with phoneme HMMs by ignoring the incompleteness of phoneme recognition. However, in practice there are limitations, arising not only from the uncertainty of phoneme recognition but also from the complexity of the Language Model.

Language modelling is beyond the scope of the acoustic front-end and is indeed a large research topic. Behind the Language Model is the meaning of P(w) of Equation (4.31) and this probability can represent prior knowledge about the language in question. Therefore,

chapter 7 describes language modelling in more detail. Only a simple network is used for the evaluation of the TIMIT test set, where the phonemes, including the silence model, are arranged in parallel without the use of any higher level mapping.

Viterbi Recognition

Once the network has been built up by HVite, the Viterbi recognition algorithm is applied to find the best path through the network. The output of the recognition algorithm is the path of HMM labels, in the case of phoneme HMMs a sequence of phonemes. Additionally, the identified time boundaries of each phoneme are also given. Phoneme output is not always helpful; for example, in the ten digit example it would be more practical if the output were the actual digit names. This can be realised by using special features of the HTK grammar, which are the key symbols "WD_BEGIN%name", "WD_END%name". A statement "WD_BEGIN%four f ou r WD_END%four" would be treated as a terminal symbol or as a terminal node in the network with the effect that the output of HVite would be "four" instead of "f ou r".

The ten digit language model of Figure 5.20 is not suited for phoneme evaluation in the TIMIT database, since TIMIT speech consists of spoken sentences. One type of Language Model might model explicitly each sentence and put all sentences in parallel, like the ten digits. Another way would be to model each word occurring in the test set explicitly and put all words in parallel building up a loop, so that each arbitrary word can follow each other word. The silence model can be used in both cases to model breaks. However, each of these methods would already incorporate a priori language knowledge and would disguise the real recognition power of the phonemes. Therefore, a network is needed which adds no further a priori knowledge to the phonemes. Such a network can be realised by arranging the phonemes in parallel according to Figure 5.21.

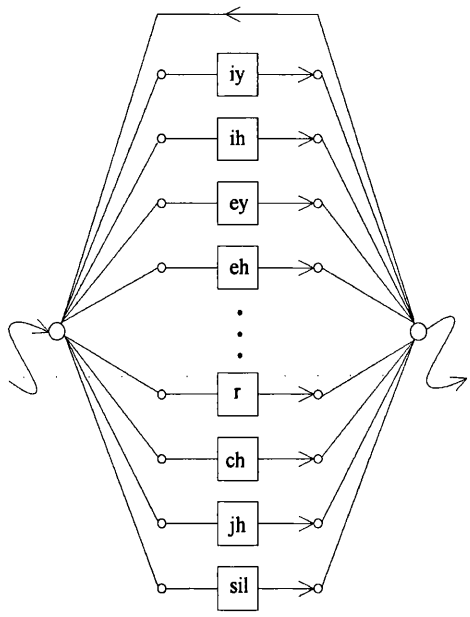


Figure 5.21: Phoneme-in-parallel network bearing minimum a priori language knowledge.

This network will be called a "phoneme-in-parallel network". It bears the minimum a priori language knowledge, which is purely the existence of phonemes. The pure existence of phoneme models must be taken into account to be correct. For example, the probability of recognising phoneme "zz" (could stand for the german sound "z" of the word "Zeug") with the network of Figure 5.21 is zero because it is not modelled in that network. In the phoneme-in-parallel network, each phoneme can follow each other phoneme with an equal likelihood, including the silence model, so that P(w) of Equation (4.31) is 1/48. This factor can be neglected in the calculation of the maximum likelihood path. The phoneme sequence can be arbitrarily long. Thus, arbitrary phoneme sequences can be generated by the phoneme-in-parallel network. If the phoneme-in-parallel network works, this means that there would be no recognition errors, arbitrary speech which can be expressed by the phonemes could be recognised and the speech recognition problem would be solved.

In conclusion, the phoneme-in-parallel network is an ideal network. The next section will show the final phoneme recognition result using the phoneme-in-parallel network of Figure 5.21. It will give a first indication how ideal that network actually is. The complete TIMIT test set, which is 1007 sentences, has been passed through the recognition tool HVite. The output of HVite is a file containing the recognised sequence of phonemes for each input utterance, see Figure 5.19.

5.11 Evaluation

After the training and the recognition task have been completed, the main question is:

How well do the trained phoneme HMMs reflect the English speech sounds?

This question is similar to the IR problem: how well do the estimated probabilities of relevance, given a small set of documents, reflect the overall queries and documents?

Evaluation can be done in quite different ways with quite different results. The speech of TIMIT is in the form of sentences. The output of the recognition algorithm is a phonemic transcription of the sentences. The existence of the correct phonemic transcription is the basic requirement for evaluation. The correct transcription is provided by the TIMIT database: it has already been required for the training task. The correct transcription has been converted to the p48 phoneme set, see section 5.9. A simple evaluation method would be to compare the phonemic transcriptions and to decide that an error occurs if the phonemic transcriptions are not equal. This could be done, for example, with standard Unix comparison tools by comparing each reference file "<sentence>.p48" with the corresponding recognition result file "<sentence>.rec". However, this method is totally inappropriate for the evaluation of phoneme recognition rate because an error would be recorded even if one phoneme in a sentence is misidentified. In each of the 1007 sentences, at least a few phonemes will be misidentified. Another method is to apply a dynamic programming procedure in which the two transcriptions are aligned against each other under the condition of minimising the phoneme errors. Substitution, deletion and insertion can be counted as errors. The principle is exactly the same as in the template-based approach discussed in section 4.5.2, except that string symbols are being compared instead of speech patterns. This method would provide the substitution, deletion and insertion errors for each evaluated sentence. These figures could then be used to measure the quality of the phonemes. In particular, it provides information about phoneme confusion, which is interesting for phonemes denoting similar sounds.

This evaluation method is supported by the HTK tool HResult. This tool has been used for the evaluation. Figure 5.22 shows the organization of the evaluation procedure.

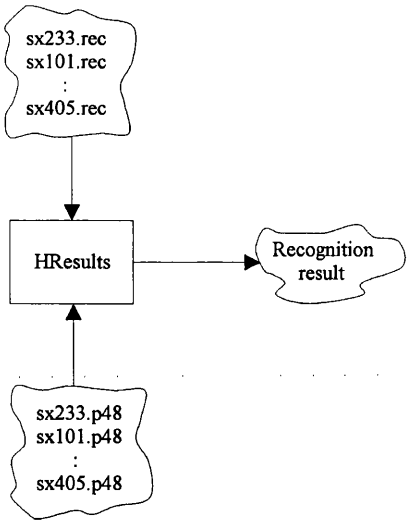


Figure 5.22: HTK architecture for evaluation.

The recognition result is stored in files with the name "<sentence_id>.rec". The reference transcription is provided in corresponding files of the form "<sentence_id>.p48". The output is the optimal alignment between each sentence and the substitution, insertion and deletion errors. Figure 5.23 shows such an optimal alignment for a typical recognition result. The dynamic programming procedure uses different penalties for the distance minimisation. A choice can be made between standard penalties 10, 7, and 7 for substitution, insertion and deletion and US NIST (National Institute of Standards and Technology) penalties which are 3, 3, and 4 respectively. The NIST penalties have been used in this evaluation.

```
Correct:     sil   dh ix cl k   ao z w ey eh n vcl d ix dx ix vcl b r ah cl p l iy   ix cl dh ix sh    ao r  sil
Recognised: sil p dh ix cl k f aa z w iy eh n     d ix dx ax vcl b r aa cl p w iy y ix th dh ix sh er ao er sil
```

Figure 5.23: Correct and recognised phonemic transcription of TIMIT test sentence "Those musicians harmonise marvellously".

The Recognition Rate and the Recognition Accuracy are two commonly used measures for the quality of phoneme recognition. These are defined as:

$$Recognition\ Rate = \frac{H}{N} \times 100\%$$

$$Recognition\ Accuracy = \frac{N - S - I - D}{N} \times 100\% \quad (5.4)$$

where H is the number of correctly recognised phonemes, S the number of substitution errors, I the number of insertions, D the number of deletions, and N the total number of phonemes. Table 5.11 shows the quality of the phonemes. Remember that the speakers from the test set are not present in the training set. Additionally, evaluation has also been done over the training set in order to show any bias. The figures represent the sum of all evaluated sentences.

	Test set	Training set	Total set
Recognition Rate	59.85 %	67.9 %	65.76 %
Recognition Accuracy	47.52 %	52.82 %	51.42 %
Total Number N	50754	140225	190979
Hits H	30377	95209	125586
Substitutions S	16447	37341	53788
Deletions D	3930	7675	11605
Insertions I	6259	21118	27377

Table 5.11: Phoneme recognition rate of the acoustic front-end.

The results are comparable with other evaluations using the same database. The recognition rate reported by Lee and Hon [45] is 59.85%; however, they used a reduced set of 39 phonemes. Robinson and Fallside [57] reported a recognition rate of 66.7% with a bigram model along with a reduced set of 39 phonemes, but using the connectionist approach. The HMMs may not yet have reached their maximum recognition accuracy. There might still be some space for improvement. However, the comparison indicates that the basic decisions need not be wrong. It is also noticeable that the insertion and deletion errors are relatively small; it is plausible that, in general, these types of errors are harder to correct than substitution errors. However, the substitution error rate is quite high. It is possible that many recognition errors are caused by similarities in speech sounds, which are exaggerated by non-optimal pronunciations. A measure which deals with the kind of substitutions is the phoneme confusion matrix as shown in Figure 5.24.

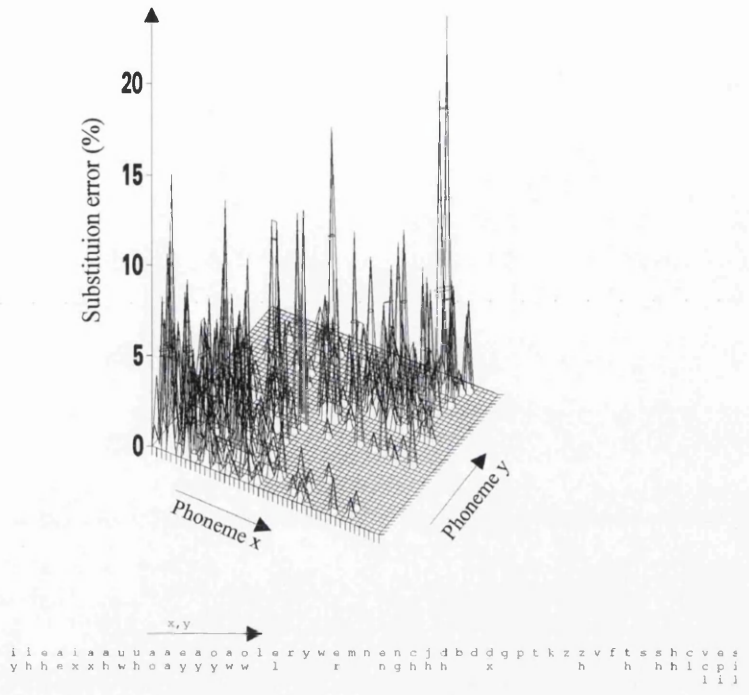


Figure 5.24: Phoneme confusion matrix.

This matrix clearly shows that most confusions happen between phonemes denoting similar sounds. For example, consider the phoneme "ao" occurring, e.g., in "/a/bout". It occurs a total of 605 times in the test set and is recognised 306 times, and is very often (116 times) confused with the phoneme "aa" occurring, e.g., in "c/o/t". On the other hand, take the phoneme "w" as in "/w/ant" with which "ao" is never confused, since "w" denotes a completely different sound. The information from the phoneme confusion matrix can be used to reduce the phoneme set to a smaller size by clustering the most confused phonemes. This would increase the phoneme recognition rate; however, one must be aware that reducing the phonemic transcription of a language also increases the number of indistinguishable words. This statement is verified by the experimental results shown in Figure 5.1. The recognition problem would only be postponed to the Language Model. It is certainly a good idea to cluster phonemes which have a high confusion rate.

Table 5.11 shows only the overall results. More detailed evaluation is presented in appendix C.

5.12 Conclusion

This chapter has discussed the design of an acoustic front-end. Many decisions have been taken. A comprehensive literature study and real experiments have been used to explore the speech recognition parameter space. The main decision was to use phonemes as the basic linguistic unit. It is not yet certain whether the right choices have been made.

The phoneme training was quite expensive in terms of time, increased even further by experiments to find optimum parameters. The expense of generating such phoneme models is one of the main reasons why such models are not provided in a public domain. The work described in this chapter has provided a set of phoneme HMMs. The quality of the phonemes can be regarded as a basis for further work. It must always be remembered that the phoneme is an abstract or an ideal unit. Furthermore, the phonemes have been trained with just 6 hours of speech, compared to the training speech collected by a human, a much larger figure. More importantly, the evaluation took place without a priori language knowledge, which is the major component in human speech understanding ability. These results show that a sophisticated Language Model is necessary. However, it cannot be concluded that even the most sophisticated Language Model can remove all uncertainty which has not been solved already by the acoustic front-end. Perhaps phonemes are not the correct basic linguistic unit. Nevertheless, chapter 7 will show that these phonemes can be used for serious recognition tasks by applying some simple language models.

Chapter 6

The Speech Recognition Evaluation Interface

6.1 Introduction

The acoustic front-end is basically a set of trained phoneme HMMs. They have been trained for use in speaker-independent systems. The evaluation presented in chapter 5.11 was intended to assess only the pure phonemes. This has been done on the TIMIT database with support from the toolkit HTK. HTK is an excellent toolkit for research, training and evaluation of HMMs but provides no facilities for setting up a real speech interface with a real-time speech input channel. Such a speech interface is required for testing the acoustic front-end in a real world environment. This gap has been filled by the design and implementation of a Speech Recognition Evaluation Interface (SREI). This chapter will present an overview of this tool.

6.2 Requirements

A task analysis for SREI amounts to the following requirements. Basically, an environment is required to give real-time recording, speech recognition, and evaluation based upon the acoustic front-end. It must be able to incorporate various Language Models. It should be independent of the currently used linguistic unit, the phoneme, so that in the future other units modelled by HMMs can be used. Furthermore, the evaluation of simple recognition tasks should be supported by assessment and statistical facilities. Simple recognition tasks should be generated automatically regardless of grammar. The collected statistics and recognition tasks should be storable for later use. Important parameters should be changeable for research purposes. The speech interface should also be able to incorporate additional externally defined recognition steps and it should be directly attachable to target applications like text-based IR systems. The tool should run under Unix and be easily portable between Unix architectures.

6.3 Implementation

A tool has been designed to fulfil the above requirements. It uses as the core of the speech recognition the recognition routines from HTK. The HTK Release 1.5 offers a collection of executable programs, but there is no software library which can be linked with other applications allowing access of the essential algorithms within the application code. The HTK tools are ideal for incorporating into shell scripts, however the lack of a software library makes it difficult to employ HTK code in other applications despite the availability of the source code. Nevertheless, the recognition routines have been extracted, and the source code has been modified so that the algorithms are reentrant. The speech signal analysis stage of HTK has also been borrowed and modified.

Real-time audio input is always critical in terms of machine dependencies. Currently, there is no audio hardware standard. Each workstation manufacturer provides its own audio setup, which varies on the offered sample frequencies, word length and audio coding schemes. Even the workstation series of Sun Microsystems has five different audio hardware devices. For example, the Sparc 10/20 series offers various sample rates up to 44,1 kHz. The Sparc 5 series offers 8 kHz and 16 kHz sampling rates and Sparc 1/2 only 8 kHz. Therefore, the hardware differences must be considered despite the abstract C audio interface to Unix.

In conclusion, there is a need for an abstract level which compensates for the differences in the audio hardware. A good example of such an abstract level is X-Windows. For example, no application software designer needs to be concerned about the horizontal frequency of the screen; it is hidden by the X-server which runs for each physical display. A similar principle for audio devices is required, where for each audio device there is a server which compensates for the hardware dependencies. An abstract level provides independent functions for accessing the audio hardware. Fortunately, such an audio system is currently in development and the first versions are already available. It is called AudioFile [58] and works analogously to X-Windows, so it is also possible, among other things, to transmit audio data over networks (network transparency). One drawback of AudioFile is that there are not many audio servers available yet. AudioFile has been used to guarantee some kind of hardware independence and to support this project. Furthermore, the XView/OpenWin library has been used to guarantee standard graphical hardware independence, according to the X-Windows system. Thus, the software architecture of SREI is based on the four components sketched in Figure 6.1. It shows a typical client-server architecture, which is an important feature of SREI. In this way, the client (SREI) can run on a powerful, expensive, remote workstation, and the graphics and audio interface can run on a cheap, low power, local machine. This architecture makes research with SREI for large, time-consuming recognition tasks (large vocabulary) feasible.

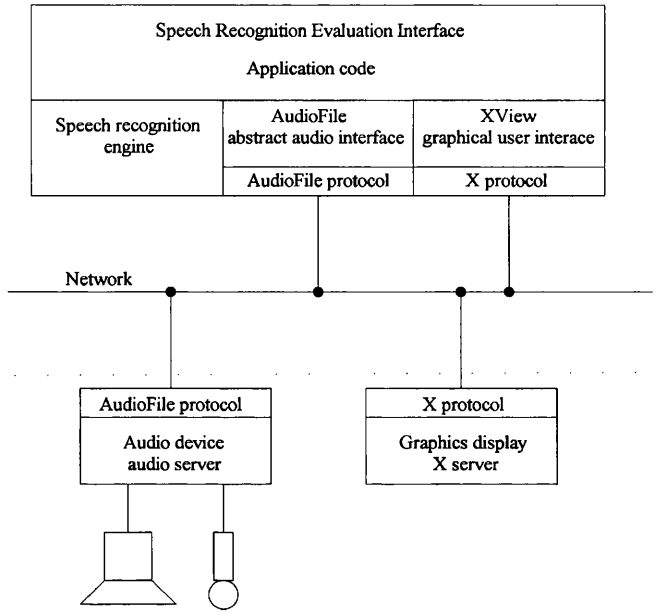


Figure 6.1: Software architecture of the Speech Recognition Evaluation Interface.

6.4 Properties

What can actually be done with the Speech Recognition Evaluation Interface? Figure 6.2 shows the graphical user interface (GUI) of SREI. The GUI is composed of 8 different function areas. The main panel is responsible for performing the recording, recognition and evaluation tasks. The control panel is for setting up the recording, recognition and evaluation parameters. The message display describes the current action. Two busy signs in the right upper corner indicate when recording or recognition is busy, since these processes take some time. The active vocabulary displays the vocabulary of the current recognition task. Next to the active vocabulary is the corresponding statistical display. The active vocabulary is an essential part of the recognition task. There are two types of active vocabulary: "special vocabulary" and "general vocabulary". Figure 6.2 shows a special vocabulary which is based on a constrained language model, yet has the benefit of an automatic evaluation facility and various other features. The general vocabulary type has no constraints on the Language Model, but there is no automatic evaluation facility. The most important features and the two different vocabularies will be explained in more detail.

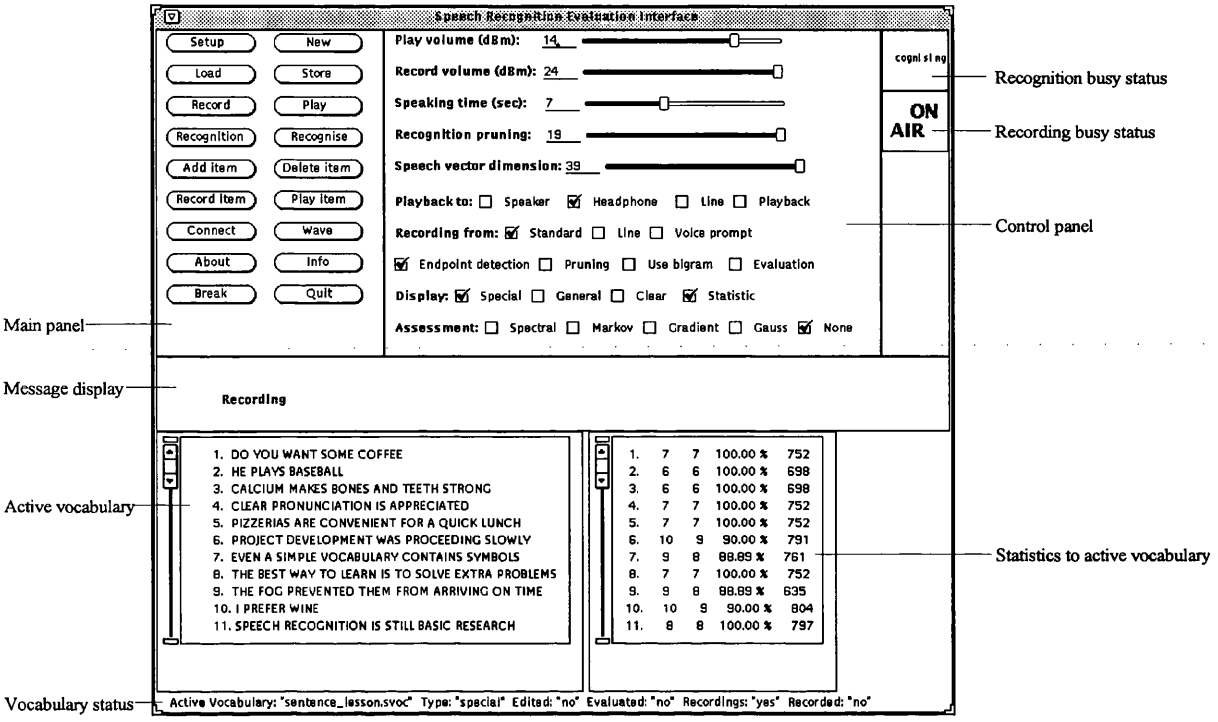


Figure 6.2: View of the Speech Recognition Evaluation Interface.

The Audio Interface

Before SREI can operate, the appropriate AudioFile server must be installed on the workstation where the audio hardware is being used. This is, usually, the same machine on which the X-server is running. Speech input can be provided by a microphone which can be connected to the workstation's built-in soundcard line or microphone connector. Of course, it is also possible to connect any external sound source with a line level, like a tape. The Sennheiser HMD 414 is recommended as the microphone. It is especially designed for speech recording in noisy environments. There is an enormous difference between the standard workstation microphones and such high quality microphones. The sample frequency is determined by the AudioFile server. Depending on the machine various sample frequencies are offered. The sample frequency is set as a command line option on the AudioFile server. The SREI application automatically takes over the correct sample frequency. The recommended sample frequency is 16 kHz. The audio output, for example, a playback of recorded speech, can be passed to the workstation's built-in speaker, headphone or line connector. Additionally, speech can also be imported and exported via files by accepting and generating Sun-Audio-format-headered files. Speech can be recorded and is kept in a single speech buffer. This single buffer feature allows different combinations of the functions

provided. The buffered speech can be used by several other functions, like the play back or recognise function. Speech can also be recorded and passed directly to the recognition task, which corresponds to a real speech interface mode. The maximum speaking time can be set manually by using the speaking-time slider. It must be set according to the maximum expected duration of the utterance. The utterance must then be spoken within that time. If the speaking time is set to zero, an automatic detection feature is switched on, which detects the end of the utterance. In either case, the recording busy status indicates when recording is in progress.

An additional wave display, see Figure 6.3, allows the assessment of the recorded utterance. For example, the Clipping Factor is an indicator for an input signal level which is too high, causing the speech signal to be clipped. The SNR measures the speech signal energy with respect to the background noise. The Oscilloscope shows details of the speech signal.

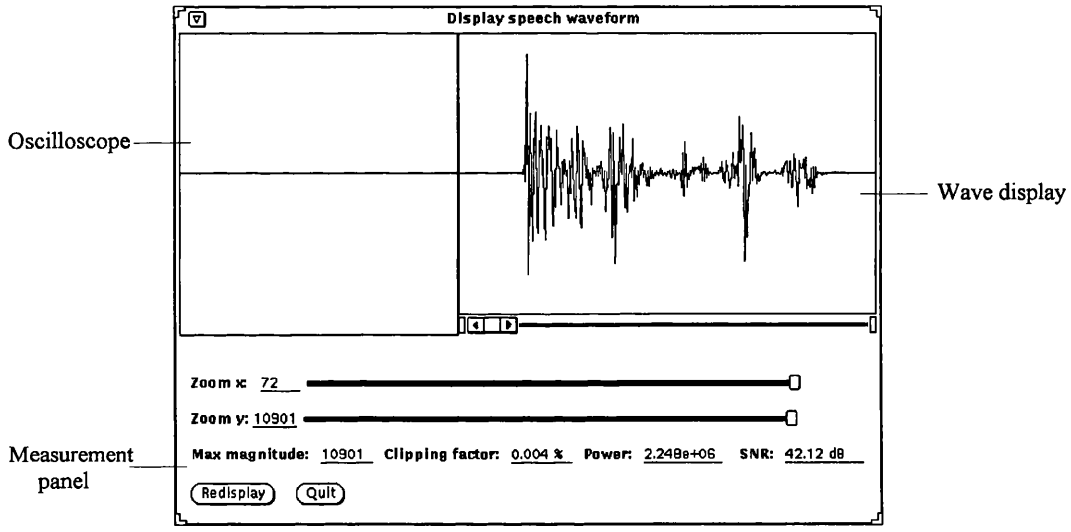


Figure 6.3: Wave display of the Speech Recognition Evaluation Interface.

The Special Vocabulary

Two different types of vocabulary can be defined. Once defined, instances of each vocabulary type, can be saved and retrieved. Recognition is performed on one of those predefined vocabularies. The special vocabulary type has been designed especially for the testing and evaluation of the acoustic front-end, applied to higher level linguistic units. The acoustic front-end of chapter 5 is essentially the 48 phoneme HMMs, so the special vocabulary is based upon these. However, it can also be used for other modelled linguistic units, although this property will not be discussed any further. The special vocabulary can be generated interactively with the GUI of Figure 6.2. The special vocabulary is actually a Language Model based on word-pair modelling. Word-pair modelling is explained in chapter 7.4. It is a

constrained model of connecting phoneme HMMs together to obtain higher level units like words or sentences. A special vocabulary is a list of sentences, where the individual sentences are the units to be recognised. Using the extended Backus Naur Form, the list can have the following form:

$$
\begin{array}{lll}
\textit{LIST} & \rightarrow & (\textit{ID SENTENCE})^{+} \\
\textit{SENTENCE} & \rightarrow & (\textit{sil WORD})^{+}\ \textit{sil} \\
\textit{WORD} & \rightarrow & \textit{PHONEME}^{+} \\
\textit{PHONEME} & \rightarrow & \textit{ih} \mid \textit{ih} \mid \textit{ey} \mid \ldots \mid \textit{r} \mid \textit{ch} \mid \textit{jh} \\
\textit{ID} & \rightarrow & \textit{DIGIT}^{+} \\
\textit{DIGIT} & \rightarrow & 0 \mid 1 \mid \ldots \mid 9
\end{array}
\qquad \textbf{(6.1)}
$$

The "+" symbol means one or more repetitions of expressions. The "|" symbol denotes alternative expressions. The upper-case words denote nonterminals, and the lower-case words denote the phoneme HMMs according to the p48 phoneme set given in appendix A. Grammar (6.1) states only the syntactical constraints. It states that a list can be made up by a list of sentences preceded with an integer identifier. The first semantic constraint is that the identifiers and the sentences must be unique, and that the list consists of unique sentences. A sentence can be composed of one or more silence-word combinations, ended by silence. Thus, a sentence can be just a single word enclosed by silence, and is thereby failing to conform to the linguistic definition of a sentence. A word is composed of a phoneme string. The most important semantic constraint is that the word must be a valid English word, which can be translated into a phoneme string using the p48 phoneme set. This is an essential part of the special vocabulary. The phoneme output of a word or sentence is unimportant; it is the orthographic transcription of the word which is required. Therefore, it is the orthographic words of a sentence rather than its phoneme string, which are displayed, see Figure 6.2. The phoneme string is hidden, although the sequence of phonemes making up the sentence is actually recognised. A word must be checked for being a valid English word and then translated into the corresponding phoneme string. The SREI user need not bother with the grammar definition (6.1). She is only prompted to type in the text of the required sentence. The sentence is automatically translated into phonemics and added to the list. Figure 6.4 shows the process of generating a special vocabulary.

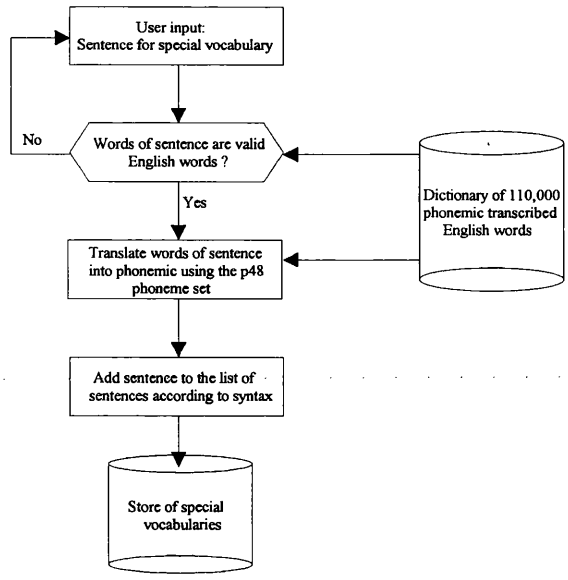


Figure 6.4: Generation of a special vocabulary.

A dictionary, maintained by Carnegie Mellon University, provides 110,000 phonemically transcribed English words. Requested words which are not in the dictionary are rejected, however, the dictionary is a good approximation to the complete English vocabulary. Once a special vocabulary has been created, it is kept on the hard disk and can be changed later in various ways, e.g. by adding and deleting sentences. In addition, there is no restriction to a single vocabulary, so it is possible to define many different special vocabularies. A file-based organisation is used to access and retrieve different special vocabularies. For example, Figure 6.2 shows a special vocabulary consisting of 11 sentences. In addition, it is also possible to save a template utterance for each sentence. Lists of single phonemes, words, and phrases, as well as syntactically and semantically complete sentences, may be defined.

How can recognition be performed on a special vocabulary? Once a special vocabulary has been defined, it must be loaded. It is then automatically translated into a recognition network appropriate to the recognition algorithm. Recognition can be performed interactively by speaking one of the sentences contained in the list. The recognition algorithm recognises the most likely sentence with respect to the spoken utterance. The recognition result is displayed. An automatic evaluation facility can be used to record and save the recognition result. Grammar (6.1) states that words are modelled by a surrounding silence, although it does not show how this works in detail at the HMM level. This silence unit, which is actually modelled by an HMM, should capture natural pauses between words. At the HMM level, it is modelled by an optional unit, as in Figure 6.5.

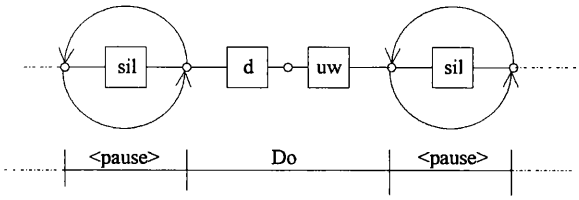


Figure 6.5: Modelling of pauses in the special vocabulary.

Figure 6.5 shows the word "do" modelled by phoneme HMMs. It could signify the beginning of a sentence. The silence model is placed before and after the word "do", but there is a zero transition and a repetition transition included in the silence model. Thus, zero pauses or pauses of undefined length can be modelled. The theoretical aspect of the underlying Language Model of Figure 6.5 is described in mode detail in chapter 7.4. When an utterance is recorded by SREI, it is always surrounded by a number of pauses, which are actually background noises. The surrounding noise of the complete utterance can be captured either by the silence models, as in Figure 6.5 or by switching to an endpoint algorithm, which automatically removes the surrounding silence. The endpoint algorithm is based on [59] and is a further feature of SREI for experimental purposes. The pauses or silence between words can only be captured by the silence models.

The General Vocabulary

The special vocabulary is restricted to word-pair modelling, giving the advantage of being able to define recognition vocabularies interactively and have them evaluated automatically. The phoneme HMMs, however, can be concatenated into arbitrary Language Models of varying complexities. Language modelling itself is described in chapter 7. HTK provides a Language Model interface. Arbitrary phoneme networks can be specified, based on production rules of a regular grammar. The specified Language Model is accommodated in an ASCII file, see the example in Figure 5.20. SREI is able to import these Language Models. It is possible to apply arbitrary Language Models; they must only conform to the HTK syntax stated in [13]. For recognition, a Language Model must be loaded. SREI prompts the user to speak an utterance. The utterance is then matched against the Language Model. The recognition output is the most likely path through the Language Model, with respect to the utterance. The output symbols depend on the specification of the Language Model. The recognition output for the language model specified in Figure 5.20 would be the pure phoneme sequence. However, there are facilities which support the output of higher level units, e.g., the digit names, instead of the phoneme sequence of the digits.

Furthermore, SREI offers an important feature which is particularly useful when a general vocabulary is loaded. For example, the phoneme-in-parallel network of Figure 5.21 outputs the phoneme sequence for each spoken utterance. Each English spoken utterance can be aligned to this Language Model because there are no constraints. But there is no language knowledge incorporated. Of course, the recognised phoneme sequence is noisy, but this noisy

phoneme sequence could be further processed by error-correction modules. The noisy phoneme sequence could be passed to a string-matching algorithms, and the output of the string-matching algorithm could then be passed, e.g., to a text-based IR system as a query. SREI offers a mechanism which allows the execution of third-party applications. The recognition output can be redirected to the third-party applications. The output of the third-party applications can then be displayed by SREI. Several third-party applications can be concatenated in a row where the output of the former is the input of the latter and so on. Thus, SREI can be employed as a speech recognition server for various client applications.

6.5 Conclusion

In this chapter, a tool has been described, which allows the testing of acoustic front-ends in real environments by providing a real-time speech interface. The tool supports diverse Language Models. A constrained Language Model has been especially designed for interactive evaluation of simple recognition tasks, like recognition of words, phrases, or sentences. The 48 phoneme HMMs are the basic resource of the tool. A broad class of speech recognition systems can be covered by this constrained Language Model; for example, digit recognition systems, small vocabulary systems, discrete utterance recognisers etc. The tool is particularly interesting for experimentation on recognition tasks. The response to various parameters can be investigated. The general vocabulary provides the whole language modelling facility of HTK. The tool can even serve as a real speech recognition server for several client applications. The next chapter goes beyond the acoustic front-end into language modelling. The Speech Recognition Evaluation Interface has been extensively used for carrying out research on several Language Models.

Chapter 7

Beyond the Acoustic Front-End, Language Modelling

7.1 Introduction

It is obvious that phonemes are only an intermediate representation of speech. What is finally required is the speech content, in the form of words. Phonemes are the basis for vocabulary independence, because arbitrary words can be modelled simply by concatenating the phoneme HMMs. However, this chapter will show that, as the vocabulary size grows, the Language Model becomes more important. The acoustic front-end solves only one part of Equation (4.31). The Language Model which solves P(w) is the current research challenge for large vocabulary speech recognition systems. Language modelling already starts at the phoneme level. It will be shown that applying simple Language Models enhances the quality of phoneme recognition. This chapter will discuss the most important Language Models, as well as their application in the acoustic front-end of chapter 5. Real world evaluation with the Speech Recognition Evaluation Interface will demonstrate what can be achieved with the 48 phoneme HMMs.

7.2 N-Gram Modelling

The redundancy of a language can be exploited in various ways. This is actually what we understand by language knowledge. Redundancy is just a more pragmatic expression of Coding Theory. This suggests the application of proven techniques from that area. Shannon [44] investigated English language redundancy. He wrote:

> *"...anyone speaking a language possesses, implicitly, an enormous knowledge of the statistics of the language. Familiarity with the words, idioms, clichés and grammar enables him to fill in missing or incorrect letters in proof-reading, or*

to complete an unfinished phrase in conversation".

He demonstrated with a simple experiment that English is highly predictable. For example, knowing the previous spoken words, the next word can be guessed. This fact can be exploited in language modelling by reformulating the last sentence as: what is the probability of the next word, given the previous spoken words. Given this probability and the evidence of the previous spoken words, P(w) of the fundamental Equation (4.31) can be replaced by $P(w_j|w_1w_2...w_{j-1})$. This probability states a priori knowledge and must therefore be estimated in some way. It is impossible to estimate reliably this conditional probability for all words and all possible sequences in a given language. Therefore, in practice, the following approximation is used:

$$P(w_j|w_1w_2...w_{j-1}) \approx P(w_j|w_{j-N+1}w_{j-N+2}...w_{j-1}) \qquad (7.1)$$

It makes the assumption that only the previous N-1 linguistic units have any effect on the probability for the next linguistic unit. This is one of the most popular language modelling techniques in large vocabulary speech recognition. Equation (7.1) is usually estimated over large text corpora. Typically used N-grams are bigrams, trigrams and tetragrams. Extending the acoustic front-end by a reliably estimated N-gram model of, say, about 65,000 words basically gives today's large vocabulary system. The above introduction has been in terms of words; however, the symbol w can be any linguist unit. Therefore, in the context of this thesis, the symbol w always denotes a linguist unit; special instances of linguist units will be pointed out. In the rest of this chapter, the ideas about language modelling will be applied at the phoneme level; here, w denotes a phoneme. However, the examples work in the same manner as for all other linguistic units. The following examples are based on the recognition of sequences of phonemes, which has been discussed in section 4.7.5. The following Language Models are special cases of the universal Language Model of Equation (4.60).

The phoneme-in-parallel network of Figure 5.21, which has been used for the evaluation of the acoustic front-end, is a no-grammar model, in which it is assumed,

$$P(w_j|w_1w_2...w_{j-1}) = 1 \text{ for all } j \qquad (7.2)$$

where w_j denotes an individual phoneme. This means that every phoneme has equal likelihood of being followed by every other phoneme. The no-grammar model provides the minimum on a priori language knowledge, which is just the pure existence of the phonemes. The acoustic front-end will be enhanced by N-gram models. Two models have been applied, the unigram model and the bigram model. The unigram model is the consequence of considering no previous phonemes. It is just P(w), so that it is assumed:

$$P(w_j|w_1w_2...w_{j-1}) = P(w_j) \qquad (7.3)$$

However, P(w) may be justified. It is simply the a priori probability of measuring or observing a certain phoneme. The simple fact that some phonemes occur more often than other phonemes shows how useful this knowledge is. For example, phoneme "ix" occurs more often than phoneme "zh". Thus, the probability P(w) can be estimated and used as a Language Model. P(w) can then be estimated over a given corpus of phonemic transcribed text.

This has been done over the complete TIMIT database. The complete set contains 190,979 phonemes, so that the estimates are significant. The estimated probability for each phoneme is given in appendix D. It can be seen that, for example, the estimated probability for phoneme "zh" is 0.0012 and for phoneme "ix" 0.052. This is quite a considerable difference, which causes recognition problems for rare phonemes. Therefore, a lower limit should be set to prevent total misrecognition of rare phonemes. The following simple smoothing function has been applied:

$$P'(w_j) = \begin{cases} 5\times10^{-3} & for\ P(w_j) < 5\times10^{-3} \\ P(w_j) & else \end{cases} \tag{7.4}$$

The constant has been chosen such that the original probability applies to 85% of the phoneme set. The next model which has been applied is the bigram model. It considers the previous spoken phoneme. It is assumed:

$$P(w_j|w_1w_2...w_{j-1}) = P(w_j|w_{j-1}) \tag{7.5}$$

The conditional probability has also been estimated over the TIMIT database by using frequency information. Equation (7.5) must also be smoothed for the same reasons as stated above. This is even more important for the bigram model because the effective training data for estimating Equation (7.5) is less than for estimating Equation (7.3). The unigram probability could be used to smooth rough estimates of the bigram probability. The following smoothing function has been used:

$$P'(w_j|w_{j-1}) = \begin{cases} 5\times10^{-3} & for\ P(w_j) < 5\times10^{-3} \\ 0.2\times P(w_j) + 0.8\times P(w_j|w_{j-1}) & else \end{cases} \tag{7.6}$$

The weights are chosen so that Equation (7.6) is weighted heavily in favour of the bigram probability. The conditions on the weights are such that they have a sum of one, in order to satisfy the probability framework.

Both models should improve the phoneme recognition rate, the bigram model to a greater degree than the unigram. Evaluations of both Language Models have been carried out to demonstrate their impact. The phoneme-in-parallel network, sketched at Figure 5.21, has been used as the core Language Model of the acoustic front-end. The incorporation of the unigram and bigram probabilities can be seen just as an additional compulsory state, which follows each final HMM state where the state occupation probability is just the bigram probability. Table 7.1 represents the phoneme recognition rate and accuracy for the three models including the no-grammar model, evaluated on the TIMIT test set.

Language Model	Recognition rate	Recognition accuracy
No-grammar model	59.85 %	47.52 %
Unigram model	60.43 %	47.95 %
Bigram model	62.18 %	50.73 %

Table 7.1: Impact of various Language Models on the phoneme recognition rate.

The unigram model improves recognition only slightly. However, the bigram model is apparently a helpful Language Model for improving the quality of phoneme recognition. Furthermore, the computational costs of incorporating the bigram model in the recognition cycle are low because it involves only one extra addition (due to the use of logs) per phoneme HMM. Thus, in both cases, use of the bigram model is recommended.

7.3 The String-Matching Experiment

A simple approach to finally produce words would be just to apply a string-matching algorithm, which matches the recognised phoneme string against a phonemic described dictionary. The phoneme-in-parallel network augmented, with the bigram model, can be used as the Language Model. It provides language knowledge only at the phoneme level, but it is, on the other hand, the most flexible model because there is, in practice, no vocabulary limit. As discussed in section 5.11, recognised phoneme strings are always noisy. Therefore, the string-matching algorithm must be error-tolerant to some extent. There are plenty of string-matching algorithms which can do this. Expectations of recognition performance were very low; however, its simplicity and some curiosity were the motivation for doing this experiment. Figure 7.1 shows the flow chart for the string-matching experiment. The complete experiment has been carried out with the Speech Recognition Evaluation Interface. It works as follows. Individual spoken words are requested. The surrounding silence is removed by the endpoint detection algorithm. The acoustic front-end provides the corresponding phoneme string by

applying the Viterbi recognition algorithm to the phoneme-in-parallel network. The length of the recognised phoneme string (the number of phonemes) is determined. Then all phonemic transcriptions of the words in the dictionary of size V, which have the length m of the recognised phoneme string ± u phonemes tolerance, are matched against the recognised phoneme string by the string-matching algorithm. The output is a list of the N best recognised words.

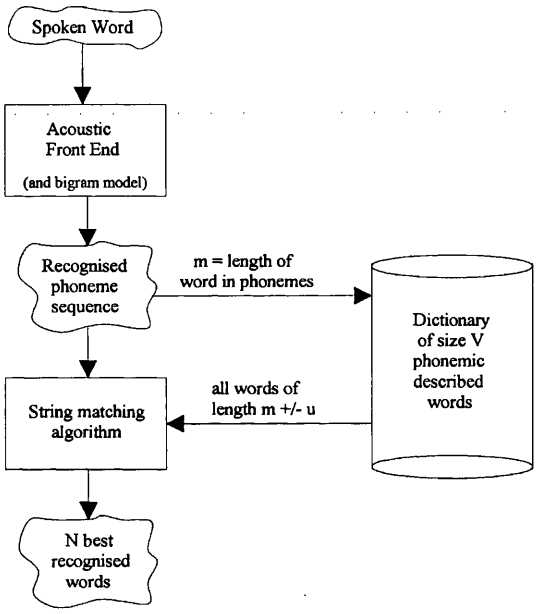


Figure 7.1: Architecture of the string-matching experiment.

The Shift-And algorithm described in [60] and a dynamic programming (DP) algorithm described in [61] have been applied for string matching. Both algorithms are based on the Levenshtein metric and allow string comparisons with k differences. A string x has a distance k to a string y if string x can be converted into string y by a sequence of k substitutions (S), insertions (I) or deletions (D) of characters. Another class of string-matching algorithms contains algorithms that allow k mismatches, which means that only substitution errors would be considered. These algorithms would not be accurate enough for phoneme recognition because of the high number of insertion and deletion errors, see Table 5.11. The string comparison is carried out between the recognised phoneme string and the reference phoneme string from the dictionary, so the characters, are in this case, the phonemes. The reference phoneme string with the smallest distance k to the recognised phoneme string is the match. The following error weighting function has been used for the DP algorithm,

$$w(S,I,D) = 21 * S + 17 * I + 15 * D \tag{7.7}$$

where the weights reflect the frequency of each of the errors, as in Table 5.11. The Shift-And algorithm uses equal weights for the three types of errors. The advantage of having such a weighted string-matching result is that it allows ranking of the N best strings. This ranking scheme has indeed been used, so that the output is the N best strings, ranked by the number of errors.

An interesting question is the assessment of the probability that a word, say w_x, has been spoken and the string-matching algorithm outputs the word w_y. This is the posterior probability $P(w_x|w_y)$, where w_y is the evidence provided by the string-matching algorithm. It is clear that w_x,w_y are from the same vocabulary, say a dictionary of size V. To incorporate the whole framework of the acoustic front-end and the Language Model, the acoustic measurement $\underline{s}(m)$ is added as further evidence, thus $P(w_x|\ \underline{s}(m),\ w_y)$. By applying Bayes' rule, this becomes,

$$P(w_x|\underline{s}(m),w_y) = \frac{P(\underline{s}(m),w_y|w_x)\ P(w_x)}{P(\underline{s}(m),w_y)} \tag{7.8}$$

and by resolving the joint probabilities of Equation (7.8):

$$P(w_x|\underline{s}(m),w_y) = \frac{P(\underline{s}(m)|w_x)\ P(w_x)}{P(\underline{s}(m))}\ \frac{P(w_y|\underline{s}(m),w_x)}{P(w_y|\underline{s}(m))} \tag{7.9}$$

The first term of Equation (7.9) corresponds directly to Equation (4.31) which is the acoustic front-end combined with a Language Model. In the case of the bigram model, Equation (7.6) can be used for $P(w_x)$. It must be remembered that Equation (7.6) is the bigram probability for phonemes and w_x here denotes a word, thus w_x must be replaced by the conjunction of the phonemes of w_x as in Equation (4.60). The second term of Equation (7.9) can be further resolved by checking the conditions. The acoustic measurement $\underline{s}(m)$ brings no evidence for the belief in w_y. This is intuitively not obvious, but consider the input of the string-matching algorithm; the algorithm cannot recognise any string based on speech features at all. Thus, Equation (7.9) can be rewritten as:

$$P(w_x|\underline{s}(m),w_y) = P(w_x|\underline{s}(m))\ \frac{P(w_y|w_x)}{P(w_y)} \tag{7.10}$$

The prior probability $P(w_y)$ is dependent on the dictionary size V, and additionally on the length m. The length of the phoneme string w_x can be obtained from w_y with the tolerance of $\pm$ u, but where can we get $P(w_y|w_x)$ from? It must be a result of the string-matching algorithm. In [61] it is shown that the distance k, defined above, between strings x and y can be interpreted as:

$$distance = -\log(likelihood) \tag{7.11}$$

This is a likelihood measure, not a direct probability, but it will be used as a quantity to assess the probability of obtaining string y, given string x. The distance k is an additional output of the string-matching algorithm. Thus, the second term of Equation (7.10) can be assessed by,

$$\frac{P(w_y|w_x)}{P(w_y)} \sim \frac{10^{-k}\ V}{|w_y| \pm u} \tag{7.12}$$

where $|w_y|$ means the length of the phoneme string in phonemes. Equation (7.12) can be used for error assessment or for probability-based ranking.

The evaluation has only been carried out by a single speaker, but it took place, in a real natural environment, using the Speech Recognition Evaluation Interface. The test speaker is not a native English speaker and has a German accent. In general, there is a noticeable degradation in the recognition rate for non-native speakers. So, although evaluation was only carried out with a single speaker, the fact that the speaker was not English and, more importantly, that the speaker's voice was not included in the training data (speaker-independent evaluation) must be taken into consideration. This kind of evaluation is quite time consuming. Therefore, only a few examples have been tested; however, they demonstrate a trend. The dictionary used has 6300 words. A word means, in this sense, a unique string, not a lexical word. For example, the words "university" and "universities" are treated as different words. This definition of a word is reasonable for speech recognition, in particular for this string-matching experiment, since "universality" is as similar to "university" as to "universities", for example. The dictionary contains small words, such as prepositions, as well as long, content-bearing words. The tolerance u of the phoneme string has been set to 2. Appendix E shows the results. The Shift-And algorithm was outperformed by the dynamic programming algorithm, possibly because errors were not weighted. Therefore, the dynamic programming algorithm has been used for the evaluation presented in appendix E. The number of test examples is too small to calculate reliable recognition rate figures. This is the reason why the results are presented in the form of the 15 best recognised strings for various words. The results show that, on average, the spoken word is among the 15-best recognised but it must be admitted that, in general, it works only for long words which have more than two or three stressed syllables, like the word "taxi". It does not work very well for short words like "the", "car" and so on. One reason for this is the ratio between the string length and the number of errors. This ratio is, in general, smaller for long words than for short words. The consequence is that errors in small words result in many more possible alternative words. A simple experiment was carried out and the results confirmed this.

In a dictionary of 110,000 phonemic described words, all possible substitution errors of the kind "substituting one phoneme for another" were made. The consequence was that some words became the same. For example, substituting the phoneme "hh" of the word "hh ou s

e" (house) by "m" caused a new valid word "m ou s e". All these new valid words were counted and the number of the newly produced words was recorded, according to the length of the words measured in phonemes. The algorihm is demonstrated below.

$$
\begin{array}{l}
c(k) = 0 \quad for\ k = 1,\ldots,\infty \\
for\ i = 1,\ldots,|D|\ do \\
\quad for\ k = 1,\ldots,|w_i|\ do \\
\quad\quad for\ j = 1,\ldots,|P|\ do \\
\quad\quad\quad if\ w_i(k) \neq p_j\ then \\
\quad\quad\quad\quad w_i(k) = p_j \\
\quad\quad\quad\quad if\ w_i \in D\ then\ c(k) = c(k) + 1\ endif \\
\quad\quad\quad endif \\
\quad\quad done \\
\quad done \\
done
\end{array}
\qquad \textbf{(7.13)}
$$

$c(k)$ ≡ *counted number of valid words per word length k in phonemes*

$|D|$ ≡ *number of words contained in investigated dictionary D*

$|w_i|$ ≡ *number of phonemes per word* w_i

$|P|$ ≡ *number of phonemes contained in investigated phoneme set P*

p_j ≡ *jth phoneme of phoneme set P*

w_i ≡ *ith word of dictionary D*

$w_i(k)$ = *kth phoneme of word* w_i

Figure 7.2 shows the results of the experiment.

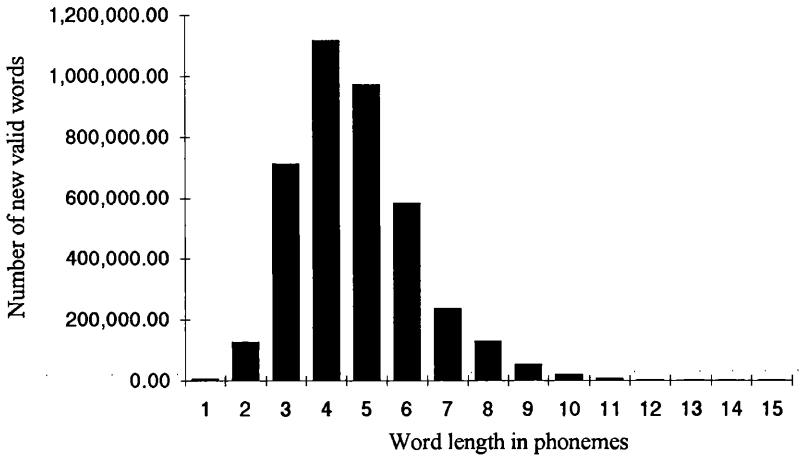


Figure 7.2: Histogram showing the number of new valid words produced by substitution errors, according to word length.

Figure 7.2 shows that the shortest words up to phoneme length 3, like the longer words with the length 8 and 9, cause relatively few new words. However, the difference in occurrence frequencies between words of different length must be considered.

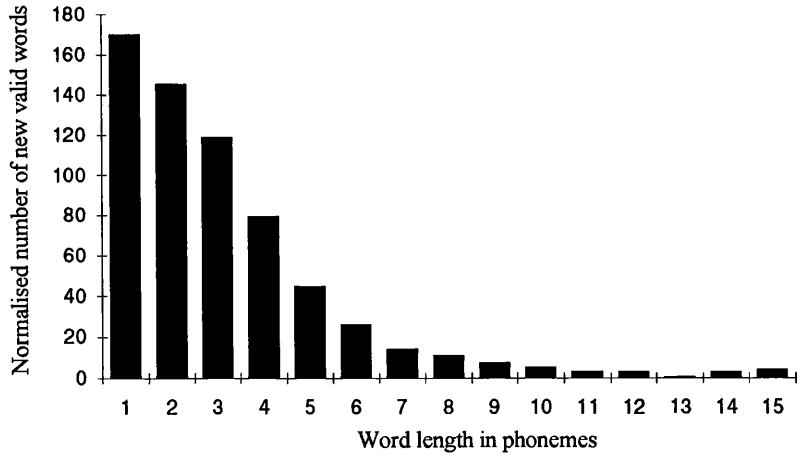


Figure 7.3: Histogram showing the normalised number of new valid words produced by substitution errors, according to word length.

Thus, to obtain more realistic results, the number of newly produced words should be normalised by the occurrence frequency of words with the corresponding word length. Figure 7.3 shows the normalised results. The occurrence frequencies were calculated using the same dictionary. It is clear that, based upon substitution errors for long words, fewer new

words were produced than for short words. Thus, long words have a much better chance of being recognised than shorter ones.

Admittedly, recognition using string-matching was found to be poor in this experiment. Nevertheless, it was expected that nothing would be recognised and so the results were significant, considering that the poorest Language Model was used, the dictionary size was modest, and the evaluation was carried out in a real environment with many disadvantages like computer noise. It could be argued that this experiment works only with single words, but extending it to speaking word sequences, with only the constraint of speaking with a pause of 200 ms between words, poses no problem. Pauses can be detected and the sequence of words can then be split and treated by isolated word recogniser. These speech recognisers are also known as discrete word recognisers, and many commercial large vocabulary speech recognition systems work like recognisers.

These results demonstrate that it is feasible to achieve much better results based on this approach by improving some conditions. For example, the size of the phoneme set could be reduced which would lead to a loss of precision but it would be more likely that the correct word would be among the N-best recognised words. Another more radical improvement would be to use a more robust unit than the phoneme, e.g., triphones or syllables, with the drawback of having a greater number of HMMs.

Assuming that the recognition rate were more robust, this approach would be highly appropriate as a speech interface for a text-based Information Retrieval system. The list of the N-best recognised words fits in with the concept of modern Information Retrieval systems.

First, the stemming process of IR systems often considerably reduces the N-best list. Table 7.2 shows the 15-best recognised words and their stemmed forms for the spoken word university. The Porter algorithm [62] has been used for stemming. The stemmed form "univers" appears three times, which can be regarded as a reduction of the list.

Second, the N-best list supplies weights or, by using Equation (7.10), probabilities for the recognised words. These weights or probabilities can be directly passed to the IR system as term weights or term probabilities. This transfer of weights is appropriate since the best recognised words should get higher priority in the retrieval task than the last recognised words. However, the last recognised words are also considered and retrieval results (usually a list of ranked document references), corresponding to the last recognised words, are also presented, although with lower ranks. The probability given by Equation (7.10) can be used for assessing the belief in a query term in the sense of the probability that the user has spoken a certain term. Probabilistic IR-engines can directly use P(term) for calculating the probabilistic ranking of the documents. For example, P(term) is independent from the relevance term probabilities $P(term_{Rel})$ obtained by the probabilistic IR assessment, so P(term) can just be multiplied with $P(term_{Rel})$. P(term) can just be summed up (they are mutually exclusive) where several occurrences of the same word are found, due to stemming. Consequently P(univers) would increase as in Table 7.2. P(term) could be applied to the well-known probabilistic IR weighting formula F4 [63] by just adding the logarithm of P(term):

$$g(\underline{term}) = \sum_{i=1}^{n} \{term_i (\log\ relevance(term_i) + \log\ P(term_i))\} + constant \qquad (7.14)$$

where $\underline{term}$ expresses a document as a binary vector, $\underline{term} = (term_1,...,term_n)$ where $term_i = 0/1$

denotes absence/presence of the ith index term with respect to the query. The function relevance($term_i$) indicates the probabilistic IR relevance assesment of the ith index term. The application of P(term) to probabilistic IR has not been assessed in this work.

Third, the assessment of the retrieval results by the user provides relevance feedback information which helps implicitly to locate the correct words.

In addition, tools like word-sense-disambiguators and semantic-cooccurrence filters can be applied to reduce the uncertainty of the N-best word list.

In conclusion, the inherent properties of the IR system could help to reduce the uncertainty of the recogniser in a way which is transparent to the user.

Rank	Recognised word	Weight	Stemmed word
1	university	44	**univers**
2	unanimity	54	unanim
3	curiosity	58	curios
4	universality	58	**univers**
5	university-wide	61	**univers** wide
6	complicity	64	complic
7	analyticity	65	analyt
8	unrealistic	68	unrealist
9	eccentricity	68	eccentr
10	electricity	71	electr
11	explicitly	71	explicitli
12	consistently	72	consist
13	possibility	74	possibl
14	uni-directional	74	uni direc
15	instability	75	instabl

Table 7.2: Impact of stemming on the 15-best word list for the spoken word "university".

7.4 Word-Pair Modelling

A simple way of Language Modeling is to explicitly fix the valid sequence of linguistic units. The Language Model simplifies to:

$$P(w_j|w_i) = \begin{cases} k & \text{if } w_j \text{ is a vaild successor of } w_i, \quad k \leq 1 \\ 0 & \text{else} \end{cases} \qquad \textbf{(7.15)}$$

where k is a constant. Equation (7.15) fixes the valid transitions in a finite state network of linguistic units.

A simple example of word-pair modelling is a digit recognition task. Such tasks were one of the earliest commercial applications of speech recognition, for example, the automatic call router Amtrac of AT&T [64], which was established in 1982. Amtrac was an automatic operator which was able to recognise telephone numbers spoken as a sequence of digits with short pauses between them. This application used the template-based approach (one template for each digit) but even so reached a performance of 93%. The more promising HMM approach should outperform this application. The word-pair model will be used to set up a simple Language Model for the ten digits based on phonemes. Figure 7.4 shows the Language Model of the ten digit task. The individual digits can be regarded as non-terminal linguistic units. They are made by concatenating the corresponding phoneme HMMs, as stated for the digit "one". The silence HMM is used to allow natural pauses between the digit utterances. The closed loop, back from the final state to the initial state, allows theoretically infinitely long digit sequences. The word-pair probability between the digits, including the silence symbol, is just 1/11 and, for example, between the phonemes "w" and "ah" of digit "one" just 1. The evaluation has been carried out in the evaluation mode and the general mode of the Speech Recognition Evaluation Interface. A serious evaluation of digit sequences is only made possible by having a corresponding database. Therefore, the evaluation has only been recorded for single spoken digits, since it is easier to get a representative number of trials. Several speakers were used for the evaluation, although one speaker did most of the trials. The amount of used test data and the recognition results are presented in appendix F. They show that digit recognition is basically no problem with a set of 48 phoneme HMMs trained on 6 hours speech. Spoken digit sequences can also be handled with a high recognition accuracy. The accuracy of spoken digit sequences decreases when the coarticulation between the words increases, i.e. when the digit sequence is spoken faster. The recognition time was about 2-3 times the duration of the digit utterance. The recognition time is, in general, a function of the number of HMM states (N) and the duration of the utterance (t). The proportion is about $O(N^2t)$ according to the Viterbi recognition algorithm, see section 4.7.3. Although the number of states is relatively low for the digit model, the recognition time increases noticeably as the number of explicit word-pair models increases. So, large vocabulary systems, whether based on word-pair models or on N-gram models, have not

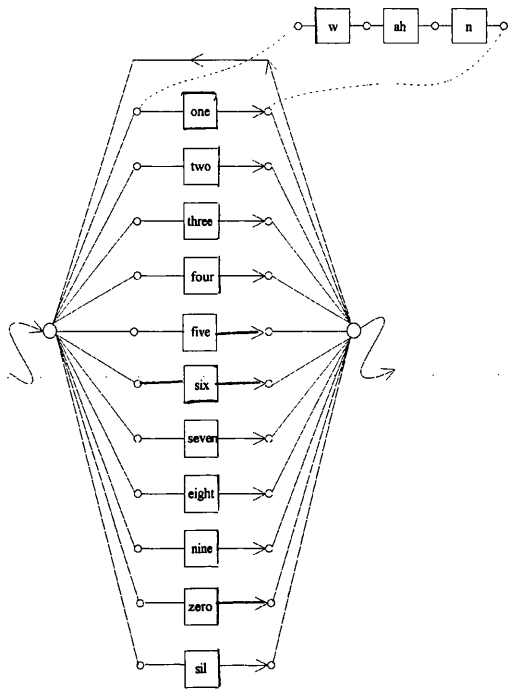


Figure 7.4. Language Model for the ten digit recognition task.

worked in real time up to now. The phoneme-in-parallel network provides the minimum number of states, which would mean relatively good speed, but its Language Model is too poor for large vocabulary systems.

Word-pair modelling can be applied to a finite vocabulary of arbitrary words in the same way as the digit model of Figure 7.4, where words are also just arranged in parallel. This has been done with a finite set of 58 words. They have been arranged as in Figure 7.4. Therefore, arbitrary sentences can be formed by this list of words, as well as sentences which make no syntactic or semantic sense. This list of words has been tested in the same manner as the digits (just single spoken words). Again extending sequences of words is no problem when the breaks between the words are clearly uttered. If the words are uttered coherently, the performance decreases. For example, "dd uw sil y uw" (do you) is often uttered as "dd y uw" in colloquial speech. In this case the recogniser would have trouble because "dd y uw" is, in terms of speech recognition, a new word, and the two models "dd uw" and "y uw" cannot catch this new word. Such word coarticulation effects are explicitly modelled in large vocabulary systems. In these experiments such effects are not considered. Appendix F shows the amount of used test data and the recognition results for single spoken words. They are quite promising, despite the fact that the vocabulary is limited. They show that it is feasible to carry out speaker-independent recognition tasks with small vocabulary, for example, voice command applications and so on. However, as the vocabulary size grows, the recognition performance goes down. Tests have been made with a vocabulary size of 100 words. It still worked for most of the words, but the performance decreased considerably for short words.

It depends, obviously, on the kind of words. In general, long words are easier to recognise than short words. The string-matching experiment was an extreme example of this fact. The simple explanation for the fact that the arrangement of the words in parallel works only up to a modest vocabulary size, is again the a priori language knowledge provided. That each word can follow each other word with equal likelihood is not natural. All linguists, whether behaviorists or Chomskyians, agree that, primitively speaking, the arrangement of the words is driven by a language knowledge ability and is not done arbitrarily. This language knowledge is often subdivided into syntax, semantics, and pragmatics. It is, therefore, also plausible that humans use this ability for recognising speech. The automatic speech recognition system, which is basically a poor approximation of the human speech recogniser, needs that ability, too. Thus, the arrangement of the words in parallel is a poor Language Model. In contrast, a strong Language Model, which incorporates nearly all syntactic and semantic knowledge, would be to model each possible utterance explicitly. This is practically impossible; one reason is the recognition time, considering $O(N^2t)$. Nevertheless, word-pair modeling can easily be used, for example, to form complete sentences explicitly. This is done by using the same vocabulary as used above. The sentences are arranged in parallel as in Figure 7.4. Thus, each sentence can be recognised with equal likelihood from the view of the Language Model. This provides two advantages. The first advantage is that a sentence model can simply be regarded as a long word and is consequently easier to recognise. The second advantage is that there are much less competing recognition hypotheses since only valid word sequences are modelled; this was also the implicit assumption of the training data. Appendix F presents the recognition performance of explicitly modelled sentences.

7.5 Conclusion

This chapter has investigated the most important Language Models. It has been shown that bigram modelling on the phoneme level has increased the recognition performance, where unigram modelling brought only little improvement. In general, N-gram modelling can be applied beyond the phoneme level in the same way. A phoneme-based acoustic front-end and trigram, or tetragram modelling on the word level, makes up the basis of a large vocabulary system. The string-matching experiment is certainly far from perfect. Despite this and considering the weak but flexible Language Model, the recognition of long words demonstrated that it is worthwhile pursuing this approach. It is particularly appropriate for target applications which can deal with some uncertainty like, Information Retrieval systems. Word-pair modelling is a simple but powerful technique for carrying out simple recognition tasks. It has been shown that the 48 phoneme HMMs can be applied to various speaker-independent recognition tasks with satisfactory performance.

Chapter 8

Conclusions

Speech recognition is currently a highly active research area. During this MSc research, enormous advances have been made. For example, in mid 1994, wordspotting was the state-of-the-art speech recognition technique for target applications where some noise (speech recognition errors) can be admitted, like text-based Information Retrieval systems. When this MSc work was started, it was not quite clear whether wordspotting should have been used, or even a recognition system based on words as subword units. A speech recognition system based on words as subword units could only have been built for a modest number of words, which would be of practically no use as an interface for an IR system. Designing a speech interface with words as subword units can be done for a vocabulary of, say, between 500-1000 words, if a suitable database can be provided. This is no longer a big challenge; only the speech database for the HMM training must be provided. Research has also shown that wordspotting can only be carried out with a modest vocabulary of about 100 words. Towards the end of this MSc, it seems that there has been a breakthrough for large vocabulary systems with a vocabulary size of about 100,000 words. The conclusion is that it no longer makes sense to pursue other techniques like wordspotting for tasks where a large vocabulary is needed. Instead, the application of a large vocabulary component as speech interface to a text-based IR system is recommended. A large vocabulary speech recognition system will soon be a standard component of desktop computers.

However, there are still many challenges in speech recognition research. One big challenge is open vocabulary recognition. Today's large vocabulary recognition systems are, on the one hand, based on phoneme-like subword units but, on the other hand, a high recognition performance still cannot be achieved without an explicit Language Model for each word. That is the reason why the words must be known in advance. Therefore, words which are not explicitly modelled, so-called out-of-vocabulary words, are simply not recognised, which may cause inconvenience to the user. This must be taken into consideration in assessments of speech recognition systems outside the speech research community. Some assessments exaggerate often the out-of-vocabulary problem, by not appropriately weighting the 100,000 words which can be recognised. They also do not consider that the average vocabulary of the educated adult English speaker is limited to about 50,000 words [41]. Typically, domain dependent words are the source of most out-of-vocabulary words in speech recognition systems. However, the main difference between an automatic speech recognition system and

a human is that the human can understand the speech pattern of a word without knowing the actual meaning of the word. She can repeat the unknown word, she can also roughly conclude the phonemic transcription, and can use a dictionary to find out the actual meaning of the utterance. This could be the direction for a design of an "open vocabulary" speech recognition system.

This approach was based on phonemes as the basic linguistic unit, which have been modelled by Hidden Markov Models. There is no doubt that phonemes can be applied to many recognition tasks. However, they have limitations; in particular if the linguistic units modelled by phonemes become smaller, the recognition performance degrades. The borderline case is the zerogram model where each phoneme is equally likely and thus the a priori language knowledge provided is a minimum. The evaluation of the zerogram model has shown that the acoustic model based on phonemes is too weak by itself. Therefore, strong support from the Language Model is needed. Certainly, the coarticulation effect is responsible for the weakness of the zerogram model. Phonemes are highly context sensitive and it is not natural that each phoneme can follow each other phoneme equal likelihood. This suggests that the potential of other subword units should be investigated, although main-stream research relies on phonemes or phoneme-like units like triphones. For example, some linguistic evidence indicates that the syllable is the basic linguistic unit of spoken English and many other languages. Syllables show clear boundaries in the spectrum of the speech signal. Coarticulation of syllables is much lower than for phonemes. Syllables always have a vowel or a vowel-like unit as their core, so that there is a certain signal energy level guaranteed. Conversely, however, there are many more syllables than phonemes for a given language. Thus, there would be more Hidden Markov Models needed and consequently more training data would also be required. Nevertheless, the number of syllables is much below the word number; for English, a figure of about 5000 is a good conservative guess. This syllable core set is vocabulary-independent, once you have well trained syllable models. Therefore, new words which do not yet exist can be modelled in the future. Furthermore, if the recognition performance of the syllable-based acoustic front-end were stable, new words could even at be generated at runtime, maybe with some confirmation from the user. This is analogous to the fact that we can recognise the sound structure of words (syllables) without knowing the actual content. With regard to speech interfaces for text-based Information Retrieval systems, the string-matching experiment has shown some surprising results. This research could be continued to show how far this approach could be optimised. This approach fits, in particular, with probabilistic Information Retrieval. The problem with such research is again the competition of large vocabulary systems.

Finally, I will summarise what has been contributed by this research. A literature survey of the state-of-the-art of speech recognition research was presented. The potential for doing speech recognition research starting from the bottom was pointed out. A speech interface for text-based Information Retrieval systems constituted the target application, and thus gave a goal to the work. A realistic strategy was chosen which allows for incorporation of speaker-independent large vocabulary tasks. A major decision was to use the phoneme as the basic linguistic unit and to model the phonemes by Hidden Markov Models. The construction of these phoneme models was investigated. The HTK toolkit provided the algorithmic framework for training and evaluation of the models. So far, however, there have been no guidelines

given as to how these models should be set up. Experiments and literature studies were used to set up the huge parameter space of Hidden Markov Model based speech recognition. A suitable training database was chosen for model training. The result was a set of 48 phoneme models which was called the acoustic front-end. A tool was designed and implemented to allow the interactive evaluation of the acoustic front-end in a real time, live environment. This tool also demonstrates which recognition tasks can be accomplished with phonemes and different Language Models were applied. Several Language Models were investigated and some experiments with word and sentence recognition were undertaken.

The general emphasis of this research was not on an ad hoc solution, but rather treating each step and related research area with respect. It is important that, in times of "plug and play", Computing Scientists are not seduced by this mentality.

Appendix A

Phonemic Alphabets

IPA: English part of the International Phonetic Alphabet established in 1988.

ARPAbet: Alphabet defined by the United States Advanced Research Projects Agency in two version: (1) single and lower case letter set and (2) double and upper case set.

p48: Alphabet used in the approach described in this thesis.

CMU: Alphabet used at Carnegie Mellon University (CMU), in particular in the 110,000 phonemic described lexicon provided by CMU.

IPA	ARPAbet		p48	CM	Examples	Articulation type
i	i	IY	iy	IY	heed, beat, eat	Continuant,vowel, front
ɪ	I	IH	ih	IH	hid, bit, it	Continuant,vowel, front
e	e	EY	ey	EY	hayed, bait, ate	Continuant,vowel, front
ɛ	E	EH	eh	EH	head, bet	Continuant,vowel, front
æ	@	AE	ae	AE	had, bat, at	Continuant,vowel, front
Y	a	AA	aa	AA	hod, cot, odd	Continuant,vowel, back
]	c	AO	ao	AO	hawed, about, ought	Continuant,vowel, back
o	o	OW	ow	OW	hoed, boat, oat	Continuant,vowel, back
ʊ	U	UH	uh	UH	hood, book	Continuant,vowel, back
u	u	UW	uw	UW	who'd, boot, two	Continuant,vowel, back
ɡ	R	ER	er	ER	heard, bird, hurt	Continuant,vowel, mid
c	x	AX	ax		ago, the	Continuant,vowel, mid
ʌ	A	AH	ah	AH	mud, butt, hut	Continuant,vowel, mid
Yɪ	Y	AY	ay	AY	hide, bite	Noncontinuant, Diphthong
Yu	W	AW	aw	AW	how'd, bough, cow	Noncontinuant, Diphthong
]ɪ	O	OY	oy	OY	boy, toy	Noncontinuant, Diphthong
v	X	IX	ix		roses	Continuant,vowel, mid
p	p	P	p	P	pea, pop	Noncontinuant, consonant, stop, unvoiced
b	b	B	b	B	bat, bob, be	Noncontinuant, consonant, stop, voiced
t	t	T	t	T	tea, tot	Noncontinuant, consonant, stop, unvoiced
d	d	D	d	D	deep, dad, dee	Noncontinuant, consonant, stop, voiced

IPA	ARPAbet		p48	CM	Examples	Articulation type
k	k	K	k	K	kick, key	Noncontinuant, consonant, stop, unvoiced
g	g	G	g	G	go, gag, green	Noncontinuant, consonant, stop, voiced
f	f	F	f	F	five, fief, fee	Continuant, consonant, fricative, unvoiced
v	v	V	v	V	vice, very, vee	Continuant, consonant, fricative, voiced
θ	T	TH	th	TH	thing, thief, theta	Continuant, consonant, fricative,unvoiced
ð	D	DH	dh	DH	then, they, thee	Continuant, consonant, fricative, voiced
s	s	S	s	S	so, sis, sea	Continuant, consonant, fricative, unvoiced
z	z	Z	z	Z	zebra, zoo, zee	Continuant, consonant, fricative, voiced
ʃ	S	SH	sh	SH	show, shoe, she	Continuant, consonant, fricative, unvoiced
f	Z	ZH	zh	ZH	measure, seizure	Continuant, consonant, fricative, voiced
h	h	HH	hh	HH	help, hay, he	Continuant, consonant, whisper
m	m	M	m	M	mom, me	Continuant, consonant, nasal
n	n	N	n	N	noon, non, knee	Continuant, consonant, nasal
ŋ	G	NX	ng	NG	sing, ping	Continuant, consonant, nasal
l	l	L	l	L	love, led, lee	Noncontinuant, semivowel, liquid
l	L	EL	el		cattle, bottle	Noncontinuant, semivowel,
m	M	EM			some	Continuant, consonant, fricative, voiced
n	N	EN	en		son, button	Continuant, consonant, fricative, voiced
f	F	DX	dx		batter, butter	Noncontinuant, consonant, stop, voiced
§	Q	Q				Glotal stop
w	w	W	w	W	want, wet, we	Noncontinuant, semivowel, glide
j	y	Y	y	Y	yard, yet, yield	Noncontinuant, semivowel, glide
r	r	R	r	R	race, red, read	Noncontinuant, semivowel, liquid
t	C	CH	ch	CH	church, cheese	Continuant, consonant, affricate
df	J	JH	jh	JH	just, judge, gee	Continuant, consonant, affricate
M	H	WH			when	
			cl			Unvoiced closure
			vcl			Voiced closure
			epi			Epinthetic closure
			sil			Silence

Table A.1: Table of Phonemic alphabets.

Appendix B

Organisation of Training and Recognition Environment

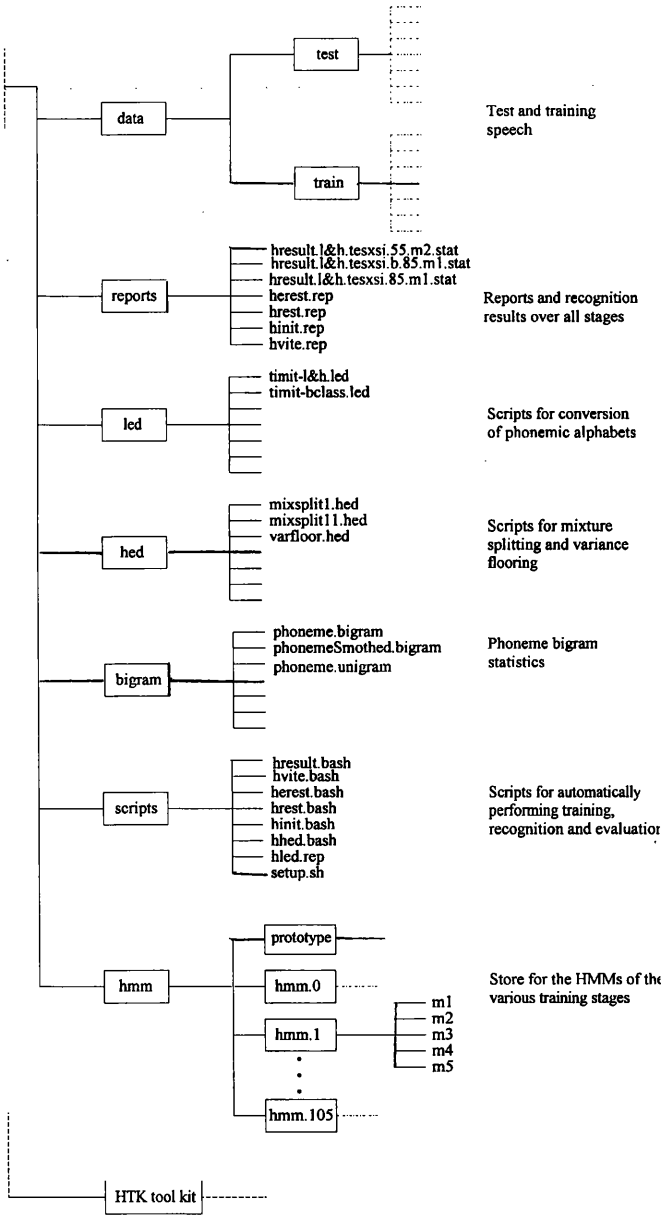


Figure B.1: File structure reflecting the organisation of the training, recognition and evaluation environment for building phoneme HMMs.

Appendix C

Detailed Evaluations of the Acoustic Front-End

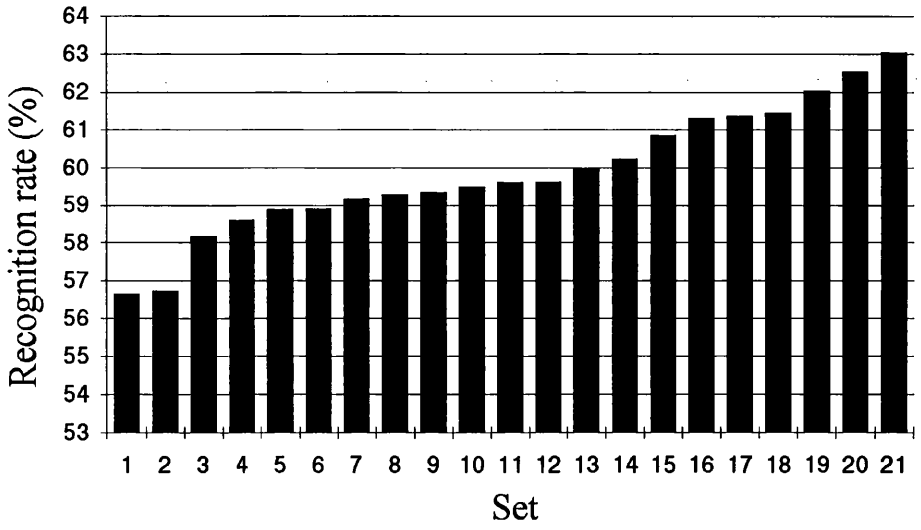


Figure C.1: Recognition rate measured over randomly sorted subsets of the TIMIT test set.

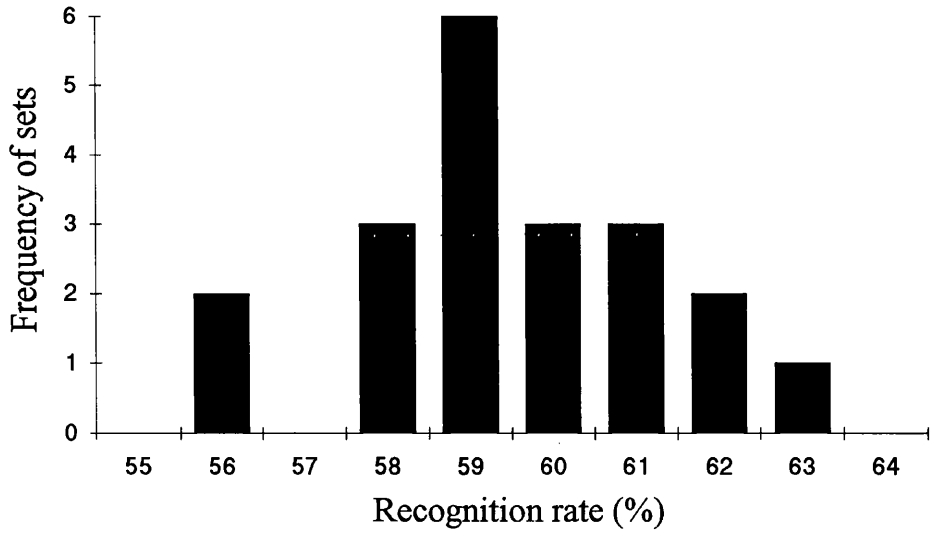


Figure C.2: Histogram of the recognition rate over randomly sorted subsets of the TIMIT test set. The average value is 59.89 %, with a standard deviation of 1.7 %.

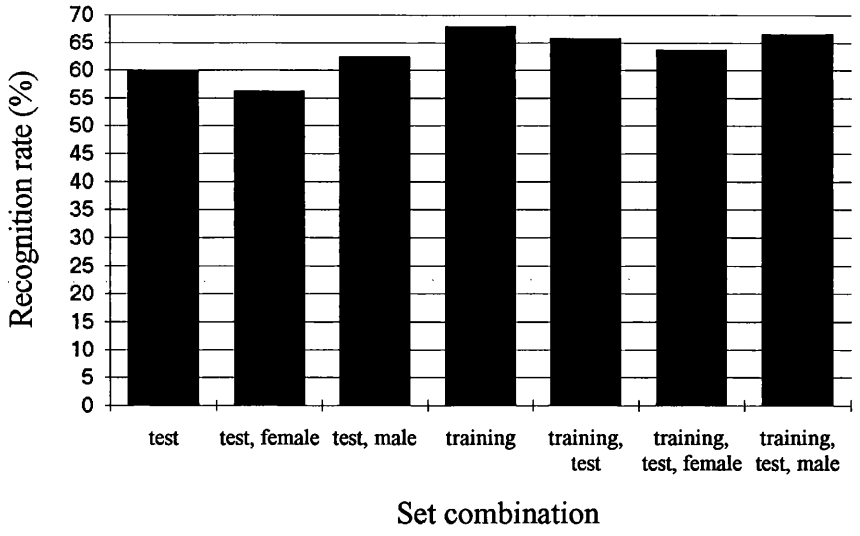


Figure C.3: Recognition rate over various set combinations of the TIMIT database.

Set combination	Test	Training	Test and female	Test and male	Training test and female	Training test and male
Rec. Rate	59.85 %	67.90 %	55.72 %	61.96 %	63.74 %	66.63 %
Rec. Accuracy	47.52%	52.82%	41.90 %	50.38 %	48.18 %	52.85 %
Total Number N	50754	140225	17130	33624	58582	130967
Hits H	30377	95209	9544	20833	37342	87265
Substitutions S	16447	37341	6286	10161	17921	35496
Deletions D	3930	7675	1300	2630	3319	8206
Insertions I	6259	21118	2366	3893	9120	18054

Table C.1: Recognition rate and accuracy over various set combinations of the TIMIT database.

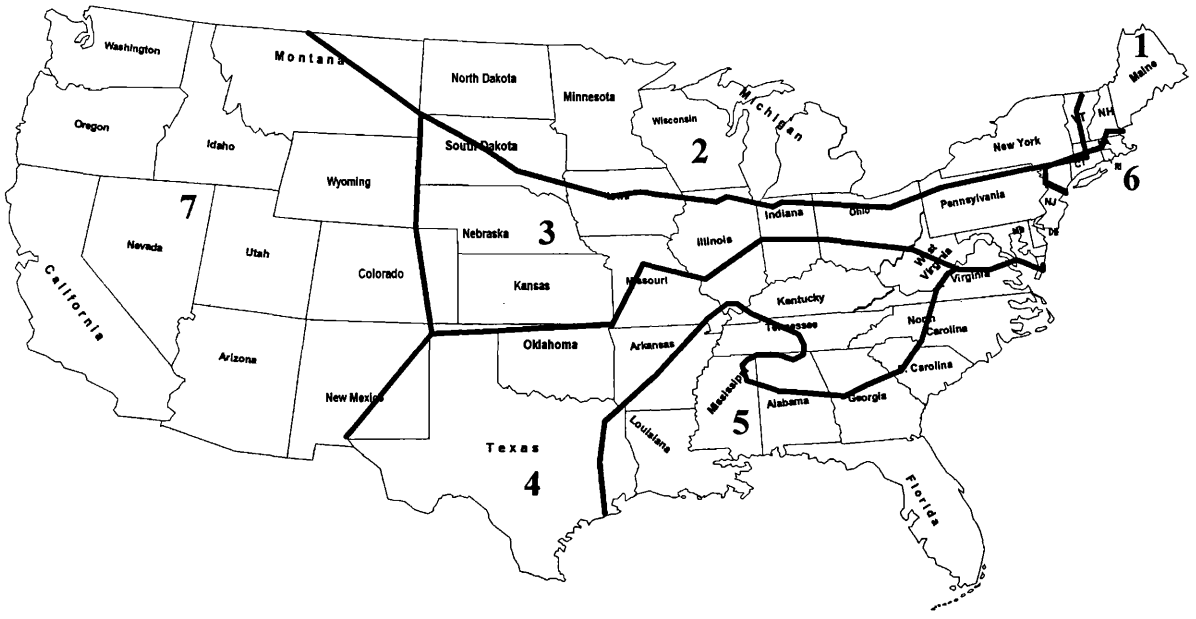


Figure C.4: Dialect regions showing the origin of the TIMIT database speakers.

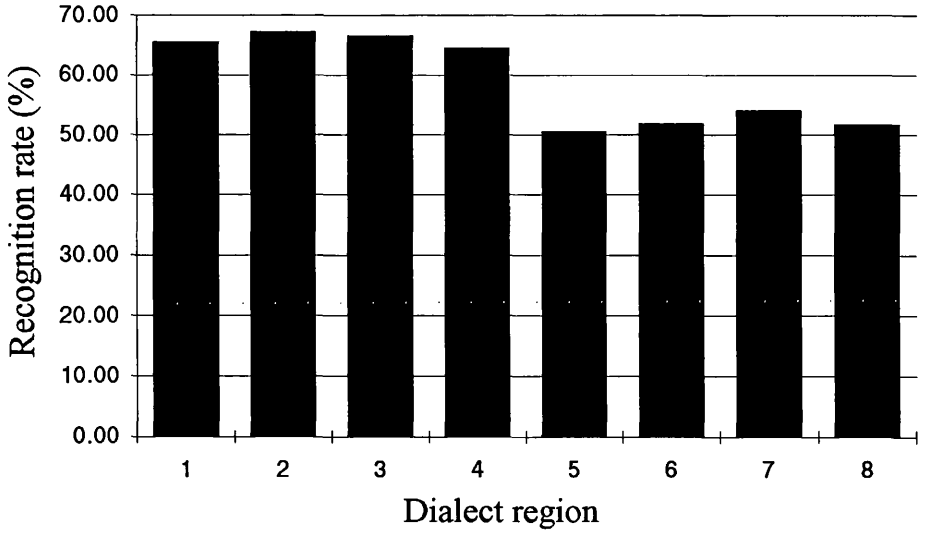


Figure C.5. Recognition rate evaluated for the different dialect regions of the TIMIT database; only speech from the test set has been used.

Appendix D

Statistics for N-Gram Modelling

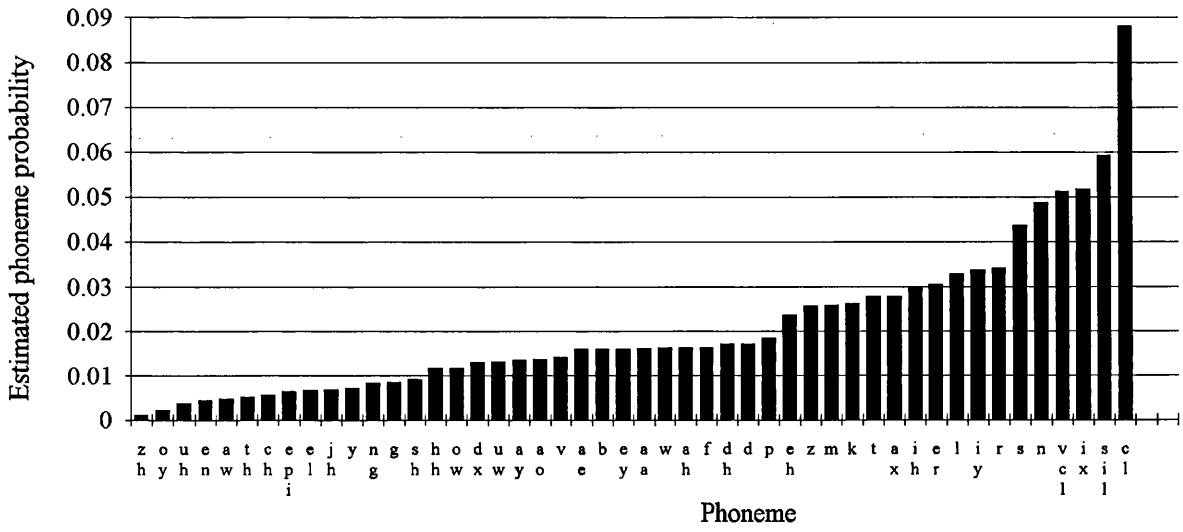


Figure D.1: Phoneme occurrence probabilities estimated over the TIMIT database, using unigram modelling.

Appendix E

Results of the String-Matching Experiment

The heading is the word which was spoken. The trials are laid out horizontally. The first string in the list is the recognised phoneme sequence. The 15 best strings are listed below the phoneme sequence, each with the string-matching score.

University

y uw n iy v ih ax s ix jh iy v	y uw n iy v ih l z ix t iy	y ax n y v uh ah s ix jh iy v	y uw n iy v ah s ix jh iy v	dh y uw n iy dh ih l ax s ix jh
1 university 44	**1 university 37**	**1 university 54**	**1 university 37**	1 uni-directional 82
2 unanimity 58	2 universality 41	2 universality 68	2 universality 51	2 universality 82
3 universality 58	3 unanimity 47	3 university-wide 71	3 university-wide 54	3 eccentricity 88
4 university-wide 61	4 humidity 48	4 curiosity 72	4 curiosity 61	4 university-wide 89
5 uni-directional 68	5 mentality 54	5 complicity 74	5 community 66	5 magnificently 91
6 conceivably 68	6 community 56	6 unconvincing 75	6 unanimity 67	6 consistently 92
7 curiosity 68	7 possibility 57	7 miraculously 75	7 grievances 68	7 continuously 92
8 complicity 74	8 divinities 58	8 complexity 77	8 convincing 68	8 ridiculously 94
9 analyticity 75	9 instability 58	9 conceivably 78	9 curiously 68	9 probabilities 98
10 unconvincing 75	10 utilizing 58	10 confronted 78	10 humidity 68	10 applicability 98
11 unrealistic 78	11 university-wide 58	11 unjustified 78	11 furiously 68	11 flexibility 98
12 influences 78	12 curiosity 61	12 contentedly 81	12 unjustified 71	12 eventualities 98
13 conferences 78	13 triviality 61	13 construction 81	13 uni-directional 71	13 consolidation 102
14 eccentricity 78	14 viscosity 62	14 analyticity 81	14 conceivably 71	14 configuration 102
15 undeniably 78	15 velocities 62	15 electricity 81	15 uniforms 72	15 responsibility 102

y uw n iy dh ih ah s ix t iy v	y uw n iy v uw ax s ix jh iy dh b	y uw n iy v ih ax z ix th jh iy v	y uw n y v uw ah s ix jh hh iy	y uw n y v uw ah s ix jh hh iy
1 university 44	1 university-wide 64	1 universality 71	**1 university 44**	**1 university 44**
2 unanimity 54	2 universality 65	2 uni-directional 75	2 universality 58	2 universality 58
3 curiosity 58	3 uni-directional 75	3 university-wide 78	3 university-wide 65	3 university-wide 65
4 universality 58	4 continuously 85	4 considerably 88	4 unanimity 74	4 unanimity 74
5 university-wide 61	5 analyticity 88	5 consistently 91	5 influences 78	5 influences 78
6 complicity 64	6 eventualities 91	6 instability 92	6 unjustified 78	6 unjustified 78
7 analyticity 65	7 instability 92	7 unconvincing 92	7 curiosity 78	7 curiosity 78
8 unrealistic 68	8 unconvincing 92	8 astonishingly 92	8 analyticity 81	8 analyticity 81
9 eccentricity 68	9 archeological 94	9 analyticity 92	9 continuously 82	9 continuously 82
10 electricity 71	10 considerably 94	10 prerequisite 92	10 constantly 82	10 constantly 82
11 explicitly 71	11 electricity 94	11 possibilities 94	11 ambiguity 82	11 ambiguity 82
12 consistently 72	12 complexity 94	12 explicitly 94	12 conclusions 84	12 conclusions 84
13 possibility 74	13 investigation 95	13 contentedly 94	13 conferences 84	13 conferences 84
14 uni-directional 74	14 eccentricity 95	14 non-supervisory 95	14 unspecified 84	14 unspecified 84
15 instability 75	15 non-supervisory 95	15 eccentricity 95	15 complicity 84	15 complicity 84

Administration

f eh th hh iy n iy s dh uh iy zh ix dh th	f eh th t dh iy n iy z t dh ey sh dh v	eh iy n iy z dh ix iy zh ix v en b b	th ae th ax m iy n iy s th g ih iy zh ch iy v	f ih th dh iy ng iy z dh uh iy zh ix dh
1 resin-saturated 105	1 representation 92	1 uni-directional 104	1 quarter-century-old 142	1 bibliographies 104
2 representation 112	2 administrative 104	2 representation 105	2 contradictorily 146	2 inarticulate 104
3 administration 114	3 reorganization 104	3 generalizations 106	3 -	3 unappreciated 108
4 elementary-school 116	**4 administration 104**	4 unfortunately 108	4 -	4 idiosyncrasies 109
5 responsibility 117	5 interpretations 108	5 eventualities 110	5 -	5 representation 111
6 idiosyncrasies 118	6 demineralization 112	6 unappreciated 110	6 -	6 sophistication 112
7 self-sufficiency 118	7 hospitalization 112	7 eccentricity 110	7 -	7 sophisticated 112
8 electrostatic 118	8 rehabilitation 114	8 magnificently 111	8 -	8 unenthusiastic 113
9 self-confident 118	9 investigations 116	9 unquestionably 111	9 -	9 reupholstering 114
10 predispositions 121	10 electrostatic 118	10 head-in-the-clouds 112	10 -	10 nymphomaniacs 114
11 geocentricism 122	11 fasciculations 118	11 demonstration 112	11 -	11 distributed 114
12 confabulation 122	12 elementary-school 118	12 investigation 112	12 -	12 instantaneous 114
13 unexpectedly 122	13 resin-saturated 121	13 organizations 112	13 -	13 micrometeorite 114
14 university-wide 122	14 micrometeorites 122	14 pronunciation 112	14 -	14 investigation 114
15 understandingly 122	15 confabulation 122	15 reorganization 113	15 -	15 interpretation 114

Taxi

t eh k z iy	t eh k z iy	t ah cl k z iy	t sil t eh k s iy	t ih t s iy
1 tells 24	1 lucky 24	1 technique 38	1 city 24	1 takes 14
2 technique 24	2 husky 27	2 tensely 38	2 pity 24	**2 taxi 17**
3 tellers 27	3 monkey 27	3 texts 38	3 tips 24	3 shaky 24
4 tends 27	**4 taxi 27**	4 texture 38	4 titter 24	4 makes 24
5 text 27	5 touches 27	5 tetanus 44	5 fifty 27	5 takin' 24
6 taxi 27	6 tunnels 27	6 index 44	6 fists 27	6 tapes 24
7 tensely 30	7 uneasy 27	7 jealousy 44	7 gifts 27	7 tax 24
8 stems 31	8 truck 28	8 trunks 44	8 guilty 27	8 racy 24
9 terrier 31	9 stuck 28	9 twenty 44	9 lists 27	9 taking 27
10 turquoise 31	10 study 31	10 techniques 45	10 patsy 27	10 tasks 27
11 techniques 31	11 turquoise 31	11 cutbacks 47	11 tastes 27	11 tastes 27
12 errors 34	12 trucks 31	12 beatniks 47	**12 taxi 27**	12 taken 27
13 heads 34	13 guns 34	13 turnpikes 47	13 tests 27	13 text 27
14 heavy 34	14 lovie 34	14 typically 47	14 tinsel 27	14 tracks 27
15 every 34	15 jerky 34	15 status 48	15 traits 27	15 trucks 27

Meanwhile

dh iy n v ay ey ix

1 convey 34
2 invade 34
3 inviolate 34
4 meaning 38
5 meanness 38
6 vices 38
7 virus 38
8 leaving 38
9 surveying 41
10 naked 42
11 nathan 42
12 vases 42
13 enzyme 44
14 evening 44
15 inside 44

th v iy n b ay ng v b th dh b t v b

1 contributory 118
2 administrative 122
3 radioactivity 122
4 rehabilitation 122
5 university-wide 122
6 magnificently 124
7 nonsystematic 124
8 idiosyncrasies 124
9 contributions 124
10 unenthusiastic 124
11 encyclopedias 124
12 responsibility 125
13 confabulation 128
14 unquestionably 128
15 grade-equivalent 128

th m iy n l ay ey uw hh

1 meanwhile 48
2 mandates 54
3 realizing 54
4 meaningful 57
5 routinely 58
6 someplace 58
7 thirty-five 58
8 ladylike 58
9 commonplace 58
10 misplaced 61
11 symbolize 61
12 mealynosed 61
13 insulate 62
14 machinery 62
15 monopolize 62

th m iy n l ay ey ih

1 moonlight 34
2 meanwhile 41
3 implied 44
4 sunlight 44
5 unlined 44
6 maintain 44
7 three-inch 48
8 midnight 48
9 tomblike 48
10 mayonnaise 48
11 mandates 51
12 ladylike 51
13 realizing 51
14 routinely 51
15 someplace 51

dh iy n uw ay iy uw

1 scenery 34
2 nuclear 41
3 sinewy 41
4 unduly 41
5 nicely 42
6 nightly 42
7 canoes 44
8 candy 44
9 cruelty 44
10 eagerly 44
11 easily 44
12 enemy 44
13 energy 44
14 entire 44
15 county 44

y iy n l ao ey hh v

1 yielded 44
2 slavery 52
3 evenly 52
4 ballyhoo 54
5 betrayed 54
6 belongs 54
7 chlorine 54
8 contain 54
9 chloride 54
10 details 54
11 downpour 54
12 acclaimed 54
13 feelings 54
14 fjords 54
15 detailed 54

th m iy uw l ay ey f v

1 thirty-five 48
2 realizing 54
3 delightful 54
4 meanwhile 58
5 medieval 58
6 himself 58
7 shoelaces 58
8 ladylike 58
9 someplace 58
10 mealynosed 61
11 meaningful 61
12 misplaced 61
13 symbolize 61
14 stimulates 61
15 improved 62

m y n m ay iy ih b

1 moonlight 44
2 music 44
3 museum 47
4 mutineer 47
5 managed 48
6 unity 48
7 midnight 48
8 anything 48
9 mannerism 51
10 meanwhile 51
11 memorized 51
12 cornmeal 51
13 binomial 51
14 amoebas 52
15 myopia 52

0 f iy l ay ey f v

1 feeling 34
2 female 34
3 fields 34
4 flying 34
5 realize 34
6 feelings 37
7 layoffs 38
8 freeway 38
9 freely 38
10 delightful 40
11 fieldwork 40
12 realizing 40
13 skywave 41
14 lively 42
15 climbed 44

hh iy n dh uh ay ey uw th

1 meanwhile 54
2 henceforth 57
3 handful 58
4 endurance 64
5 contained 64
6 container 64
7 entirely 64
8 contains 64
9 beefsteak 64
10 handling 64
11 hesitate 64
12 holidays 64
13 honeybees 64
14 horseplay 64
15 confines 64

Information

iy n ih z l m iy sh ix ih v

1 information 64
2 whiplashes 64
3 relationship 65
4 informative 67
5 resolution 68
6 unlimited 68
7 alternative 68
8 realization 68
9 irradiation 68
10 infiltration 68
11 salesmanship 71
12 confirmation 71
13 consumption 71
14 executive 71
15 conditions 72

th g iy n ih f dh l ao m iy zh

1 nymphomaniacs 96
2 magnificently 97
3 nonsystematic 101
4 configuration 104
5 consolidation 104
6 eventualities 104
7 rehabilitation 105
8 antagonistic 106
9 unfortunately 106
10 articulation 108
11 pronunciation 108
12 unappreciated 108
13 contributory 109
14 confabulation 109
15 consistently 110

ix uw v iy n ix en z ao m iy zh ix ih v

1 informative 68
2 enthusiastic 81
3 negotiations 81
4 competitive 82
5 executive 84
6 salesmanship 84
7 unanimity 86
8 reconnaissance 86
9 citizenship 86
10 inexhaustible 87
11 inventories 88
12 legislative 88
13 panelization 88
14 penetrated 88
15 integration 88

iy n ih v ow m iy sh ix

1 depletion 58
2 revulsion 58
3 information 61
4 invitation 61
5 vocational 62
6 anaesthesia 64
7 conviction 65
8 irradiation 65
9 unlimited 65
10 negotiations 65
11 condition 66
12 initiative 66
13 intuition 66
14 interviewee 68
15 infection 68

dh dh iy n ih th ow m iy zh en v

1 informative 84
2 unanimity 86
3 radioactive 88
4 undeniably 88
5 irremediable 92
6 economically 92
7 utopianism 92
8 nymphomaniacs 92
9 kindergarten 94
10 presumably 94
11 confirmation 94
12 consumption 94
13 experience 94
14 ingredient 94
15 fulfillment 94

Information

t ax iy n ih th ah m iy zh eh n v th

1 veterinarian 92
2 contemporary 94
3 conformists 94
4 pronunciation 95
5 instability 96
6 considerably 96
7 astonishingly 96
8 denunciation 98
9 ever-increasing 98
10 enthusiastic 98
11 one-upmanship 98
12 renunciation 98
13 uncomfortable 98
14 construction 98
15 development 98

iy n ih z ax m iy zh ih n uw v b

1 examination 82
2 unconvincing 84
3 nymphomaniacs 85
4 instruments 88
5 conformational 88
6 fingerprints 88
7 inexhaustible 92
8 predicament 92
9 requirements 92
10 environment 92
11 ingredients 94
12 analyticity 94
13 installations 94
14 understanding 94
15 instability 94

iy n ih v ah m iy zh eh n

1 anaesthesia 54
2 revulsion 54
3 inhuman 64
4 norwegian 64
5 precision 64
6 reduction 64
7 depletion 64
8 consumption 64
9 punishment 65
10 management 65
11 invitation 67
12 information 67
13 nevertheless 67
14 phenomenon 67
15 determined 68

dh iy n ih z ah m ey zh eh n ix

1 confirmation 74
2 consumption 74
3 conformational 77
4 panelization 78
5 examination 79
6 infiltration 81
7 seamstresses 82
8 recommended 82
9 confronted 82
10 preservation 84
11 remembrance 84
12 conclusions 84
13 accusations 84
14 integration 84
15 presumably 84

iy n ih s l ao m iy zh eh n uw

1 historians 67
2 exploring 72
3 explosion 72
4 exclaiming 72
5 historical 74
6 misperceives 74
7 mysterious 74
8 distorted 74
9 unlimited 74
10 anti-slavery 75
11 extremely 76
12 vietnamese 76
13 historically 77
14 necessitates 77
15 inclusions 78

iy n th b ao m iy sh ih m uw

1 informative 71
2 audio-visual 72
3 information 74
4 anatomical 74
5 vietnamese 76
6 unbelievable 77
7 many-bodied 78
8 individual 78
9 uniforms 78
10 unauthentic 78
11 cinematic 78
12 absorption 78
13 enthusiastic 78
14 one-upmanship 78
15 consumption 81

k iy n uw th ah m iy zh eh n uw th

1 conformational 84
2 contemporary 84
3 documented 86
4 construction 88
5 consequences 91
6 constructions 91
7 renunciation 92
8 one-upmanship 92
9 denunciation 92
10 enthusiastic 94
11 conformists 94
12 uncomfortable 94
13 ingredients 94
14 contentedly 96
15 experiment 98

th f iy n uw z v l ao m iy zh ix uw

1 eventualities 98
2 non-supervisory 102
3 consolidation 104
4 continuously 106
5 universality 106
6 nonsystematic 107
7 rehabilitation 107
8 pronunciation 108
9 unappreciated 108
10 unenthusiastic 109
11 unfortunately 110
12 encyclopedias 111
13 university-wide 111
14 investigation 112
15 organizations 112

iy n ih th ow n iy zh ix uw

1 anaesthesia 48
2 loneliness 58
3 antithesis 61
4 increasing 62
5 continue 62
6 increases 62
7 depletion 64
8 misleading 64
9 precision 64
10 proceeding 64
11 reminded 64
12 unbroken 64
13 witnesses 64
14 witnessing 64
15 norwegian 64

iy n ix z ax m iy zh ix ih v

1 ineffective 64
2 information 64
3 interweaving 64
4 competitive 65
5 informative 67
6 cooperative 68
7 anatomical 68
8 cinematic 68
9 confirmation 71
10 incomplete 72
11 inferences 74
12 interesting 74
13 inclusions 74
14 intrusions 74
15 interested 74

iy ng ih z l m iy zh uw v

1 feverishly 62
2 improving 62
3 anaesthesia 64
4 depletion 64
5 misleading 64
6 primitive 64
7 sixty-five 67
8 examined 68
9 resulted 68
10 examples 68
11 audio-visual 69
12 reasonably 70
13 exactly 70
14 presumably 70
15 disapproves 71

iy n ih th ax m iy zh ix uw v

1 information 64
2 interweaving 64
3 mathematics 67
4 antithesis 68
5 disapproval 68
6 unlimited 68
7 disapproves 68
8 confirmation 71
9 considering 72
10 incomplete 72
11 indemnity 74
12 individual 74
13 ineffective 74
14 inferences 74
15 entertaining 74

iy ih z ah m iy zh ix uw b

1 resulted 54
2 examined 58
3 impossible 62
4 depletion 64
5 increases 64
6 increasing 64
7 misleading 64
8 decisions 64
9 reduction 64
10 revulsion 64
11 irremediable 64
12 intrusions 67
13 inclusions 67
14 resistance 67
15 repugnant 67

b iy ih th ah m iy ey sh ix uw v b

1 denunciation 61
2 renunciation 61
3 determination 71
4 depreciation 71
5 consumption 74
6 agglomeration 74
7 associations 74
8 civilization 74
9 confirmation 74
10 pronunciation 74
11 examination 75
12 infuriation 75
13 imagination 76
14 negotiations 77
15 conformational 77

iy dx ih z l m iy sh eh n uw d b

1 inclusions 68
2 irremediable 71
3 explosion 72
4 resistance 74
5 interweaving 74
6 intrusions 74
7 awe-inspiring 74
8 illuminated 75
9 illuminating 75
10 exclaiming 76
11 remembrance 77
12 indemnity 78
13 compliance 78
14 information 78
15 impressions 78

Appendix F

Evaluation of Word-Pair Models

The following lists should be read as follows. The first entry is just an order number. The second entry is the linguistic unit which is a digit, word or sentence. The third entry is again the order number. The fourth entry is the number of recognition trials performed on the linguistic unit. The fifth entry is the number of times the unit has been correctly recognised. The last entry is just the the number of correctly recognised units in percent. All the linguistic units in the list are explicitly modelled by the phoneme HMMs of the p48 set. Word-pair modelling has been applied as the Language Model.

1. ZERO	1.	15	14	93.33 %
2. ONE	2.	14	14	100.00 %
3. TWO	3.	11	11	100.00 %
4. THREE	4.	12	12	100.00 %
5. FOUR	5.	13	7	53.85 %
6. FIVE	6.	19	14	73.68 %
7. SIX	7.	10	10	100.00 %
8. SEVEN	8.	10	10	100.00 %
9. EIGHT	9.	10	10	100.00 %
10. NINE	10.	10	10	100.00 %
11. TEN	11.	10	8	80.00 %

Figure F.1: Recognition performance of explicitly modelled digits.

1. DO YOU WANT SOME COFFEE	1.	10	10	100.00 %
2. HE PLAYS BASEBALL	2.	10	9	90.00 %
3. CALCIUM MAKES BONES AND TEETH STRONG	3.	10	10	100.00 %
4. CLEAR PRONUNCIATION IS APPRECIATED	4.	10	10	100.00 %
5. PIZZERIAS ARE CONVENIENT FOR A QUICK LUNCH	5.	10	10	100.00 %
6. PROJECT DEVELOPMENT WAS PROCEEDING SLOWLY	6.	10	10	100.00 %
7. EVEN A SIMPLE VOCABULARY CONTAINS SYMBOLS	7.	10	9	90.00 %
8. THE BEST WAY TO LEARN IS TO SOLVE EXTRA PROBLEMS	8.	10	10	100.00 %
9. THE FOG PREVENTED THEM FROM ARRIVING ON TIME	9.	10	10	100.00 %
10. I PREFER WINE	10.	10	8	80.00 %
11. SPEECH RECOGNITION IS STILL BASIC RESEARCH	11.	10	8	80.00 %

Figure F.2: Recognition performance of explicitly modelled sentences.

1. SPEECH	1.	3	3	100.00 %	27. CONTAINS	27.	3	2	66.67 %
2. RECOGNITION	2.	3	3	100.00 %	28. SYMBOLS	28.	3	3	100.00 %
3. STILL	3.	4	3	75.00 %	29. PROJECT	29.	3	3	100.00 %
4. BASIC	4.	3	2	66.67 %	30. DEVELOPMENT	30.	3	3	100.00 %
5. RESEARCH	5.	4	2	50.00 %	31. WAS	31.	3	2	66.67 %
6. I	6.	3	3	100.00 %	32. PROCEEDING	32.	3	3	100.00 %
7. PREFER	7.	5	1	20.00 %	33. SLOWLY	33.	4	1	25.00 %
8. WINE	8.	3	0	0.00 %	34. PIZZERIAS	34.	4	3	75.00 %
9. FOG	9.	3	0	0.00 %	35. ARE	35.	3	0	0.00 %
10. PREVENTED	10.	3	2	66.67 %	36. CONVENIENT	36.	4	4	100.00 %
11. THEM	11.	4	1	25.00 %	37. FOR	37.	4	1	25.00 %
12. FROM	12.	3	0	0.00 %	38. A	38.	3	0	0.00 %
13. ARRIVING	13.	3	3	100.00 %	39. QUICK	39.	3	0	0.00 %
14. ON	14.	3	3	100.00 %	40. LUNCH	40.	3	3	100.00 %
15. TIME	15.	3	2	66.67 %	41. CLEAR	41.	3	0	0.00 %
16. THE	16.	3	0	0.00 %	42. PRONUNCIATION	42.	3	3	100.00 %
17. BEST	17.	3	3	100.00 %	43. IS	43.	5	3	60.00 %
18. WAY	18.	3	2	66.67 %	44. APPRECIATED	44.	4	4	100.00 %
19. LEARN	19.	3	1	33.33 %	45. CALCIUM	45.	4	4	100.00 %
20. TO	20.	3	0	0.00 %	46. MAKES	46.	3	3	100.00 %
21. SOLVE	21.	3	0	0.00 %	47. BONES	47.	4	4	100.00 %
22. EXTRA	22.	3	1	33.33 %	48. AND	48.	3	2	66.67 %
23. PROBLEMS	23.	4	3	75.00 %	49. TEETH	49.	3	3	100.00 %
24. EVEN	24.	3	3	100.00 %	50. STRONG	50.	3	2	66.67 %
25. SIMPLE	25.	3	0	0.00 %	51. HE	51.	4	3	75.00 %
26. VOCABULARY	26.	3	3	100.00 %	52. PLAYS	52.	4	0	0.00 %
					53. BASEBALL	53.	4	4	100.00 %
					54. DO	54.	6	3	50.00 %
					55. YOU	55.	3	2	66.67 %
					56. WANT	56.	3	3	100.00 %
					57. SOME	57.	3	1	33.33 %
					58. COFFEE	58.	3	3	100.00 %

Figure F.3: Recognition performance of explicitly modelled words.

References

1. Sanderson, M., van Rijsbergen, C.J. NRT (News Retrieval Tool), Electronic. Publishing: Origination, Dissemination and Design, 4.4, pp. 205-217, 1991.

2. van Rijsbergen, C. Information Retrieval. Butterworths. London, 2. edition, 1979.

3. Salton, G., McGill M. J. Introduction to Modern Information Retrieval. McGraw-Hill Inc., New York, 1983.

4. Aust, H., Oerder M., Seide, F., Steinbiss, V. Experience with the Philips Automatic Train Timetable Information System. Proceedings of the 2nd IEEE Workshop on Interactive Voice Technology for Telecommunications Applications, Kyoto Japan, pp. 67-72, September 1994.

5. Wilcox, L.D., Bush, M.A. HMM-Based Wordspotting for Voice Editing and Indexing. Proceedings of the European Conference on Speech Communication and Technology, vol. 1, Genoa, pp. 25-28, September 1991.

6. Lee, K.-F., Hon, H.-W., Reddy, D. R. An overview of the SPHINX speech recognition system. IEEE Transactions on Acoustic, Speech, and Signal Processing, vol. 38, pp. 35 45, January 1990.

7. Gerber, C. A General Approach to Speech Recognition. In Ian Ruthven, editor, MIRO'95, Workshops in Computing, Springer Verlag, 1995.

8. Flanagan, J. L., Del Riesgo, C. J. Speech Processing: A Perspective on the Science and its Application. AT&T Technical Journal, vol. 69, pp. 2-13, September/October 1990.

9. Chris, E. ARPA continuous speech recognition evaluation - 1994. the Computer bulletin, The British Computer Society, vol. 7, p. 18, September 1995.

10. Tatham, P. Voyaging through Strange Seas. the Computer bulletin, The British Computer Society, vol. 7, pp. 14-15, September 1995.

11. Glavitsch, U., Schäuble, P. A System for Retrieving Speech Documents. Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval, Copenhagen, pp. 168-176, June 1992.

12. Wechsler, M., Schäuble, P. Indexing Methods for a Speech Retrieval System. In Ian Ruthven, editor, MIRO'95, Workshops in Computing, Springer Verlag, 1995.

13. Young, S.J., Woodland, P.C., Byrne, W.J. HTK: Hidden Markov Model Toolkit V1.5, Cambridge University Engineering Department Speech Group and Entropic Research Laboratories Inc., September 1993.

14. Rose, C.R., Paul, D.B. A Hidden Markov Model Based Keyword Recognition System. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Glasgow, Scotland, pp. 129-132, April 1990.

15. Jones, G. J. F., Jones, K. S., Young, S. J., Foote, J. T. Video Mail Retrieval by Voice. In Ian Ruthven, editor, MIRO'95, Workshops in Computing, Springer Verlag, 1995.

16. Kupiec, J., Kimber, D., Balasubramanian, V. Speech-based Retrieval Using Semantic Co-Occurrence Filtering. Proceedings of the Human Language Technology (HLT) Conference, ARPA workshop, pp. 373-377. 1994.

17. Jones, Gareth. Private communication at the final workshop on Multimedia Information Retrieval, Glasgow, October 1995.

18. Croft, B. Text and Multimedia - possibilities and challenges. Presentation at the MIRO Workshop, Glasgow, September 1995.

19. Zue, V., Zahorian, S., Wernstein, C., et al. The Challenge of Spoken Language Systems: Research Directions for the Nineties. IEEE Transactions on Speech and Audio Processing, vol. 3, pp. 1-21, January 1995.

20. Kay, M., Gawron M. J., Norvig P. Verbmobil: A Translation System for Face-to-Face Dialog. CSLI, Lecture notes, No. 33, Leland Stanford Junior University, USA, 1994.

21. Vater, H. Einführung in die Sprachwissenschaft. Aus der Reihe: Uni Taschenbücher für Wissenschaft, Wilhelm Fink Verlag, München 1994.

22. Chomsky, N., Halle, M. The Sound Pattern of English. Harper & Row Publishers, New York, p. 11, 1968.

23. Herter, B., Eppinger, B. Sprachverarbeitung. Carl Hanser Verlag, Muenchen, Wien, 1993.

24. Deller, JR., John, R. Discrete-Time Processing of Speech Signals. Macmillan Publishing Company, New York, p. 330, pp. 791-800, 1993.

25. Storm, T.I. IBM zeigt Verständnis, c't magazin für computer technik, pp. 68-69, October 1994.

26. Itakura F. Minimum Prediction Residual Principle Applied to Speech Recognition, IEEE Transactions on Acoustic, Speech, and Signal Processing, vol. 23, pp. 67-72, January 1975.

27. Baker, J. K. The Dragon System - An Overview, IEEE Transactions on Acoustic, Speech and Signal Processing, vol. 23, pp. 24-29, January 1975.

28. Jelinek, F., Bahl, L. R., Mercer, R. L. Design of a linguistic statistical decoder for the recognition of continuous speech. IEEE Transactions on Information Theory, vol. 21,

pp. 250-256, May 1975.

29. Jelinek, F. Continuous speech recognition by statistical methods. Proceedings of the IEEE, vol. 64, pp. 532-556, April 1976.

30. Waibel, A., Hanazawa, T., Hinton, G., Shikano, Kiyohiro., Lang, K.J. Phoneme Recognition Using Time-Delay Neural Networks. IEEE Transactions on Acoustic, Speech, and Signal Processing, vol. 37, pp. 328-339, March 1989.

31. Picone J. W. Signal Modeling Techniques in Speech Recognition. Proceedings of the IEEE, vol. 81, pp. 1215-1246, September 1993.

32. Rabiner, L. R., Schaefer, R. W. Digital Processing of Speech Signals. Englewood Cliffs, NJ, Prentice-Hall, 1978.

33. Rabiner, L. R. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, vol. 77, pp. 257-285, February 1989.

34. Rabiner, L., Juang, B.-H. Fundamentels of Speech Recognition. Englewood Cliffs, Prentice Hall, New Jersey, 1993.

35. Hopcraft, J., Almoign, J. Introduction to Automata Theory, Languages and Computation. Addison-Wesley, USA 1979.

36. Baum, L. E. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. Inequalities, vol. 3, pp. 1-8, 1972.

37. Virterbi J. A. Error Bounds for Convolutional Codes and Asymptotically Optimum Decoding Algorithm. IEEE Transactions on Information Theory , vol. IT-13 , pp. 260-269, April 1967.

38. Forney, G. D. JR. The Viterbi Algorithm. Proceedings of the IEEE, vol. 61, pp. 268-277, March 1973.

39. Morgan N., Bourlard, H. Continuous Speech Recognition. IEEE Signal Processing Magazine, pp. 25-42, May 1995.

40. Pearl, J. A Constraint-Propagation Approach to Probabilistic Reasoning. Uncertainty in Artificial Intelligence. Elsevier Science Publishers, North-Holland, p. 357, 1986.

41. Aitchison, J. Words in the Mind. Blackwell Publishers, Oxford, United Kingdom, 2. edition, p. 6, 1994.

42. Clements, G.N, Keyser, S.J. CV Phonology: A Generative Theory of the Syllable. The MIT Press, Cambridge, Massachusetts, 1983.

43. Dupoux E. The Time Course of Prelexical Processing: The Syllabic Hypothesis Revisited. In: Altmann, G., Shillcock, R. (ed) Cognitive Models of Speech Processing. Lawrence Erlbaum Associates Ltd., Publishers, Hove United Kingdom, pp. 81-114, 1993.

44. Shannon, C. E. Prediction and Entropy of Printed English. Bell System Technical Journal, vol. 30, pp. 50-64, January 1951.

45. Lee, K.-F., Hon, H.-W. Speaker-Independent Phone Recognition Using Hidden Markov Model, IEEE Transactions on Acoustic, Speech, and Signal Processing, vol. 37, pp. 1641-1648, November 1989.

46. Rabiner, L. R., Levinson, S. E., Sohndhi, M. M. On the Application of Vector Quantization and Hidden Markov Models to Speaker-independent, Isolated Word Recognition, Bell System Technical Journal, vol. 62, pp. 1075-1105, 1983.

47. Davis, S. B., Mermelstein, P. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences IEEE Transactions on Acoustic, Speech, and Signal Processing, vol. 28, pp. 357-366, November 1980.

48. Rabiner L. R., Pan K. C., Soong, F. K. On the performance of isolated word speech recognizers using vector quantization and temporal energy contours, AT&T Technical Journal, vol. 63, pp. 1245-1260, September 1984.

49. Rabiner, L. R., Wilpon, J. G., Soong, F.K. High Performance connected digit recognition using hidden Markov models, IEEE Transactions on Acoustic, Speech, and Signal Processing, vol. 37, pp. 1214-1225, August 1989.

50. Ney, H., Noll, A. Acoustic-Phonetic Modeling in the SPICOS System. IEEE Transactions on Speech and Audio Processing, vol. 2, pp. 312-319, April 1994.

51. Rabiner, L. R., Juang, B. H., Levinson, S. E., Sondhi, M. M. Some Properties of Continuous Hidden Markov Model Representations. AT&T Technical Journal, vol. 64, pp. 1251-1269, July/August 1985.

52. Rabiner, L. R., Juang, B. H., Levinson, S. E., Sondhi, M. M. Recognition of Isolated Digits Using Hidden Markov Models With Continuous Mixture Densities. AT&T Technical Journal, vol. 64, pp. 1211-1230, July/August 1985.

53. Lee, K., F. Hidden Markov Models: Past, Present, And Future. Proceedings of the European Conference on Speech Communication and Technology, vol. 1, pp. 148-155, September 1989.

54. Woodland, P.C., Leggetter, C.J., Odell, J.J., Valtchev, V., Young, S.J. The 1994 HTK Large Vocabulary Speech Recognition System, Proceedings of the International Conference on Acoustic, Speech, and Signal Processing, September 1995.

55. LDC. Introduction to the Linguistic Data Consortium, Linguistic Data Consortium (LDC), University of Pennsylvania, Philadelphia, 1994.

56. Garofolo S. J., Lamel F. L., Fisher W. M., Fiscus J. G., Pallett, S. D., Dahlgren, L. N. DARPA TIMIT: Acoustic-Phonetic Continuous Speech Corpus CD-ROM, NISTIR 4930, National Institute of Standards and Technology (NIST) Gaithersburg USA, February 1993.

57. Robinson, T., Fallside, F. Phoneme Recognition from the TIMIT database using Recurrent Error Propagation Networks. Technical report CUED/F_INFENG/TR.42, Cambridge University Engineering Department, Cambridge, England, March 1990.

58. Levergood, M. T., Payne, C. A. et al. AudioFile: A Network-Transparent System for Distributed Audio Applications. Proceedings of the USENIX Summer Conference, June 1993.

59. Rabiner, L. R., Sambur, M. R. An Algorithm for Determining the Endpoints of Isolated Utterances. The Bell System Technical Journal, 54(2), February 1975, pp. 297-314.

60. Wu, S., Manber, U. Fast Text Searching Allowing Errors. Communications of the ACM, vol. 77, p 83, 1992.

61. Sankoff, D., Kruskal, J.B. Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison. Addison-Wesley Publishing Company, USA, pp. 18-21, 1983.

62. Porter, M. F. An algorithm for suffix stripping. Program, vol. 14, pp. 130-137, 1980.

63. Robertson, S. E., Sparck Jones, K. Relevance weighting of search terms. Journal of the American Society for Information Science, vol. 27, pp. 129-146, 1976.

64. Wilpon, G. J., Mikkilineni, R. P., Roe, D. B., Gokcen, S. Speech Recognition from the Laboratory to the real World. AT&T Technical Journal, vol. 69, pp. 14-23, September/ October 1990.