Automatic Alignment and Error Detection for Phonetic Transcriptions in the African Speech Technology Project Databases

EDWARD DE VILLIERS

Thesis presented in partial fulfilment of the requirements for the degree
Master of Science in Electronic Engineering
at the University of Stellenbosch

SUPERVISOR: Prof. J. A. du Preez

April 2006

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and that I have not previously in its entirety or in part submitted it at any university for a degree.

Signature Date

Abstract

The African Speech Technology (AST) project ran from 2000 to 2004 and involved collecting speech data for five South-African languages, transcribing the data and building automatic speech recognition systems in these languages. The work described here formed part of this project and involved implementing methods for automatic boundary placement in manually labelled files and for determining errors made by transcribers during the labelling process.

Various methods for aligning speech files, given the sequence of labels, are discussed and compared. The best result obtained on TIMIT was a mean boundary placement error of 9.34 ms. This compares well with the result of 8.14 ms obtained by hand aligning a portion of the same database. The best result obtained for one of the AST databases was 23.51 ms. This was a significant improvement over the result of 67.90 ms obtained when using NTIMIT models for the alignment.

Useful methods for error checking of speech databases and their label files are also presented.

Opsomming

Die African Speech Technology (AST) Projek het geloop vanaf 2000 tot 2004. Die projek het die volgende behels: insamel van spraakdata in vyf Suid-Afrikaanse tale, transkripsie van die data en die bou van spraak herkenning stelsels in hierdie tale. Hierdie tesis beskryf werk wat deel gevorm het van die AST projek. Dit fokus op metodes om handgestranskribeerde spraak outomaties te belyn, en om foute in die transkripsies te vind.

Verskeie metodes om spraak lêers te belyn, gegee 'n reeks fonetiese simbole, word bespreek en vergelyk. Die beste resultate verkry op die TIMIT databasis, was 'n gemiddelde plasingsfout van 9.36 ms. Dit vergelyk goed met 'n resultaat van 8.14 ms wat verky is deur 'n gedeelte van die databasis met die hand te belyn. Die beste resultaat op die AST databasis was 23.51 ms. Dit verbeter aansienlik op die resultaat verkry deur akoestiese modelle van die NTIMIT databasis te gebruik vir belyning.

Nuttige metodes om foute in spraak databasisse en fonetiese annotasies te vind word ook voorgelê.

Contents

List of Figures

List of Tables

Abbreviations

ASCII	American Standard Code for Information Interchange
AST	African Speech Technology
DTW	Dynamic Time Warping
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
IPA	International Phonetic Alphabet
LPCC	Linear Prediction Cepstral Coefficient
MFCC	Mel Frequency Cepstral Coefficient
ms	millisecond or milliseconds
pdf	Probability Density Function
SAMPA	Speech Assessment Methods Phonetic Alphabet
X-SAMPA	Extended SAMPA

Chapter 1

Introduction

1.1 Motivation

This work formed part of the African Speech Technology (AST) project. Two of the aims of the AST project were to create transcribed speech databases in five of the official languages of South Africa (English, Afrikaans, Xhosa, Zulu, and Southern Sotho) and to develop an automatic, interactive, multi-lingual telephone querying and booking system. The databases will form a basis for speech technology research involving South-African languages, including the aforementioned system. These databases are also very important for adapting existing speech technologies developed for foreign environments for use in South Africa. One particular field of growth in speech technology is in the callcentre market. Callcentre agents are being replaced with automatic telephone systems that guide callers to the correct service. The most expensive part of developing such a system is database collection, so the AST databases should be a stimulus for the development of speech technology in South Africa.

A speech recognition system needs to be trained on examples of the type of speech that it will be required to recognise. Therefore, speech was recorded over land-line telephones and mobile telephones for each of the languages mentioned previously. The recognisers were to be based on phoneme models; therefore, phonetic or phonemic transcriptions were desirable for training the models. Orthographic and phonetic transcriptions were produced by human transcribers. One of the components of this project was to find the boundaries between these phones. Knowing which sections of speech belong to which phone labels simplifies training and should produce more accurate models. Other uses of alignments are for easily locating sections of interest in a speech file and for linguistic research on phoneme durations.

The other component involved finding errors in the transcriptions. This was an iterative process. If an error was discovered, the file was returned to the transcribers to be corrected. If possible, automatic checks were implemented to prevent further errors of the same type from occurring. As the transcriptions became more accurate, new tests could be performed that would not have been effective on less accurate transcriptions. The transcriptions went through several checking stages. A check was run and a list of suspicious files generated. The transcribers checked the files in the list and then returned the corrected files so that the next check could be run for the next stage of checking.

1.2 Background

This section discusses some concepts necessary for understanding this document.

1.2.1 Glossary of Terms

This section describes some terms used in this document.

embedded training This is a method for training models that can be used if the acoustic file has already been labelled. The models for each label are concatenated to form a model for the entire utterance. This compound model is trained on the utterance, and then separated again to give a set of trained models modelling each label. A key feature of embedded training is that it does not require that the labels be aligned with the speech files.

feature vector This is a vector that represents part of the acoustic file. The acoustic file is converted to a sequence of feature vectors which are intended to model the acoustic properties of interest and ignore other properties. In speech recognition, feature vectors must represent those acoustic properties that distinguish phonemes, and ignore other acoustic properties such as those that can be used to distinguish speakers or moods of the speaker.

forced alignment This is a method of segmentation that can be used if the file has already been labelled. The models for each label are concatenated to form a model for the entire utterance. This compound model is then used to determine the boundaries between the segments represented by the models constituting the compound model.

International Phonetic Alphabet The IPA is a collection of symbols that can be used to represent phones unambiguously. The representation of a phone may consist of only one of these symbols or of a contiguous combination of them.

left-to-right model This is a hidden Markov model (see section 1.2.3) in which the only cycles are self loops i.e. if a state is exited by a transition that is not a self loop, the state cannot be re-entered from within the model.

phone Phones are the sounds of speech. The sounds represented by letters such as *b* and *e* are phones. The *b* in *bat* usually sounds slightly different to the *b* in *cab*, so these sounds are considered different phones even though they are represented orthographically using the same letter. In a phonetic transcription, they will be represented by the same base symbol (b), but different diacritics will be used to differentiate them.

phonemes and allophones A word is constructed from a sequence of phones. Inserting, deleting or substituting a phone in this sequence does not necessarily change the word. If the phone x can be replaced — without a change in word meaning — with the phone y in all words in which it occurs and vice versa then x and y are allophones. If one takes all the words of a language and groups the phones into allophone groups then each of these groups is called a phoneme. A phoneme is a unit used to differentiate words in a language. Two phones may be allophones in one language but assigned to different phonemes in another language. For example, the voiceless bilabial plosive (p) and the aspirated voiceless bilabial plosive ($\mathrm{p^h}$) are allophonic in English but not in Thai [12].

segmentation The process of determining time boundaries between segments of interest in an acoustic file. More specifically, it is the process of dividing an acoustic file into a number of consecutive, non-overlapping, undivided temporal regions spanning the entire file. If the label sequence is known, the number of segments is predetermined and forced alignment can be used.

transcription The process of coupling text strings with acoustic files. The process is called phonetic, phonemic or orthographic transcription if the text strings consist of phones, phonemes or words respectively.

validation This is the process of searching for errors in the data (transcriptions, segmentations and audio files).

1.2.2 Measurement of Alignment Accuracy

For the segmentation experiments in this work, the label sequence is given. The segmentation procedure therefore involves placing boundaries to divide the acoustic files into segments representing each label. The number of segments is fixed, so the number of boundaries is fixed. The segmentation reduces to a forced alignment of each file. The first boundary and the last boundary of each file are fixed at zero and the length of the acoustic file respectively. The internal boundaries move during forced alignment, but the order of the boundaries may not change. When a single internal boundary moves, one segment increases in size and another decreases by the same amount.

The results of the alignment experiments are compared by calculating the difference in time location between each boundary placed during alignment and the corresponding boundary in the reference alignments. The ideal is to have the boundaries positioned by an experiment placed at exactly the same locations as the boundaries in the reference alignments. When the automatically positioned boundaries do not match the reference boundaries, either the automatic boundaries or the reference boundaries or both may be incorrectly placed. The experiments have been performed under the assumption that the reference boundaries are correct. Therefore, the differences in boundary location should ideally be zero and any non-zero difference is considered an error. Most of the reference boundaries should be reasonably accurate but the reference alignments will contain some errors.

Because the only aim is to get the generated boundaries as close as possible to the reference boundaries, it is immaterial whether the generated boundary has been placed before or after the reference boundary. The absolute value of the difference in time location between the two boundaries is therefore used.

For the majority of the experiments, three quantities were calculated from each set of error values:

- The maximum absolute error.
- The mean absolute error.
- The standard deviation of the absolute error.

The experiments are compared according to these three values. In some of the later experiments, the median was added as a fourth measure because it was felt that large maximum errors were skewing the mean and therefore making it less useful for comparing experiments. The mean is used as the primary method of comparing experiments because it is one of the two methods used in the literature (see chapter 2). It is useful not only to be able to compare the experimental results with each other but to know how they compare with the results achieved by others.

The boundaries used in the error calculation are end times. The begin times of segments were ignored because the begin time of every segment except the first is identical to the end time of the previous segment. The begin time of the first segment should not be used either because this is always zero and including it in the calculation would give results that appear better than they actually are (because the first begin time will always be placed correctly). The final end time will also always be placed correctly, so it is excluded from the calculations too. This decision was useful for measuring TIMIT [13] errors because the worst boundary placement errors in the TIMIT reference alignments involve the placement of the final boundary.

1.2.3 Some Hidden Markov Model Basics

Figure 1.1 shows an example of a hidden Markov model (HMM) [19]. In this work, phones are modelled by HMMs. An HMM is a type of state machine. There are two types of states:

emitting states have an associated probability density function (pdf).

null states do not have an associated pdf.

In the figure, the large circles are emitting states and the small circles are null states. The objects above the emitting states represent the pdfs. The arcs show permitted transitions between the states with the arrowheads indicating the permitted direction of the

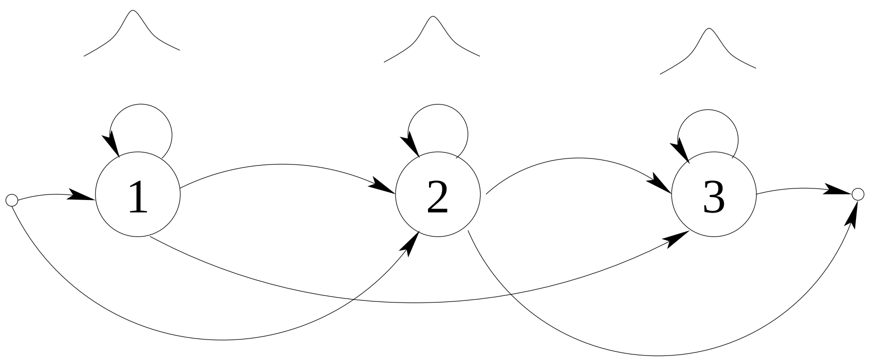

Figure 1.1: *An example of a three state HMM with two null states.*

transition. In the figure, a transition from state 1 to state 2 is permitted, but a transition in the opposite direction is not permitted. Also, a transition from the initial null state to the final null state is not permitted. The transition from a state to itself is called a self loop. Transitions have weights and the sum of the weights of all the transitions out of a state is equal to one. The pdfs model feature vectors. Emitting states may share the same pdf. If an HMM is used in *generating* mode, each time an emitting state is entered, its pdf emits a feature vector. Because null states do not have pdfs, a transition into a null state does not produce a feature vector. When an HMM is used in *recognition* mode, feature vectors in the feature file are associated with emitting states in the model. For each transition into an emitting state, the position in the feature file is increased by one. A transition into a null state does not change the position in the feature file.

1.3 Objectives

- To design methods for the accurate automatic alignment of phonetic labels with the speech of the African Speech Technology Project databases.
- To implement these methods and evaluate their accuracy.
- To use these results to provide the African Speech Technology Project databases with automatically generated alignments.
- To develop checks for the correctness of transcriptions and segmentations done by computers and people.

1.4 Contributions

- It was found that an HMM/Viterbi-based approach was suitable for aligning databases with a known label sequence. See section 4.3.3.
- Various HMM configurations were evaluated on the hand-aligned TIMIT database. The advantage of using duration modelling (see section 4.4.3) with a fine-grain feature vector sampling period of 5 ms proved particularly effective.
- It was found that it helps to separately force a minimum duration customised for each phone (see section 4.4.4).
- Together, the above findings resulted in a mean boundary error of 9.34 ms which compares well with a mean error of 8.14 ms achieved with a manual segmentation experiment.
- The AST data was automatically aligned. A check against a hand-transcribed portion of the database indicates the mean error of the order of 23.51 ms. This is a large improvement over the result of 67.90 ms when aligning with models trained on NTIMIT.
- Various automatic error checking procedures were implemented for use on the AST data. All the transcribed files underwent these checks and reported errors were corrected. This contributed greatly to the final quality of the databases.

1.5 Overview

Section 1.2 describes some background theory for this work. It gives a brief introduction to hidden Markov models and describes how alignment accuracy is measured for the experiments in chapter 4. It also contains a glossary of some important terms used in the rest of the document.

The literature review explains the work done over a twenty year period to achieve automatic alignment for speech files where the label sequence is known. It also describes some literature on experiments comparing manual and automatic transcription. In one experiment, several manual transcriptions were compared and a mean boundary error of 6 ms was found.

Section 4.2 gives reasons for automatic transcription. These include increased speed and consistency and reduced cost. Embedded training and forced alignment are described in section 4.3 and the use of a lexicon with these methods is explained. The rest of the chapter describes alignment experiments. These are divided into three groups. The first group of experiments was performed on the TIMIT database. Various model structures and feature types were compared. It is shown that models for which the minimum duration is customised for each phone give the best results. Also, of the various frame shifts used, 5 ms is the most successful. Special features like zero crossing density were found not to improve results. Neither did grouping phones or separating files according to the gender of the speaker. The second group of experiments studies the impact of limited bandwidth and channel noise, which are factors when working with telephone data. TIMIT is first downsampled and then NTIMIT is used. The bandwidth limiting does not significantly reduce results, but the channel noise does. The final group of experiments was performed on the EE database. This is one of the AST databases. A portion of this database was hand-aligned, so the success of alignment experiments could be determined. Training models on the EE data resulted in a significant improvement over using NTIMIT models for alignment of this database.

Chapter 5 lists methods for finding errors in speech databases. These error checks are useful for improving the quality of the databases which in turn improves the quality of automatic alignments performed on those transcriptions. One error check that combines duration models for phones in a transcription to produce a duration model of the entire transcription is described in section 5.6.

Finally chapter 6 gives conclusions and future work.

Chapter 2

Literature Study

This chapter is divided into two sections. Section 2.1 explains methods for performing segmentation on acoustic files with a known label sequence. Section 2.2 gives results of comparisons between segmentations produced by manual transcribers and automatically produced segmentations.

2.1 Segmentation Methods for Labelled Acoustic Files

The references in this section are arranged in the order of the publication year. From these articles, a shift from methods involving template matching or dynamic time warping/dynamic programming to methods involving either hidden Markov models or neural networks is evident. This shift has not been confined to segmentation but has occurred throughout the field of speech recognition.

2.1.1 The Eighties: Dynamic Programming Methods

Höhne et al. [7] presents a DTW method for aligning two manifestations of the same utterance. Either of the two manifestations could be natural or synthetic speech. One of the files is hand-aligned and the DTW procedure is used to map these boundaries to the second file. To use this method for database creation, one can record multiple examples of the same sentence, hand-align one of these examples and use the DTW procedure to align the other examples. The authors were primarily interested in using the segmentations for speech synthesis, so the position of the boundaries was not as important as the consistency of boundary placement. Therefore, boundary locations were chosen to coincide with prominent acoustic events. There was a 14 ms average error between boundaries found by the DTW algorithm and hand-aligned boundaries on the same file. The algorithm performed well when aligning natural speech to natural speech or synthetic speech to synthetic speech. When one of the files was natural speech and the other synthetic, there was a much better result if the synthetic utterance was the one that was hand-aligned and the natural speech was aligned with it, than when the roles of the two files were reversed.

In [14], a three stage method is used. The speech is first segmented into six broad phonetic classes using a sequence of binary classifiers. The actual label sequence is not used in this stage. Feature vectors are calculated every 5 ms (The signal is sampled at 16 kHz). The calculation and size of the feature vectors are different for each classifier. The results of the binary classifiers are combined to produce a label from the six broad phonetic classes. In the second stage, the broad phonetic labels from the first stage are matched with the actual label sequence using dynamic programming constrained by phonetic rules. The constraints are of two types: phonemes in the label sequence must be matched to the correct class in the segmented speech, and certain contextual rules must be obeyed, including durational rules. Stage two results in a segmented utterance that can have one segment mapped to one actual label, one segment mapped to more than one actual label, or more than one segment mapped to one actual label. For the first case nothing needs to be done. For the third case, the multiple adjacent segments are collapsed into one. The third stage is for handling the second case. If there is an acoustic event that is known to separate two phones, this event is located in the signal and used for boundary placement. Otherwise, an ad hoc rule is used to subdivide the segment. This algorithm achieved 75% of the boundaries correct to within 10 ms but in some cases segments still had to be manually divided to complete the segmentation.

The method described in [20] uses template matching. Templates comprise two frames in the case of diphthongs and plosives; otherwise one frame. The first frame for plosives is the same as that for silence. The reference frames are determined by averaging the autocorrelation vectors of the speech sections that they are to represent. A normalised distortion measure is computed between each speech frame and each reference vector. This gives a matrix with dimensions of number of frames by length of the reference sequence. The speech frames are then aligned with the reference vectors using dynamic programming. This procedure attempts to minimise over all segments the sum of the normalised distortion measures. The length of a segment is bounded from above and below during this process and no labels present in the reference sequence can be omitted. 92% of boundaries were placed within 45 ms of their corresponding reference boundaries. This method was combined with a method that determines boundaries via temporal clustering, but the results did not improve.

Torkkola [21] also divides the speech into classes in the first step (as in [14]), but the rest of the procedure is far more involved. He divides his segmentation process into the following steps:

1. Assign the speech frames to phonemic classes using the Phonotopic Map Method. Phonotopic maps are neural networks with a spectral template associated with each neuron. The maps are of two types: stationary and transient. The transient maps are specifically for distinguishing plosives. Each frame is assigned the label of the neuron that has the best matching spectral template.

2. Construct the phonemic transcription of the utterance. The frame labels produced in the previous step are examined and a group is assigned a phoneme class if a certain percentage of the labels in a continuous group of labels are the same.

3. Compute a smoothed spectral stationarity function. This gives a measure of the rate of change of the spectral vectors.

4. Perform a rough segmentation of the speech signal to find plosives and fricatives which will form anchor points during subsequent segmentation. Dynamic programming is used to align the phonemic transcription produced in step 2 with the input phonetic transcription. The borders of the stationary parts of the plosives and fricatives that match between the two transcriptions are determined using heuristic rules and these are fixed. The spectral stationarity information from the previous step is used, together with RMS energy and high-frequency energy, in the heuristic rules.

5. Construct a list of speech events having stationary properties. A search is conducted among the anchor points determined in the previous step to find speech events. The sequence determined in step 2, the spectral stationarity from step 3, and the signal

energy are used to determine the events. Events are assigned a confidence measure. The duration of the event is included in the calculation of the confidence measure.

6. Align the events list with the phonetic transcription. The cost in the dynamic programming algorithm depends on a distance measure and the confidence measures assigned in the previous step.

7. Fix the segment boundaries of phonemes fulfilling certain criteria. These are the boundaries that are most likely to be correct at this point.

8. Events that could not be fixed in the previous step are split or merged with adjacent events according to heuristic rules.

A 13.02 kHz sampling rate is used, the vectors have fifteen components and the frame rate is 10 ms. 74% of the boundaries were within 10 ms of the actual ones and 94% were within 30 ms.

2.1.2 The Nineties: Hidden Markov Models Become Predominant

A combination of two segmentation methods is used in [23]. The first method does not use the given phonetic transcriptions. The normalised correlation between frames that are within a certain temporal window is calculated and those frames producing a correlation value above a predetermined threshold are assigned to the same segment. The second method uses a reference spectrum for each phone in the transcription or two for diphthongs and plosives. Dynamic programming is then used to determine the segment boundaries with the function to be minimised depending on the sum of the correlations between these reference spectra and the spectra of the speech signal. This is similar to the dynamic programming method in [20]. The possible length of each segment is limited by the minimum and maximum duration allowed for the phone that it is to represent. The first method was found to position the segment boundaries more accurately, but the number of segments produced was not necessarily correct. The two segmentations are merged by first linking the boundaries of the one to the closest boundary in the other. If the number of segments in the two segmentations are different then some boundaries will not be matched up. The segments of the second method are then taken as the correct ones but any boundaries that have been linked to boundaries produced by the first method are replaced by those boundaries. 96% of automatically generated boundaries fell within 30 ms of manually generated boundaries. The second method alone was found to be inferior to the combination of the two methods.

Alignment with both phonetic and orthographic transcriptions is discussed in [15]. When phonetic transcriptions are given, a model is constructed from the phones in the transcription and this model is used to recognise the utterance. The utterance is then segmented according to which part of the model it was matched to during the recognition process. If an orthographic transcription is given, the model contains likely phonetic realisations of the orthographic transcription, and the segmentation is done by finding the best path through this model while recognising the utterance. The model that is assembled is similar to a second order ergodic continuous variable duration HMM (CVDHMM) in which each state corresponds to a phone. It is made up of three types of models: trigram phonotactics models, phone duration models and acoustic phone models. The phonotactics model represents the phone sequence that is known to occur in the utterance or one that is built up from the orthographic transcription of the utterance using a phone realisation tree. It controls the state transitions in the CVDHMM. The acoustic models consist of a sequence of three full covariance Gaussian pdfs corresponding to the beginning, middle and end of the phone respectively. Each acoustic model is associated with a state in the CVDHMM. The phone duration models (one for each acoustic phone model) are gamma distributions with four parameters: the mean and variance of the duration and the minimum and maximum allowed duration. These duration models control the state durations in the CVDHMM. Five different types of context dependencies were used in the acoustic models: triphonic (the start, middle and end of the phone depends on the left and right context), quasi-triphonic (the start of the phone depends on the left context, the middle is context independent and the end depends on the right context), left context dependence, right context dependence and context independence. Context clustering was used to reduce the number of models to be trained. Experiments were performed on TIMIT, with a reduced phone set of 47. The frame shift was 10 ms and the frame length 20 ms. On the pure segmentation experiments, 80% of the boundaries were placed within 15 ms of the boundaries from the test set. When the combined segmentation and labelling experiments (where the orthographic transcriptions were given) were performed, a 10%

labelling error rate was obtained with 80% of the boundaries correct to within 17 ms.

A Viterbi search and a level building algorithm is used in [3]. The speech signal is aligned with phonemes modelled by Gaussian mixture models. The cepstral vectors of the speech are passed through an Acoustic-Feature Map (a type of neural network) and transformed into vectors with components that indicate the features (e.g. frontness, roundness) that make up the speech frame. A principle component analysis is then done on these vectors and the resulting vectors (reduced in dimension) are used to train the Gaussian mixture models and to represent the speech that must be aligned. In an experiment using this method, 78% of the automatically positioned boundaries were within 20 ms of the manually placed boundaries.

An orthographic transcription is the input to the three stage method used in [9]. In the first stage, a pronunciation variant graph is built up from the orthographic transcription. A canonical phonetic sequence is derived by concatenating phonetic symbols obtained by a lexicon look up. An algorithm is then applied to the canonical sequence to generate other paths in the graph that represent variants of the canonical form. The nodes of the graph so formed represent phones and the edges represent transition probabilities between these phones. In the second stage, the speech features are time aligned (using a Viterbi algorithm) with the HMM generated from the pronunciation variant graph. The HMMs for each phone represented by a node are connected to each other using the transition probabilities encoded in the graph. Because the Viterbi alignment works with the feature vectors and not the actual speech signal, the accuracy of the boundary placement is limited by the spacing between feature vectors. Therefore, in the third stage, the boundaries are moved to the next zero crossing in the speech signal or, for a vowel, to the nearest zero crossing preceding the peak amplitude. In an experiment, 59% of boundaries were correct within 10 ms.

In [16], a segmentation method using a combination of HMMs and an artificial neural network (ANN) is compared with a method involving alignment with synthetic speech. The synthetic speech method takes an orthographic transcription and produces a phonetic transcription which is used to generate synthetic speech. The same feature extraction is applied to the synthetic speech and the speech to be segmented and a DTW procedure is used to align the two. The synthetic speech does not have to be of the quality of natural speech. It must only produce features that match well to the ones in the natural speech to be segmented. For example, if the features do not represent pitch, then the synthesiser does not have to determine the correct pitch contour for the utterance. In the other method, the previous speech synthesiser method was used to segment a training database that had phonetic labels but no segment boundaries. Then, the segmented database was used to train HMMs with diagonal covariance Gaussian mixture models (with sixteen components each). These HMMs were then used to segment the training database more accurately. The database was then used to train multi-layer perceptrons which were used to segment the test set. The two methods produced similar results. The speech synthesis method has the advantage of not needing training. The HMM/ANN method needs training but can handle multiple phonetic transcriptions more easily.

2.2 Comparing Manual and Automatic Segmentations

In [2], the automatic system gets a 27 ms mean error whereas manual segmentation of a small sample (10 sentences) of continuous speech by three transcribers (after a lot of practice) gave a 6 ms mean error. The transcribers' best results were on transitions from a plosive to a semivowel which had an average error of 2 ms. Their worst results were on transitions from a semivowel to a vowel which had an average error of 16 ms. 10% of cases have an error exceeding 20 ms. It is important to note that transcribers would almost certainly perform worse on a larger database due to fatigue.

Kvale [10] reports that different manual segmenters get 95% of boundaries within $\pm$ 20 ms accuracy of a reference and that his segmentation method placed 88% of boundaries within $\pm$ 20 ms of manually placed boundaries.

Wesenick and Kipp [27] compare label sequences and segment boundaries between manual transcribers and an automatic system. The test was done on German speech to evaluate the MAUS automatic transcription system. The transcribers were given the standard pronunciation for each word and asked to modify them if necessary. The label sequences were then compared using DTW between the reference and the test sequence. Each utterance was transcribed by four different persons. Each transcription was used as the reference sequence in turn. This resulted in twelve combinations of reference and test sequences being compared per utterance. Only consonants were compared. There were 21 phonemes used in the test. The worst score for a phoneme was for `d` — there was only 79.6% agreement. The best score was for x (99.4%). The worst class was the stops (89.9%). Fricatives constituted the best class (98.0%). The total agreement score was 94.8%. The automatic transcription was compared with each manual transcription. The worst result was for l (64.1%) and the best result for f (99.6%). The worst class was the stops (80.2%) and the best class was the nasals (94.4%). The total agreement was 88.4%.

Next, boundaries of segments for which labels agreed were compared. For the manual transcribers, the worst transition was from a nasal to a nasal (16 ms average error) and the best transition was from a voiceless fricative to a voiceless plosive (5 ms average). For the automatic transcriptions (compared with the manual transcriptions) the worst transition was from a nasal to a nasal (43 ms average) and the best transition was from a voiceless fricative to a vowel (6 ms average). The manual transcribers placed 73% of boundaries with less than 5 ms error, 87% with less than 10 ms error, 96% with less than 20 ms error and 99% with less than 32 ms error. The automatic system placed 36% of boundaries with less than 5 ms error, 61% with less than 10 ms error, 84% with less than 20 ms error and 90% with less than 32 ms error.

2.3 Summary

The above articles describe various methods for aligning speech files with label sequences. They show the move from template matching methods to the use of hidden Markov models and neural networks.

The two ways in which alignment errors between the reference and test segmentations are measured in the above articles are with the mean error and with the binning of error values. In the binning method, the results reported are that a% of errors are lower than x ms, b% of errors are lower than y ms etc. The measure chosen for comparing experiments in this work should also facilitate comparison with the articles above. The mean was chosen as a measure as it allows comparison with any article that uses this measure. Comparison using the binning method depends on the choice of bins and since different articles above use different sets of bins, comparisons would be very complicated. The two mean errors for automatic segmentation reported above are 14 ms [7] and 27 ms [2].

Chapter 3

Background on the African Speech Technology Project

This chapter gives some information about the African Speech Technology (AST) project. Section 3.1 describes what data was collected and how it was collected. Section 3.2 describes the transcription files. Finally, section 3.3 describes the phone set used in the transcriptions.

3.1 Speech Databases

The project collected speech data for five languages: English, Afrikaans, Xhosa, Zulu, and Southern Sotho. To cope with pronunciation variations due to non-mother-tongue speakers, English and Afrikaans were subdivided into accent groups. For English, there were five accent groups and for Afrikaans there were three. Therefore, there were eleven language / accent variants in total. Volunteer speakers were recruited from each accent group and each speaker was given a data sheet containing text in his language. The data sheets for each language had the same format but different contents. The text items included digit strings, numbers, spelt words, place names and phonetically balanced sentences. Each volunteer was required to telephone into a system and read this text over the telephone. The caller also had to provide personal information such as the type of telephone used (land-line or mobile), his age, and the town in which he attended school. The system prompted the caller to read each data sheet item. The caller's responses were recorded and one A-law [4, page 437] file was produced for each prompted answer.

Therefore, the number of A-law files per call should ideally have been equal to the number of data sheet items plus the number of personal questions. This was not the case if the caller terminated the call prematurely or read more than one text item in response to a prompt. The volunteer's details were written on the data sheet and it was returned to the recruiters. The recruiters tried to get an equal number of male and female callers, and mobile and land-line calls. They also tried to have a uniform spread of ages from twenty to sixty. About six hours of speech (excluding surrounding noise and silence sections) was gathered per database.

3.1.1 Database Acronyms

Two letter acronyms were used to distinguish the eleven databases. They were:

AA Afrikaans spoken by mother-tongue speakers of European descent.

AE South-African English spoken by Afrikaans mother-tongue speakers of European descent.

BA Afrikaans spoken by speakers of African descent.

BE South-African English spoken by speakers of African descent.

CA Afrikaans spoken by speakers of mixed African and European descent.

CE South-African English spoken by speakers of mixed African and European descent.

EE South-African English spoken by mother-tongue speakers of European descent.

IE South-African English spoken by speakers of Indian descent.

SS Southern Sotho spoken by mother-tongue speakers.

XX Xhosa spoken by mother-tongue speakers.

ZZ Zulu spoken by mother-tongue speakers.

3.2 Transcription Files

Transcribers produced orthographic transcription files for each of the recorded speech files. Silence and different types of noise were also transcribed. The transcribers placed time boundaries around isolated silence and noise sections, and around speech sections (possibly containing noise and silences) corresponding to an item of text on the data sheet. The exact words or word fragments used by the caller were transcribed.

Phonetic transcriptions were produced automatically using either letter-to-sound rules (e.g. for Xhosa) or pronunciation dictionary look up (e.g. for English). In another round of transcription, these phonetic transcriptions were checked and corrected where the speaker used a different pronunciation to the one in the original phonetic transcriptions. Once the phonetic correction process was completed, new pronunciation lexicons were generated directly from the transcription files. Although these lexicons have a small word coverage, they are useful for giving an indication of the types of phone replacements, assimilations, etc. that occur.

3.3 Phone Set Used for Transcription

The final phone set contained about 150 phones. X-SAMPA [25] was used to encode the phones so that they could be entered easily using a standard keyboard. X-SAMPA uses ASCII symbols to encode each IPA character (see Appendix A). In some cases more than one ASCII character is used to represent one IPA character, but an X-SAMPA string parses uniquely. Another phone set (ASTbet, see Appendix A) was developed as part of the AST project that could be used in filenames (so that the file containing a phone model could have the same name as the phone it contains). This was necessary because of limitations on filenames imposed by HTK [28] which was used for some parts of the project.

The set of about 150 phones included diphthongs and affricates so that they could be transcribed differently from the sequence of their two constituent phones. This policy allowed for explicit modelling of diphthongs and affricates, but they could still be modelled separately by replacing the single symbol with the sequence of the two symbols for the constituent phones of the diphthong or affricate.

The transcriptions are referred to as phonetic and not phonemic because the transcribers were not limited to using symbols for phonemes. Some examples of non-phonemic distinctions are:

- voiceless versions of phones in the set e.g. `I_0`, `@_0`, `n_0`
- lax vs tense vowels: `I` vs `i` and `U` vs `u`
- `r\` vs `r`
- glottal stops: `?`
- `h` vs `h\`
- non-phonemic long vs short distinction: e.g. `{` vs `{:`
- different pronunciations of the same diphthong e.g. `E-\I`, `E-\i`, `e-\I`

The transcriptions were done in such great detail to enable them to be useful for linguistic research in addition to training models for automatic speech recognition. The two considerations for whether a new phone symbol should be created were whether the sound to be described by the symbol was significantly different acoustically from the closest existing sound in the set and whether there were enough examples of the sound to justify a separate symbol.

3.4 Summary

Telephone speech for five languages was gathered and a phonetic transcription was produced for each file. The transcriptions were very detailed both in terms of the size of the phone set and the fact that various types of noise were transcribed. Boundaries were placed between major sections of the files.

Chapter 4

Segmentation Work

4.1 Introduction

This chapter describes the method used for segmentation, experiments to improve the results of the segmentation procedure and results obtained on the AST databases.

The first experiments use the TIMIT [13] database. TIMIT is a speech database comprising speech from several accent groups of United States speakers. This database was used for most of the experiments for two reasons:

1. TIMIT recordings are of high quality read speech. Because there is a low noise level, very little speaker noise and few speech disfluencies, these factors should have very little influence on the experiments to determine the best features for segmentation.

2. Most of the experiments in the literature study were performed on high quality speech. The exceptions is [20] which uses telephone speech, but the measure of success used is that a boundary is within 45 ms of its reference boundary which is a very rough error measure. TIMIT was used so that the results of the experiments performed here would be more comparable with the experiments in the literature which had tighter error measures.

The next database used is NTIMIT which contains the TIMIT files passed through telephone channels. This should give results that are more comparable with AST databases. Both TIMIT and NTIMIT are used because they have been hand-aligned and therefore can be used for evaluating segmentation methods.

Finally the EE database of the AST project is used. Part of this database was hand-aligned to be used for evaluating the segmentation method, but these alignments were only available at a late stage of the project, so TIMIT and NTIMIT had to be used until the alignments became available.

4.2 Automatic vs Manual Segmentation

It is desirable that the segmentation process be automated for the following reasons:

- Segmenting many files is a time-consuming process for humans. The transcribers could spend their time on other tasks if a computer could produce the segmentations.
- A computer can segment a file much faster than a person can.
- Segmentation is a repetitive and tedious process, but the correct boundary locations are not obvious, so the transcribers must remain alert.
- Skilled persons are required. They are therefore expensive to employ and they may be scarce.
- Automatic boundary placement is more consistent — automatic segmentation produces the same result on the same file, whereas a manual transcriber may place the same boundary at a different location on a second consideration of the file.

Note that in the final item above, the automatic system might be consistently wrong for a particular type of boundary. Manual transcriptions may also suffer from this problem if the transcriber has an incorrect understanding of where a boundary in a certain context should be placed.

Another important point is that for concatenative speech synthesis systems [1], the consistency of the boundary placement is more important than the actual location [7]. If the boundaries have not been placed consistently, the segments to be concatenated will not match up where they are joined.

4.3 Segmentation Procedure

In this chapter, it is assumed that the label sequence is known and is correct. This implies that transcribing an acoustic file only involves finding $n-1$ boundaries where n is the number of labels; this excludes the initial and final boundaries of the file which are fixed at zero and the length of the acoustic file respectively. Discovering and correcting errors in the label sequence is discussed in Chapter 5.

Although an acoustic file can be segmented directly, our procedure involves first calculating feature vectors from the file and then segmenting the file of feature vectors. The vectors are produced by moving a window or *frame* over the acoustic file in a series of constant length steps and calculating a single vector after each step. The step size is known as the *frame shift*. The window size is known as the *frame length* and is usually greater than the frame shift so that every part of the file will be used for calculating at least one vector. The choice of the frame length is a compromise between having sufficient data for calculating each vector and limiting the influence of surrounding parts of the file on the calculated vector. The frame shift determines the time resolution for the boundaries i.e. the minimum distance apart that boundaries can be placed. Therefore, for an accurate segmentation, a small frame shift is desirable. As an example, there is a `d` in the TIMIT test set that has a duration of 2.37 ms. The usual 10 ms frame shift used for recognition would probably not be able to detect this occurrence. Figure 4.1 shows a graph of minimum duration setting vs excluded examples for TIMIT examples of the phone `b`. A model that has been trained only to recognise phones with a duration greater than a certain minimum setting will not properly recognise the corresponding percentage of examples given in the graph. If the frame shift is 10 ms and the model must accept at least three vectors, then only phone examples with durations greater than or equal to 30 ms will be properly modelled and from figure 4.1 this would exclude over 90% of the `b` examples. Likewise, figure 4.2 shows the percentage of exclusions vs minimum duration setting for all the phones in the TIMIT files. This graph shows that a minimum setting of 30 ms would exclude about 15% of all the phone examples.

Table 4.1 lists the phone segments with the shortest examples in the TIMIT training set. Appendix B explains the phone symbols and gives more detailed duration statistics.

There is a minimum duration for segments produced by automatic boundary place-

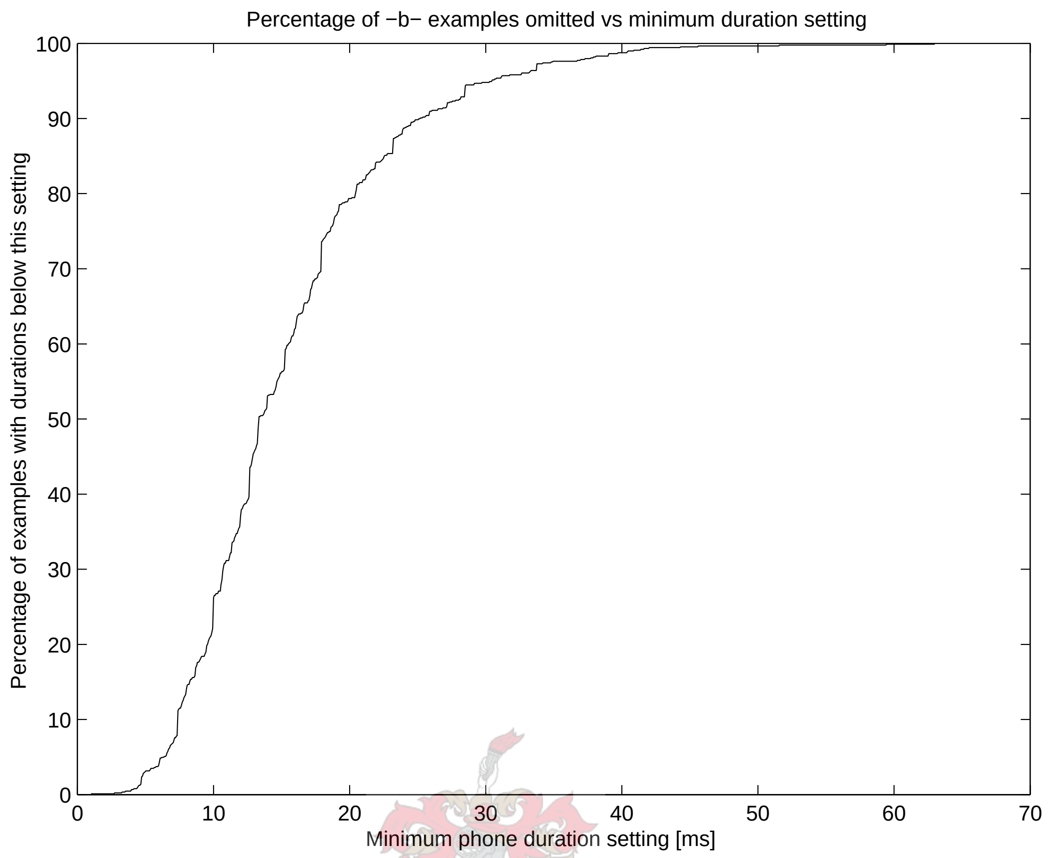

Figure 4.1: *Minimum duration setting vs examples excluded for the phone* b.

ment. If the acoustic file is segmented directly, the smallest segment is one sample, so the shortest segment length is equal to the sampling period. If a feature file (file of feature vectors) is segmented, the smallest segment is one feature vector, and the shortest segment length is equal to the frame shift. If a boundary has been placed correctly (the correct frame has been chosen — the one that has its centre closest to the ideal boundary location), the maximum error between the boundary and the ideal boundary position on the corresponding continuous waveform is equal to half the frame shift. This is true because the distance between any point on a line segment and the boundary (left or right) of the line segment closest to the point is at most half the length of the line segment. The minimum possible error is zero. If one assumes that the actual boundary positions on the continuous waveform are independent of the frame shift (i.e. they do not have some inherent period), then for a segmentation in which all the boundaries are correctly placed, the errors should be uniformly distributed between zero and half the frame shift. The mean error for a perfect segmentation is therefore a quarter of the frame shift. The inde-

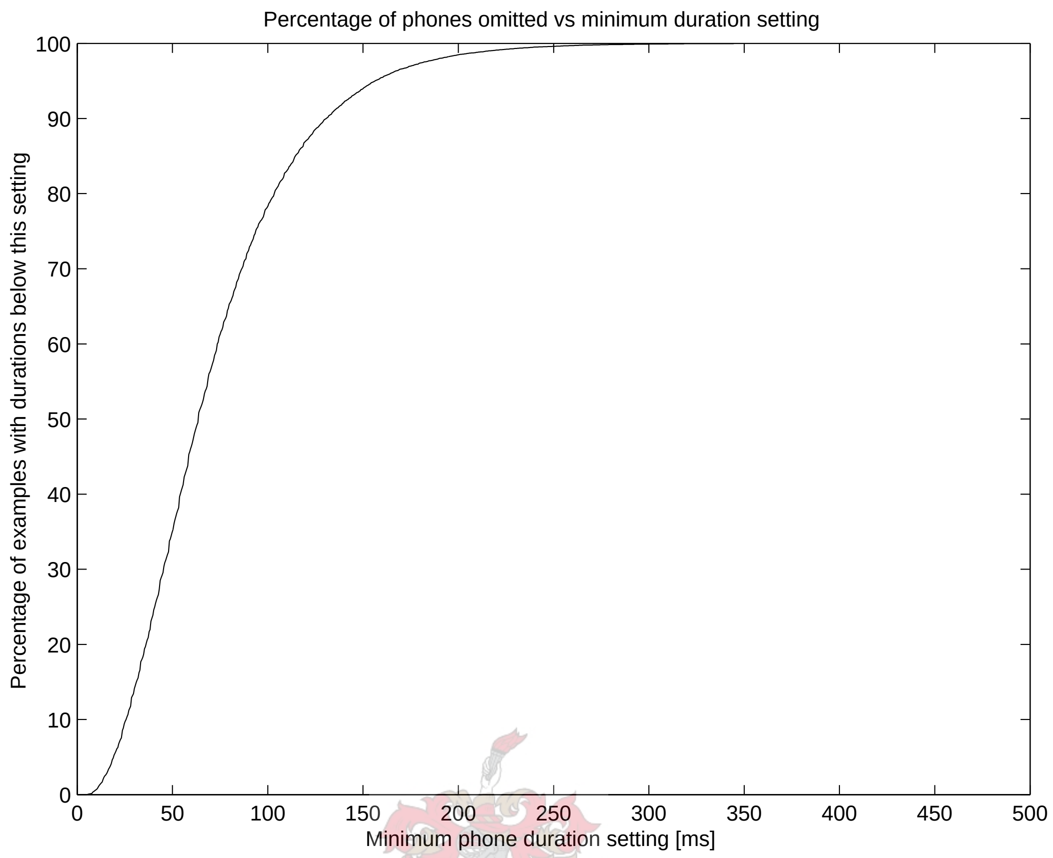

Figure 4.2: *Minimum duration setting vs examples excluded for all the phones in the TIMIT files.*

pendence assumption does not hold when there are segments shorter than half the frame shift. Then one will have a single frame being the closest to two boundaries which implies a zero length segment. Since zero length segments are forbidden, one of the boundaries will have to be linked to another frame which will be further away.

One disadvantage of a small frame shift is that the number of vectors calculated increases in direct proportion to a decrease in frame shift, resulting in an increased processing time and memory usage.

The shortest frame length we use in the experiments is 10 ms. The pitch range of the human voice is from 60 Hz to 800 Hz [6]. This means that the furthest apart that pitch pulses can be is about 17 ms. This is longer than 10 ms, so the window will not contain an entire pitch period which puts the feature extraction techniques at a disadvantage [4, pg. 232]. But the results with these short frame lengths will be compared to results with longer frame lengths and the method of measuring success (see section 1.2.2) is

Table 4.1: *Minimum durations of phone segments in the TIMIT training set (Closure and burst of stops are separate segments).*

phone	minimum duration [ms]
ax	4.62
b	2.94
d	4.25
dh	4.50
epi	3.19
h#	2.00
r	2.25

independent of the frame length.

4.3.1 Models

Hidden Markov models (HMMs) are used to model the phones in the label sequences. The basic models have three emitting states where the first state, second state, and third state are meant to model the transition into the sound, the steady state portion of the sound, and the transition out of the sound respectively. Some variations on this basic model will be described later. The emitting states have Gaussian mixture models (GMMs) as their pdfs. Given enough Gaussians in the mixture model, one can model any continuous pdf [4, pg. 706], so it is a good choice for letting the model learn about the phone without any expert knowledge of the internal structure. However, care must be taken to choose a sufficient number of Gaussians and to initialise them properly.

4.3.2 Embedded Training

The TIMIT database gives boundaries between labels, so these boundaries are used during training. For the AST databases, boundaries are absent, so embedded training must be used. It has the following steps:

1. Load the phone model for each label in the known label sequence.
2. Form a compound model by concatenating these phone models in the order given by the label sequence.
3. Implicitly train the phone models by training the compound model as a single model.
4. Break up the trained compound model.
5. Save the separate trained phone models.

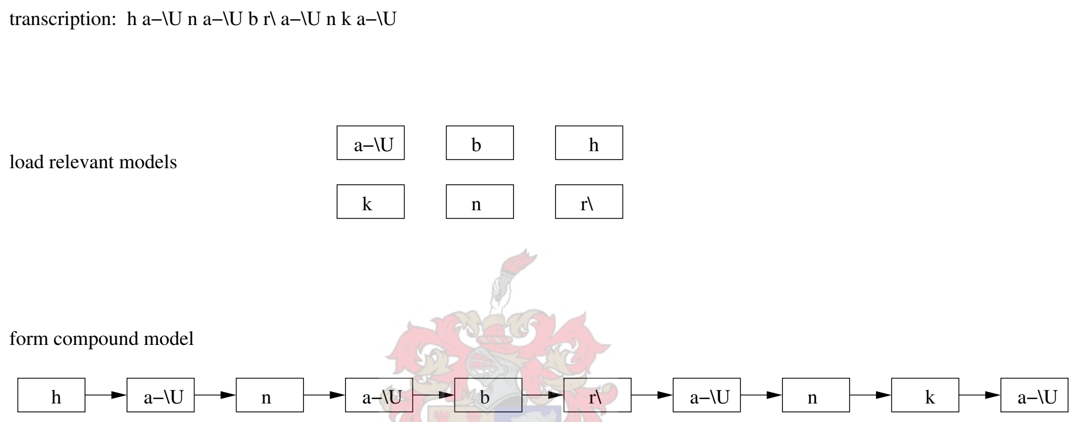

Figure 4.3: *Constructing the compound model from the phone models in embedded training.*

Figure 4.3 shows how the compound model is constructed. The transcription is of the phrase *how now brown cow*. The phone symbols are in the X-SAMPA phone set (see Appendix A). The phrase has two phones that occur more than once: `a-\U` and `n`. Notice that these phones are only loaded once. When the compound model is constructed, the phone models are linked in the same sequence as the phone labels in the transcription with a single link between each pair of models. The link is directed so that only forward transitions between models are possible. This ensures that single contiguous sections of the feature file are assigned to each constituent phone model in the compound model.

The compound model is easily separated because the training algorithms (Baum-Welch and Viterbi [19]) only change weights in the model and not the model structure. Stated differently, the training algorithms adjust model parameter values but do not change the number of parameters or the meaning of each parameter.

The method used in this work for constructing the compound model is shown in figure 4.4. The loaded phone models (shown with solid boundaries in the figure) are kept in a list in memory. All the phone models are loaded into the list before any training occurs. There is only one occurrence of each phone model in the list. For each speech file and its label sequence to be used in training, a compound model is constructed as described above. But instead of copies being made of the phone models in the list, the constituent parts of the compound model are links to the phone models in the list. When the compound model is trained, the relevant models in the list are being trained because they are part of the compound model. Once all the speech files have been used, all the phone models in the list should have been trained so they can be saved.

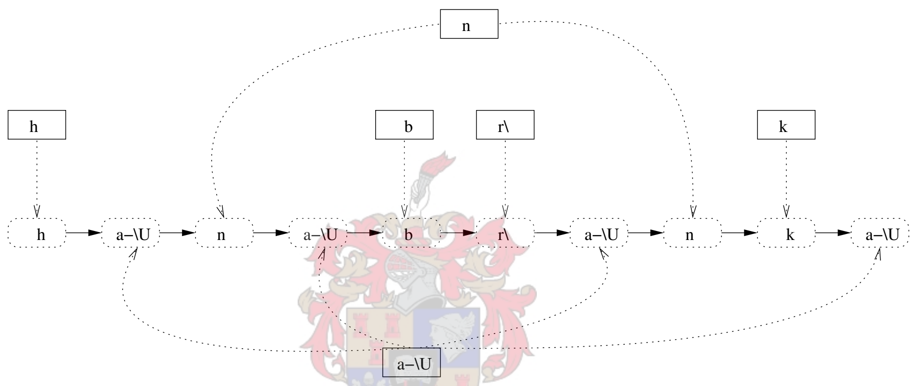

Figure 4.4: *Constructing a compound model using links to phone models.*

If some boundaries are specified and are believed to be correct, the label and feature files can be cut at the boundary locations and each part used for embedded training. The cutting is done after normalisation and feature extraction to reduce edge effects.

An early version of the program had a feature to handle optional silences. This was because the original transcriptions of the AST databases did not contain interword silences within read utterances — silences between utterances were marked. Therefore, optional silence labels were inserted automatically between words. The program handled these by placing a silence model between the final phone model of a word and the first phone model of the next word. Figure 4.5 shows an example of this. The `sil` model has been inserted between the final phone (`a-\U`) of the first word (*how*) and the first phone (`n`) of the second word (*now*). There is a loop bypassing the silence model to make it optional. This feature was removed once all optional silences were removed from the transcription files (after the transcribers had checked the transcriptions and either deleted the silences or replaced them with ordinary silence labels).

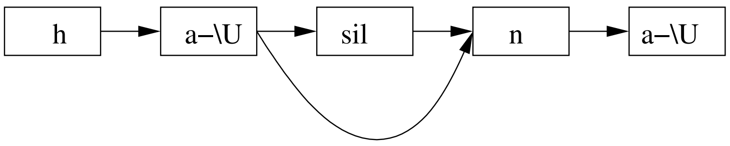

Figure 4.5: *Modifying the compound model in embedded training when optional silences are present in the transcriptions.*

Another option in the program is to work from word transcriptions and a pronunciation lexicon instead of directly from phonetic transcriptions. Figure 4.6 shows a part of the compound model for a transcription using a lexicon containing only one pronunciation for the words *how* and *brown* and two pronunciations for the word *now.* The compound model branches at the end of the word before *now* to include both pronunciations and the ends of these pronunciations join again before the next word.

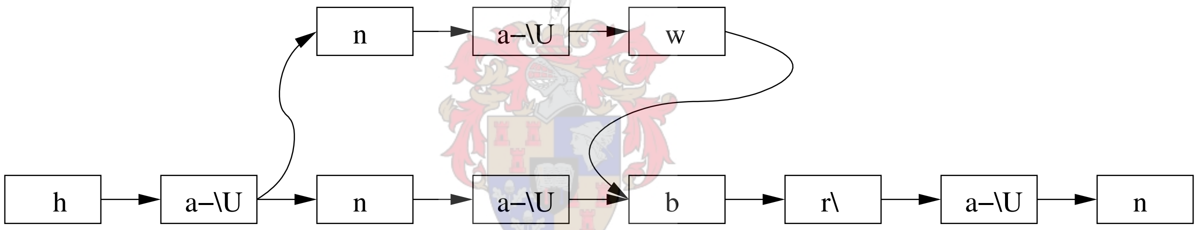

Figure 4.6: *A compound model in embedded training constructed from a pronunciation lexicon containing two pronunciations for the word* now*.*

4.3.3 Forced Alignment

Forced alignment is used to find the boundaries between labels in a known sequence. The procedure is similar to embedded training:

1. Load the model for each label in the known label sequence.
2. Form a compound model by concatenating these models in the order given by the label sequence.
3. Determine the best state sequence in the compound model.
4. Save a file containing the boundary times.

The Viterbi algorithm [4, 6, 8] finds the state sequence (through the model) that is most likely to have produced the sequence of feature vectors under consideration. The algorithm works by pretending that the vectors were generated by the model. Boundaries occur where there is a state transition between two constituent models in the compound model. State transitions internal to constituent models are ignored. The time associated with each feature vector can be determined from its position in the sequence and since the Viterbi algorithm associates each feature vector with an emitting state, a time can be assigned to each boundary. The time assigned to each feature vector is $n \times s - c$ where n is the position of the vector in the sequence (starting at 0), s is the frame shift and c is a constant that depends on the initial position of the centre of the window when the features were calculated. The two most likely values for c are $\frac{s}{2}$ and 0.

The forced alignment program was adapted for optional silences in the same way as described for embedded training. There is also an option for doing forced alignment from a pronunciation lexicon and word transcriptions with the compound model constructed in the same way as for embedded training.

4.3.4 Manual Segmentation Results

Section 2.2 mentions that transcribers described in [2] get a 6 ms average error on a small database. Expert transcribers were used and they were given guidelines for boundary placement in various contexts. It seems reasonable to assume that each boundary placement was carefully considered, so one can regard 6 ms as an upper limit on manual transcription accuracy. To have another reference, I aligned 30 TIMIT files manually after using a program to output label files containing equally spaced boundaries. The average error between my files and the ones supplied with TIMIT was 8.14 ms.

As discussed in section 2.1, some examples of mean error between automatic boundaries and manual reference boundaries are 14 ms [7] and 27 ms [2]. These values can serve as a comparison for our automatic alignment system. It is important to note however that these results were obtained on high-quality speech. The AST and NTIMIT results that are reported later in this chapter are for telephone speech. It can be expected that telephone channel noise negatively impacts alignment results. In [20], which uses telephone speech, a boundary is considered correctly placed if it lies within 45 ms of its reference boundary.

4.3.5 Significance Tests

The experiments in this chapter will be compared primarily by the difference of their means. We need a method of knowing whether a smaller mean error for experiment A than for experiment B implies that method A is better than method B. The lower mean error indicates that the method is better when applied to the test set, but we wish to know whether the method will produce better results in general (i.e. on any set similar to the training and test sets). The NTIMIT test set is a sample of the population of all English telephone speech files we might wish to segment. The applicability of a result on a sample to the entire population depends on the size of the sample.

Boundaries are not statistically independent because phone models have durations so moving one boundary closer to another will probably cause the second boundary to move away. But to have an idea of whether results are statistically significant, we will assume independence of the boundaries. Walpole [24, pg. 311] gives the following formula for testing whether the null hypothesis of $\mu_1 = \mu_2$ is true.

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{4.1}$$

where x is the sample mean, s^2 is the sample variance, n is the size of the sample and μ is the population mean. t is a value of the t distribution [24, pg. 219]. If $-t_{\alpha/2} < t < t_{\alpha/2}$ the null hypothesis is accepted; otherwise it is rejected. We can use this formula to test whether the mean errors produced by two experiments have a statistically significant difference. The results will be significant if the null hypothesis is rejected. α gives the confidence of this assertion — we reject or accept the null hypothesis with a $(1-\alpha)100\%$ confidence.

Comparisons will always be between experiments that have the same number of boundaries to place, so equation 4.1 then simplifies to

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2 + s_2^2}{n}}} \tag{4.2}$$

In our case, we use the absolute value of $x_1 - x_2$. The original pdf producing $x_1 - x_2$ is symmetrical about the the y-axis. Taking the absolute value causes the negative part of the pdf to be reflected about the y-axis therefore doubling the height of the positive part of the pdf (and the negative part goes to zero). Because there is only one side to the distribution, we must use t_α instead of $t_{\alpha/2}$. Therefore, the null hypothesis is accepted if $t < t_\alpha$ and equation 4.2 then becomes

$$t = \frac{|\bar{x}_1 - \bar{x}_2|}{\sqrt{\frac{s_1^2+s_2^2}{n}}} \tag{4.3}$$

If we divide the inquality be $\sqrt{n}$, we accept the null hypothesis if

$$\frac{|\bar{x}_1 - \bar{x}_2|}{\sqrt{s_1^2 + s_2^2}} < \frac{t_\alpha}{\sqrt{n}} \tag{4.4}$$

We will call the left hand side of the above inequality the modified t value.

n will be very large in all the experiments, so we can take the value for t_α to be the one for an infinite number of degrees of freedom. This is because the number of degrees of freedom is proportional to n [24, pg. 311] and the t_α values increase (with increasing degree of freedom) rapidly for low degrees of freedom but slowly for large degrees of freedom. To have a 90% confidence in our decision, α must be 0.1 which gives a value of 1.282 for t_α in equation 4.4.

4.4 TIMIT Experiments

This section gives results for alignments on the TIMIT database. The purpose of the experiments was to find a good method for segmenting a set of files for which the phonetic label sequence is known.

There are 1680 files in the TIMIT test set containing 64145 segments, so the number of boundaries to be placed is $64145 - 1680 = 62465$. This is the value for n to substitute into equation 4.4. The right hand side of the inequality becomes 0.0051294.

Before presenting the experimental results, some problems with the TIMIT label files are discussed. Some of these problems reduced the accuracy of the segmentations.

4.4.1 Problems with the TIMIT Transcriptions

Figures 4.7, 4.8 and 4.9 show some examples of imprecise labelling which caused alignment errors. Figure 4.7 shows the final section of a file with five labels visible: `aw tcl pau hv h#`. The segment marked as `pau` contains breath noise. Usually `pau` is used for a silent pause,

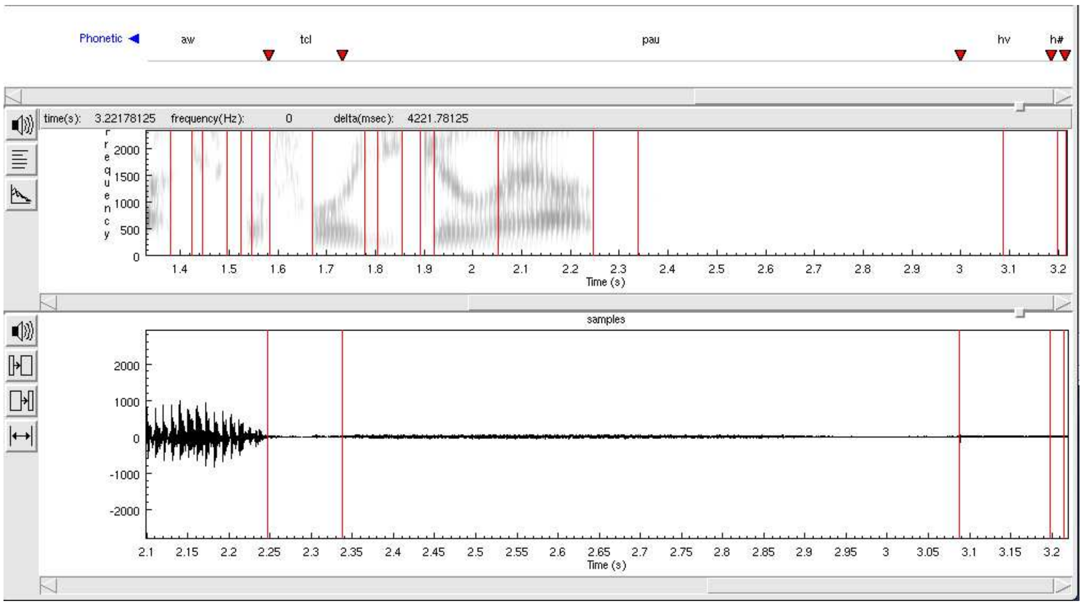
Phonetic
aw
tcl
pau
hv
h#
time(s): 3.22178125 frequency(Hz): 0 delta(msec): 4221.78125
Time (s)
samples
Time (s)

Figure 4.7: *Speaker noise in pau segment and incorrectly used hv label.*

so the `pau` model will not give a good score for this segment of speech. The following label (`hv`) marks a section of the file where there is no phone present. This is an error because this symbol represents a voiced `h`.

In figure 4.8, the final two labels are `t h#`. The `t` has a long section of aspiration [11] after the burst which is marked as part of the `h#` segment. Like `pau`, `h#` usually marks a silence section.

In figure 4.9, there is a sucking sound inside the `h#` segment.

Another problem is that the final boundary of the label file is not always placed coincident with the end of the acoustic file. The largest such error is over 1.6 s. Fortunately, the decision to ignore final boundary placement in the error calculation eliminated this problem.

The sizes of the problem segments in three figures range from 600 ms to 900 ms. The alignment procedures tend to place boundaries at transitions between sounds. If a boundary for one of these segments is misplaced, an error of at least 600 ms will be made. Therefore, in the discussion of experimental results, maximum error is only mentioned, but the mean is used for comparing the experiments. These large errors can provide a large number of features (compared to the length of a phone) for training the wrong model, but because they occur infrequently, the errors should be overwhelmed by features from correct segments. Of course, if embedded training is used to train the models, then

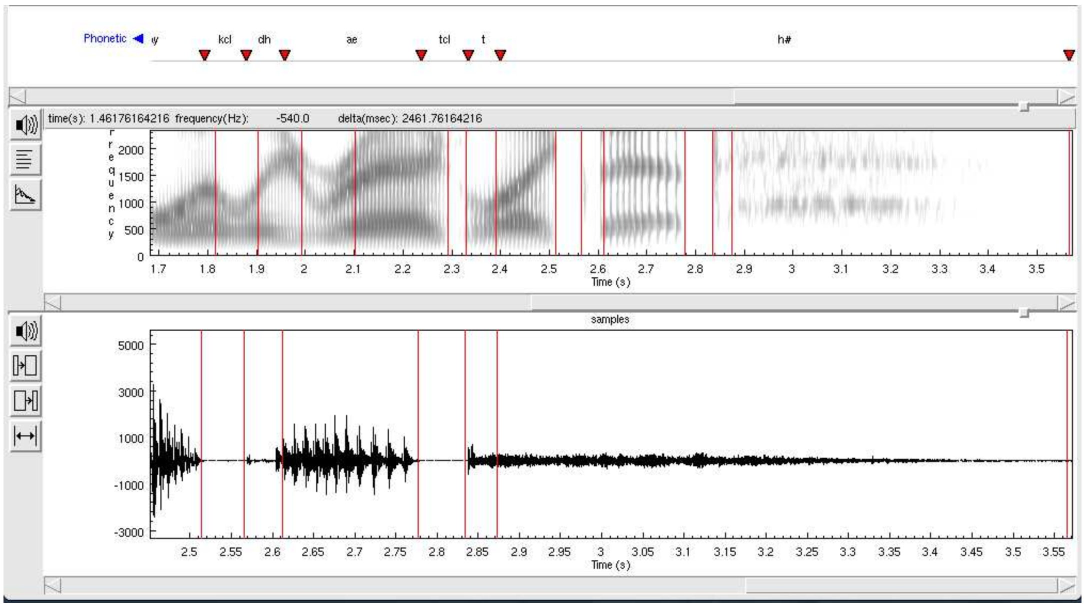
Phonetic
y
kcl
dh
ae
tcl
t
h#
time(s): 1.46176164216 frequency(Hz): -540.0 delta(msec): 2461.76164216
frequency
Time (s)
samples
Time (s)

Figure 4.8: *Aspiration of t in h# segment.*

the boundaries are ignored and these segmentation errors should not negatively affect model training.

4.4.2 Feature Comparison

The first stage of the aligner implementation involved evaluating various feature extraction methods and model structures. There is a considerable amount of literature comparing these methods for recognition, but the best features for recognition might not be the best features for segmentation. To elaborate on this point, assume that phones could be divided into classes so that the phones in each class have identical duration characteristics unique to that class. The best feature set for recognition would have to include features that would differentiate between phones within a class, whereas a feature set that only separates the classes would be preferable for segmentation, because the length of the phone is all that must be determined and this is modelled by the class. Adding unnecessary power to a feature set usually leads to a decrease in performance because the more complex model will have more parameters to train. Training more parameters with the same quantity of data results in less data for each parameter.

The full set of experimental results is given in table 4.2. The details of the experiments can be found in table 4.3. The corresponding rows of table 4.2 and table 4.3 refer to the

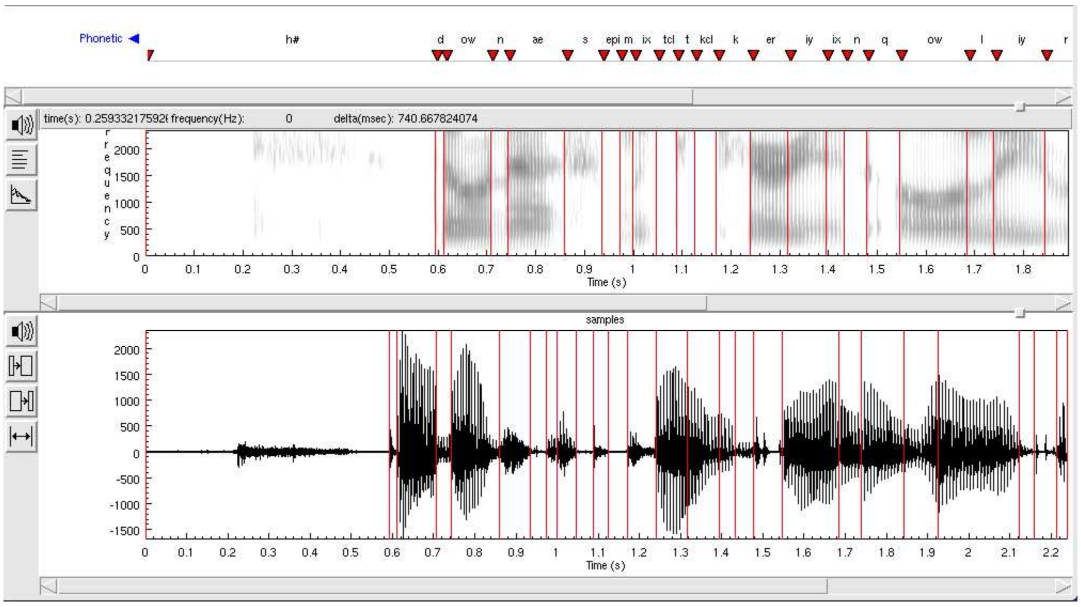

Figure 4.9: *Speaker noise in h# segment.*

same experiment and the experiment number is given in the first column of both tables. The experiments compare different combinations of features and model structures. All the models had a three state left-to-right structure, but the number of forward links was varied.

The columns in table 4.2 are as follows:

- experiment number
- maximum absolute error
- mean absolute error
- standard deviation of absolute error

These measures are described in section 1.2.2.

The columns in table 4.3 are as follows:

- experiment number
- feature type
- feature qualifier
- frame shift

Table 4.2: *Summary of experimental scores for basic feature experiments performed on TIMIT.*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
01	517.50	11.86	16.17
02	847.50	11.13	14.50
03	837.50	12.48	15.65
04	842.50	11.79	14.45
05	747.50	11.54	14.02
06	837.50	10.85	12.41
07	867.50	12.64	13.90
08	412.81	12.48	17.49
09	852.50	11.50	15.39
10	852.50	11.69	14.90
11	847.50	11.29	14.99
12	837.50	11.57	14.88
13	382.81	11.60	15.91
14	862.50	10.95	14.50
15	847.50	10.78	14.08
16	842.50	11.49	14.46
17	862.50	11.03	14.19
18	837.50	10.74	12.17
19	847.50	11.82	16.72
20	385.06	12.11	15.65
21	852.50	11.49	16.00
22	877.50	12.09	15.82

- frame length
- type of Gaussian in the mixtures
- number of mixture components
- skips ahead

The *feature type* column states whether the features are based on mel frequency cepstral coefficients (MFCCs) [4, pg. 380–385] or linear predictive cepstral coefficients (LPCCs) [28, pg. 59].

The *feature qualifier* column gives further information about the feature vectors. The first character of the entry is either a *0* or an *E*. This gives the choice between two types of energy measure. *E* means log energy was used and *0* means that the zeroth cepstral coefficient was used. The zeroth cepstral coefficient is calculated using the formula [28, pg. 60]:

$$c_0 = \sqrt{\frac{2}{N}} \sum_{j=1}^{N} \log m_j \tag{4.5}$$

where m_j is the output of the jth mel filterbank and N is the number of filterbanks. m_j is calculated by taking the Fourier transform of the portion of the signal in the window (used to calculate the feature vector) and multiplying the magnitude of each frequency component by the value of the filter at that frequency (values outside the filter band for the jth filter will be zero after this step). The scaled values in the magnitude spectrum are then summed to produce m_j. The energy value is calculated using the formula [28, pg. 61]:

$$E = \log \sum_{n=1}^{N} s_n^2 \tag{4.6}$$

where s_n is the nth sample in the window and N is the number of samples in the window. *D* indicates that delta parameters [4, pg. 385] were used and *A* indicates that acceleration parameters [28, pg. 61–62] were used. The *frame shift* and *frame length* are explained in section 4.3. The *type of Gaussian* column states whether the Gaussian pdfs in the mixture models had diagonal covariance matrices (d) or full covariance matrices (f). The *mixture components* column gives the number of Gaussian pdfs in each mixture model. If the *skips ahead* column has an entry of *0*, then each emitting state in the HMM is only linked to itself and the following state. The structure of this type of model is shown in figure 4.10. The large circles represent emitting states and the small circles represent entry and exit null states. If the column entry is *1*, the first state is linked to itself, the second state and the third state, the second state is linked to itself, the third state and the exit null state, and the begin null state is linked to the first and second states (see figure 4.11).

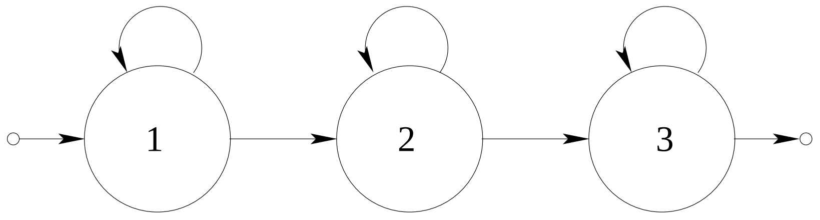

Figure 4.10: *HMM structure when no states may be skipped.*

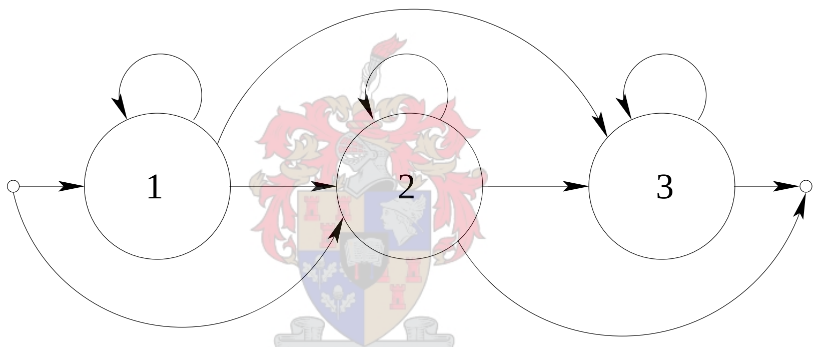

Figure 4.11: *HMM structure when at most one state may be skipped during a single transition.*

Table 4.4 compares the results using a 10 ms frame shift and 25 ms frame length with those using a 5 ms shift and 20 ms frame length. The numbers in the first column are the same as those in table 4.3. These numbers are also given in the table caption in parentheses. For the pair 02 and 06, the models had eight Gaussians in their mixture pdfs and the features used the zeroth cepstral coefficient. For the pair 15 and 18, the models had sixteen Gaussians in their mixtures and the features used energy. The 10 ms frame shift and 25 ms frame length combination performs better than the combination of a 5 ms frame shift and a 20 ms frame length, but the majority of experiments in this section used a 5 ms frame shift to lower the minimum theoretical error range (which is from zero to half the frame shift — see section 4.3). The better performance of the 10 ms frame shift could have two explanations:

Table 4.3: *Descriptions of basic feature experiments performed on TIMIT.*

	feature type	feature qualifier	frame shift	frame length	type of Gaussian	mixture components	skips ahead
01	MFCC	0DA	5ms	20ms	d	1	0
02	MFCC	0DA	5ms	20ms	d	8	0
03	MFCC	0DA	5ms	20ms	f	1	0
04	MFCC	0DA	5ms	20ms	f	2	0
05	MFCC	0DA	10ms	25ms	d	1	0
06	MFCC	0DA	10ms	25ms	d	8	0
07	MFCC	0DA	10ms	25ms	f	1	0
08	MFCC	0DA	5ms	20ms	d	1	1
09	MFCC	0D	5ms	20ms	d	8	0
10	MFCC	0D	5ms	20ms	f	2	0
11	MFCC	ED	5ms	20ms	d	8	0
12	MFCC	ED	5ms	20ms	f	2	0
13	MFCC	EDA	5ms	20ms	d	1	0
14	MFCC	EDA	5ms	20ms	d	8	0
15	MFCC	EDA	5ms	20ms	d	16	0
16	MFCC	EDA	5ms	20ms	f	2	0
17	MFCC	EDA	5ms	25ms	d	8	0
18	MFCC	EDA	10ms	25ms	d	16	0
19	LPCC	ED	5ms	20ms	d	8	0
20	LPCC	ED	5ms	20ms	f	2	0
21	LPCC	EDA	5ms	20ms	d	8	0
22	LPCC	EDA	5ms	20ms	f	2	0

Table 4.4: *Comparison between frame shifts of 5 ms (02, 15) and 10 ms (06, 18).*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
02	847.50	11.13	14.50
06	837.50	10.85	12.41
15	847.50	10.78	14.08
18	837.50	10.74	12.17

Table 4.5: *Comparison between frame lengths of 20 ms (14) and 25 ms (17).*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
14	862.50	10.95	14.50
17	862.50	11.03	14.19

1. The larger frame length (25 ms vs 20 ms) provides more samples for estimating the feature vectors.

2. The 5 ms frame shift produces more vectors for the same section of the speech file. Therefore, for a section containing a transition between two phones, the vectors for the 5 ms frame shift will change more gradually (assuming the duration of the transition is larger than 10 ms). An HMM with only three states will find it more difficult to model a gradual transition than a more rapid transition.

In section 4.4.3, experiments are performed where the only difference is in the frame shift — the frame lengths are identical. These experiments will provide more insight into this question.

Table 4.5 compares a frame length of 20 ms with a frame length of 25 ms. Both experiments use a 5 ms frame shift. The mean is slightly better for the 20 ms experiment. The difference in means of 0.08 ms is very small. Standard deviations of 14.50 and 14.19 produce a value of 0.0039432 for the modified t value which means that we accept the null hypothesis and conclude that the two methods are equivalent.

Table 4.6 compares LPC cepstral results (21) with mel cepstral results (14). The

Table 4.6: *Comparison between LPCC (21) and MFCC (14) based features.*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
14	862.50	10.95	14.50
21	852.50	11.49	16.00

Table 4.7: *Comparison between zeroth cepstral coefficient (02) and energy (14).*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
02	847.50	11.13	14.50
14	862.50	10.95	14.50

mean is better for the MFCCs. The difference in means is 0.54 ms and together with standard deviations of 14.50 and 16.00 this gives a modified t value of 0.025008. This is a significant result. This result corresponds to the usage in speech recognition where MFCCs are preferred to LPCCs.

Table 4.7 compares the use of the zeroth cepstral coefficient (02) with that of the energy (14). Energy gives the better mean with a difference of 0.18 ms. A standard deviation of 14.50 for both experiments gives a modified t value of 0.0087779. This result is significant. This result also corresponds to the usage in speech recognition where energy is used more often than the zeroth cepstral coefficient.

Table 4.8 shows the advantage of using acceleration parameters. The table compares the mean errors for six pairs of experiments. Each mean value is followed by its experiment number in parentheses. The first column gives a description of the shared characteristics of the experiments in the pair. Acceleration parameters improve the results in five out of the six cases (the exception being the pair 04 and 10). This is as expected because we are looking for transitions, so features that measure transitions better should give better results.

Table 4.9 compares the use of one-ahead skip models with the use of models with no skips. The model without skips has a better mean. The difference in means is 0.62 ms.

Table 4.8: *Comparison of mean errors for experiments with and without acceleration. Experiment numbers in parentheses.*

	with A	without A
MFCC 0D d8	11.13 (02)	11.50 (09)
MFCC 0D f2	11.79 (04)	11.69 (10)
MFCC ED d8	10.95 (14)	11.29 (11)
MFCC ED f2	11.49 (16)	11.57 (12)
LPCC ED d8	11.49 (21)	11.82 (19)
LPCC ED f2	12.09 (22)	12.11 (20)

Table 4.9: *Results for HMMs with (08) and without (01) skip transitions.*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
01	517.50	11.86	16.17
08	412.81	12.48	17.49

Standard deviations of 16.17 and 17.49 give a modified t value of 0.026029. This is significant. The no skips models probably perform better because they are enforcing a greater minimum duration and therefore limiting the error that can be made by making a segment too short. Very few phone examples will be shorter than the minimum durations of these models (both no skips and one skip), so on average the results should be better if the models are prevented from producing very short segments.

Tables 4.10 and 4.11 compare types of mixture pdfs. All experiments use a 5 ms frame shift and a 20 ms frame length. The MFCC 0DA experiments are for single diagonal Gaussians (d1), mixtures of eight diagonal Gaussians (d8), single full-covariance Gaussians (f1) and mixtures of two full-covariance Gaussians (f2). The MFCC EDA experiments have mixtures of sixteen diagonal Gaussians (d16) instead of f1. Both sets have their best results for the largest diagonal Gaussian mixtures (8 for MFCC 0DA and 16 for MFCC EDA).

The experimental characteristics discussed above are not independent of each other,

Table 4.10: *Comparison of types of mixtures for MFCC 0DA features. Entries are for d1 (01), d8 (02), f1 (03) and f2 (04).*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
01	517.50	11.86	16.17
02	847.50	11.13	14.50
03	837.50	12.48	15.65
04	842.50	11.79	14.45

Table 4.11: *Comparison of types of mixtures for MFCC EDA features. Entries are for d1 (13), d8 (14), d16 (15) and f2 (16).*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
13	382.81	11.60	15.91
14	862.50	10.95	14.50
15	847.50	10.78	14.08
16	842.50	11.49	14.46

so it is not guaranteed that the best setup will be produced by using the best choice from each comparison. But testing all combinations of all the characteristics would be too time consuming, so it was assumed that combining the best choices would produce the best result. Under this assumption, the best setup should be 10 ms frame shift, 20 ms frame length, MFCC EDA features and models with no skips and sixteen dimensional Gaussian mixture pdfs. The table does not contain an experiment with this description, but the best mean is for experiment 15 which has an identical description except that the frame shift is 5 ms instead of 10 ms. The 10.78 ms mean error obtained by experiment 15 can be considered reasonably good from the discussion in section 4.3.4.

4.4.3 Duration Modelling

This section describes the use of higher order HMMs [5] to model state duration. It was hoped that duration modelling might improve the results of the previous section. In higher order HMMs, a transition is not only dependent on the source and destination states, but also on states occupied before the source state. For example, in a third order model, transitions depend on the source state and the two states occupied before the source state.

The higher order models are represented by an equivalent first order model [5]. Therefore, the number of states in the model increases as the order increases (to model the transition history), but the number of pdfs remains constant. This implies that states in the equivalent first order model share pdfs. A first order left-to-right model will remain a left-to-right model when the order is increased.

The higher order models in this section are created from three state left-to-right first order models with no skips (see figure 4.12). As the order of the models increases, there is more history of the occupation of each state. Figure 4.13 shows the first order equivalent model of the second order version of the model in figure 4.12. In the figure, states 1 and 2 share a pdf, as do states 3 and 4 and states 5 and 6. If the transition from state 1 to state 3 has a larger weight than the transition from state 1 to state 2, then it is more likely that only one feature vector will be absorbed by the pdf for the first group of states. The model therefore encodes a shorter duration for the first group than if the transition from state 1 to state 3 has a smaller weight than the product of the transition from state 1 to state 2 and the transition from state 2 to state 3.

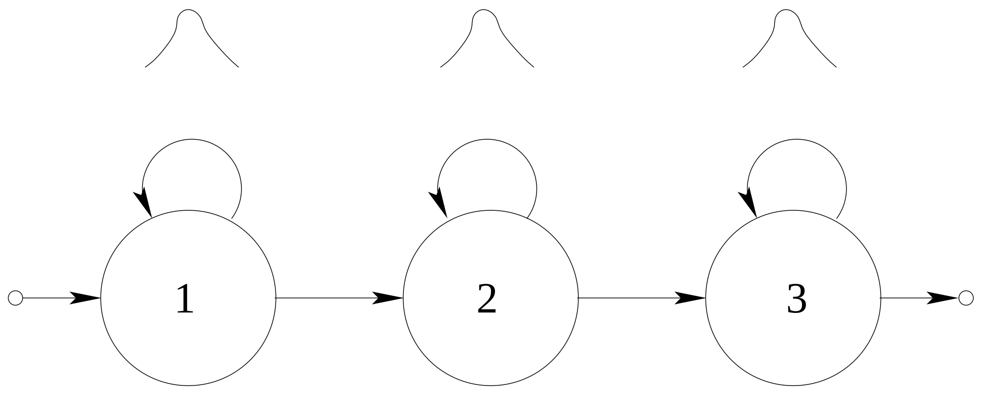

Figure 4.12: *A first order left-to-right HMM with no skips.*

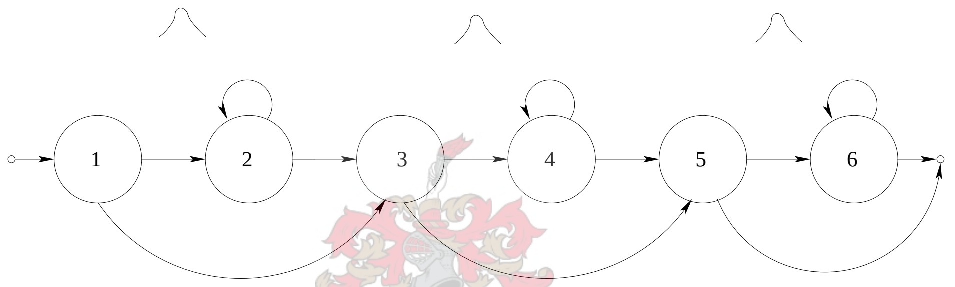

Figure 4.13: *The first order equivalent of the second order version of the HMM in figure 4.12.*

Experimental Setup

The training and testing data comprised the entire TIMIT training and testing sets. The features extracted were 12 dimensional mel cepstra with energy, delta and acceleration parameters appended — resulting in 39 dimensional feature vectors. HTK [28] was used to calculate these features. Each of the 61 phones in the TIMIT files was represented by a left-to-right HMM. Each emitting state had a mixture of eight diagonal Gaussian pdfs attached to it. The first order models had three emitting states with the only links being self loops and those to the next state. As stated above, the equivalent first order models (of the higher order models) were also left-to-right, but forward links were not only to the next state. Nevertheless, the higher order models retain the property that no pdfs may be skipped i.e. the only skips allowed in the equivalent first order models are within a group of states linked to the same pdf.

Results

The results for the experiments using a frame shift of 2 ms and a frame length of 10 ms are given in table 4.12 and those for the experiments using a frame shift of 5 ms and a frame length of 10 ms are given in table 4.13. The 5 ms frame shift produces its best results (mean error of 9.43 ms) with a fifth order model. Table 4.12 shows the results for the 2 ms frame shift improving with model order. We stopped at 14th order where the mean error had been reduced to 9.92 ms. The duration modelling has improved the results from 10.78 ms for the experiments in the previous section to 9.43 ms.

It would have been interesting to see up to which order the results for the 2 ms frame shift models continue improving. We stopped at 14th order because the processing time was getting too long and because we were most interested in the results for the 12th order and 13th order models because they should have corresponded to the fifth order 5 ms frame shift results (see the first paragraph of the next section for the explanation of this).

Table 4.12: *Duration modelling by means of higher order models. Results for frame shift of 2 ms and frame length of 10 ms.*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	685.50	11.00	18.98
2	479.50	11.01	18.14
3	479.50	10.95	17.99
4	479.50	10.81	17.68
5	479.50	10.71	17.50
6	479.50	10.61	17.28
7	479.50	10.48	16.94
8	859.50	10.38	17.10
9	807.50	10.29	16.87
10	752.50	10.18	16.55
11	759.50	10.09	16.30
12	757.50	10.01	16.07
13	757.50	9.98	16.01
14	753.50	9.92	15.61

Table 4.13: *Duration modelling by means of higher order models. Results for frame shift of 5 ms and frame length of 10 ms.*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	807.50	9.74	13.71
2	807.50	9.67	13.63
3	807.50	9.61	13.57
4	807.50	9.48	13.41
5	807.50	9.43	13.15
6	802.50	9.46	12.86
7	797.50	9.67	12.64
8	807.50	10.19	12.63
9	807.50	11.23	13.22

4.4.4 Increasing the Minimum Duration

The experiments in the previous section achieved better results with a 5 ms frame shift than with a 2 ms shift. The frame length was the same for both frame shifts, so the quantity of information available when calculating the feature vectors was the same. A model based on 2 ms frames would have to use more vectors than its 5 ms equivalent to model the same part of a speech file. Therefore, the history of the 2 ms model needs to be longer than that of the 5 ms model to cover the same time interval. Because the best result for the 5 ms frame shift is at order 5, the best result for the 2 ms frame shift should be at order 12 or 13 ($5 \times 5 \div 2 = 12.5$), and the result should be similar to the 5 ms result. But this is not the case. A possible explanation is that even though the history in the higher order models does model duration, there is still a chance for a model to recognise a short duration segment. In the three state left-to-right models with no skips used in the previous section, the minimum duration enforced by the model is 6 ms for 2 ms frame shift models and 15 ms for 5 ms frame shift models. To test whether it was this property that was causing the difference in results, the minimum duration of the 2 ms frame shift model was successively increased. Table 4.14 gives the percentages of excluded and included phone examples at various minimum duration settings.

Table 4.14: *Relation between frame shift and durations of phone examples.*

frame shift	percentage $> 3 \times$ frame shift	percentage $< 3 \times$ frame shift
2	99.99	0.01
4	99.27	0.73
5	98.06	1.94
6	96.83	3.17
10	87.19	12.81
16	68.85	31.15

Table 4.15 gives the results for a model with a minimum duration of 12 ms. The structure of the model used is shown in figure 4.14. The mean error has improved with respect to table 4.12, but the maximum error is worse. This suggests that most of the segments are longer than 12 ms.

Table 4.16 gives the results for a 14 ms minimum duration. The model for this has

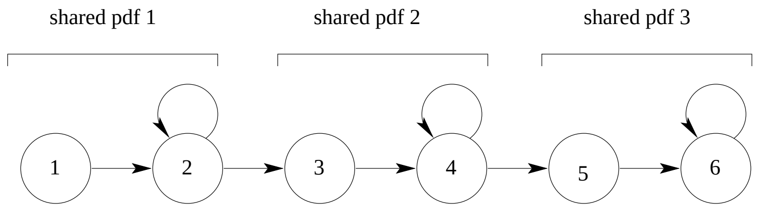

Figure 4.14: *Model structure for 12 ms minimum duration.*

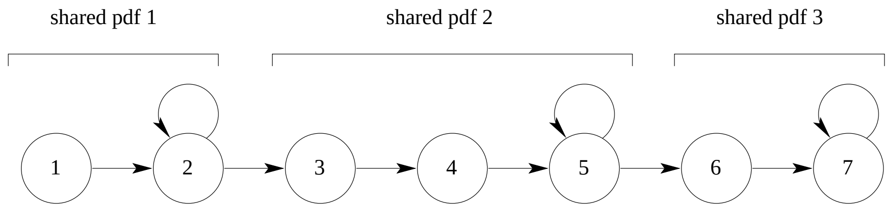

Figure 4.15: *Model structure for 14 ms minimum duration.*

three states sharing the central pdf and two states each for the left and right pdfs. The results have improved from the 12 ms case. The structure is shown in figure 4.15.

The results for a 16 ms minimum duration are given in table 4.17. The results have again improved, but they still do not match those obtained with the 5 ms frame shift even though the minimum duration for those models was 15 ms. Table 4.18 gives the results with models with minimum duration 15 ms but using a frame shift of 2.5 ms instead of 5 ms. Notice that the results are closer to the 5 ms frame shift results. This could either be due to the increase in frame shift size or that 15 ms is the best minimum duration setting (out of integer values for the duration setting and assuming that there is only one minimum).

Minimum Duration Modelling for Individual Phones

In the next experiment, each phone model had its own minimum duration setting which was chosen so that 1% of the examples of each phone in the training set had durations less than the minimum setting. This experiment was only run using first order models. The result for the 2 ms frame shift models is given in table 4.19 and the result for the 5 ms frame shift is given in table 4.20. Both results are the best scores obtained for the frame shifts under consideration.

Because the time duration of an utterance to be aligned is fixed, imposing minimum durations on phone models in the utterance implies an implicit maximum duration for

Table 4.15: *Results for models with 12 ms minimum duration (2 ms frame shift).*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	683.50	10.41	17.60
2	865.50	10.45	17.40
3	865.50	10.44	17.40
4	865.50	10.36	17.10
5	865.50	10.39	17.35
6	773.50	10.22	16.65
7	857.50	10.14	16.61
8	853.50	10.05	16.40
9	849.50	9.93	16.16
10	847.50	9.87	16.00
11	847.50	9.81	15.85
12	825.50	9.75	15.59
13	825.50	9.73	15.51

each phone model too. This maximum duration is calculated by subtracting the sum of the minimum durations of the other phones in the utterance from the length of the utterance. But this maximum duration will probably be larger than it would have been if maximum duration had been modelled explicitly. Figure 4.16 shows an HMM that models both minimum and maximum duration. The HMM has two pdfs. The numbers inside the circles indicating states in the figure indicate which pdf is attached to each state. The small circles represent null states. The first three non-null states are attached to the first pdf and the last three non-null states are attached to the second pdf. Because of the link structure, each pdf will absorb a minimum of two and a maximum of three feature vectors. Maximum phone durations were not explicitly modelled in this work.

Table 4.16: *Results for models with 14 ms minimum duration (2 ms frame shift).*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	681.50	10.25	17.24
2	479.50	10.22	16.09
3	771.50	10.23	16.57
4	863.50	10.18	16.69
5	863.50	10.15	16.60

Table 4.17: *Results for models with 16 ms minimum duration (2 ms frame shift).*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	675.50	10.08	16.89
2	771.50	10.11	16.42
3	771.50	10.09	16.33
4	771.50	10.03	16.21
5	771.50	10.02	16.20

Table 4.18: *Results for models with 15 ms minimum duration (5 ms frame shift).*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	835.00	9.87	15.54
2	835.00	9.88	15.52
3	835.00	9.86	15.49
4	835.00	9.80	15.42
5	832.50	9.76	15.24

Table 4.19: *Results for 2 ms models with minimum duration customised for each phone.*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	583.50	9.52	14.51

Table 4.20: *Results for 5 ms models with minimum duration customised for each phone.*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	802.50	9.34	12.78

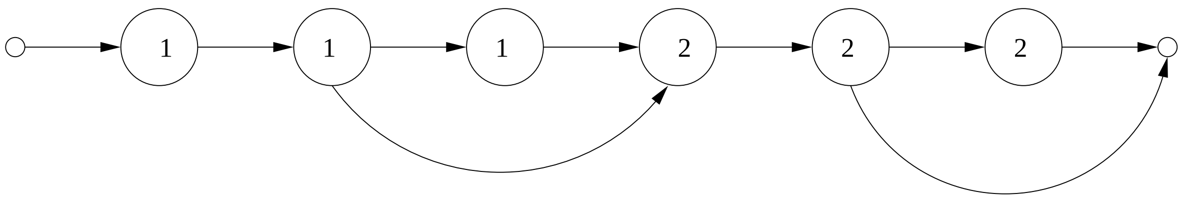

Figure 4.16: *An HMM that models both minimum and maximum duration.*

4.4.5 Special Features

This section describes some experiments that were performed to determine whether adding extra information to the feature vectors described in section 4.4.2 would improve results. Table 4.21 describes the features and table 4.22 gives the scores. In these experiments, an extra element was appended to the basic feature vectors. This increased the dimension of the pdfs by one. The first column of table 4.21 gives the experiment number which has the same meaning as the first column in table 4.22. The third column gives the length of the frame used to calculate the feature. This frame length is independent of the one used to calculate the base features (but the frame shift must be the same so that there is one base vector for each new feature element calculated). The fourth column indicates whether the signal was preemphasized [4, pg. 329–330] before the extra feature elements were calculated. The method for calculating the features is given in the second column. The entries have the following meanings:

lfhf This is the natural logarithm of the ratio of the energy in the low frequency part of the signal in the frame to its high frequency part. The low frequency part is taken from 100 Hz to 1 kHz and the high frequency part from 1 kHz to 3 kHz.

zc This is the zero crossing density and is proportional to the number of zero crossings in the frame divided by the number of samples in the frame. A zero crossing occurs when a sample and its successor have different signs.

power This is the natural logarithm of the power in the frame.

The basic feature vectors were MFCCs with thirteen components (including the zeroth component) and delta and acceleration parameters.

The first row of table 4.22 gives the result for vectors that do not have any information appended. None of the experiments listed in the tables produces any significant improvement over this baseline result.

Table 4.21: *Descriptions of special feature experiments on TIMIT database.*

	feature type	frame length [ms]	preemphasis
01	none	-	-
02	lfhf	5	no
03	lfhf	10	no
04	lfhf	5	yes
05	lfhf	10	yes
06	lfhf	20	yes
07	lfhf	100	yes
08	zc	10	no
09	zc	16	no
10	zc	40	no
11	zc	100	no
12	zc	10	yes
13	power	5	no
14	power	10	no
15	power	5	yes
16	power	10	yes
17	power	16	yes
18	power	40	yes

Table 4.22: *Scores for special feature experiments on TIMIT database.*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]
01	855.5	11.5911	19.4573
02	855.5	11.6259	19.6169
03	855.5	11.538	19.6272
04	855.5	11.6558	19.6739
05	491.5	11.6208	19.2341
06	855.5	11.6836	19.962
07	885.5	11.8463	20.0752
08	855.5	11.6105	19.8738
09	885.5	11.6108	19.7077
10	885.5	11.7193	19.8862
11	855.5	11.7738	19.6893
12	885.5	11.6558	19.9267
13	859.5	11.7795	20.2743
14	855.5	11.6093	19.7421
15	494.625	11.6268	19.363
16	835.5	11.564	19.7414
17	837.5	11.6112	19.6304
18	507.5	11.7815	19.6867

4.4.6 Grouping Phones

Two experiments were performed to determine whether grouping phones into classes could improve the results. Using a common model for several phones has the benefit of increasing the training data for the model. If the phones are significantly different, however, the model will describe each phone less accurately and the results should deteriorate.

In the first experiment, thirteen groups were used. Table 4.23 gives the group names in its first column and the phones in each group in the second. The result is given in table 4.24. The mean error of 21.75 ms is far worse than the mean for the base system in the previous section (11.59 ms).

The second experiment had twenty four groups which are given in table 4.25. The result is given in table 4.26. The mean error of 16.39 ms is better than that of the previous experiment, but still not as good as the case where no grouping is done. One may therefore deduce that TIMIT has enough training data for its phones and that they are different enough for merging them to decrease modelling accuracy.

4.4.7 Male/Female Split

In the final TIMIT experiment, the training and testing files were separated according to the gender of the speaker. A separate set of phone models was trained for each gender and files were segmented using the gender specific models. Table 4.27 gives the distribution of files in the TIMIT training and testing sets between the two genders. Because over 70% of the training data is from male speakers, one would expect the male results to be better than the female results. Table 4.28 shows that this is not the case. The female result is almost the same as the base result of 11.59 ms. The male result is worse. The third row of table 4.28 gives the combination of the male and female scores which represents the male results more because male speakers constitute two thirds of the test set.

Table 4.23: *TIMIT phones combined into 13 groups.*

Group	Phones in group
clos	bcl, dcl, gcl, pcl, tcl, kcl
burst	b, d, g, p, t, k, dx
glot	q
afric	jh, ch
fric	s, sh, z, zh, f, th, v, dh
nas	m, n, ng, em, en, eng, nx
glsem	l, r, w, y, el
whisp	hh, hv
vowel	iy, ih, eh, ae, aa, ah, ao, uh, uw, ux, er, ax, ix, axr, ax-h
diph	ey, aw, ay, ow, oy
pau	pau
epi	epi
sil	h#

Table 4.24: *Results for combining TIMIT phones into 13 groups.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
1734.69	21.75	48.02	6.38

Table 4.25: *TIMIT phones combined into 24 groups.*

Group	Phones in group
clos	bcl, dcl, gcl, pcl, tcl, kcl
voiburst	b, d, g, dx
unvburst	p, t, k
glot	q
afric	jh, ch
unvfric	s, sh, f, th
voifric	z, zh, v, dh
nas	m, n, ng, nx
sylnas	em, en, eng
liquid	l, r
glide	w, y
el	el
hh	hh
hv	hv
shfront	ih, eh, ae
shmid	ah, ax, ix, axr, ax-h
shback	aa, uh
lofront	iy
lomid	er
loback	ao, uw, ux
diph	ey, aw, ay, ow, oy
pau	pau
epi	epi
sil	h#

Table 4.26: *Results for combining TIMIT phones into 24 groups.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
1734.69	16.39	36.44	5.87

Table 4.27: *Gender distribution of TIMIT files.*

	male	female	total
train	3260 (70.56%)	1360 (29.44%)	4620
test	1120 (66.67%)	560 (33.33%)	1680

Table 4.28: *Results for male/female split.*

	Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
male	859.50	12.83	22.99	5.56
female	470.06	11.60	21.26	5.12
combined	859.50	12.41	22.43	5.50

4.5 NTIMIT Experiments

The experiments in this section are performed using the NTIMIT database which contains a telephone speech version of the TIMIT files. Because the initial audio files are the same, the label files for the two databases are identical. This means that the only change from the TIMIT experiments is the lower bandwidth of the telephone channel (the upper bound is 3 kHz vs 8 kHz for TIMIT) and higher noise levels due to noise on the telephone channels.

The experiments in this section are on the following data:

1. the original TIMIT files

2. lowpass filtered and subsampled TIMIT files

3. lowpass filtered and subsampled NTIMIT files

The purpose of the first experiment is for comparison with the other two. The same experiment was performed in section 4.4.3, but the software had changed so the experiment was redone to allow for a more valid comparison. The second experiment shows the effect of the lower bandwidth of a telephone channel without the influence of channel noise. The third experiment then shows the additional effect of the channel noise. Like TIMIT, NTIMIT is sampled at 16 kHz even though the only useful frequency components in the files will be below 3 kHz.

Table 4.29 shows the alignment results on TIMIT where all the available frequency components (0 Hz to 8 kHz) were used. The audio files were preemphasised and thirteen dimensional MFCC feature vectors were calculated from them. Delta and acceleration parameters were appended to the feature vectors, resulting in thirty nine dimensional vectors. These were used to train three state left-to-right phone models with eight mixture diagonal Gaussian pdfs. The frame shift was 2 ms and the frame length was 10 ms. The results obtained in the duration modelling section are reproduced in table 4.30. The mean error has increased from 11.00 ms to 11.44 ms and the maximum error has increased from 685.50 ms to 863.50 ms.

Table 4.31 gives the alignment results on TIMIT files that have been lowpass filtered to remove frequency components above 4 kHz. They were also subsampled at 8 kHz. The median error is exactly the same as before (5.5 ms) and the mean and has increased slightly (11.44 ms to 11.61 ms). The maximum error has dropped from 863.50 ms to

Table 4.29: *Alignment results on TIMIT files for a 2 ms frame shift and a 10 ms frame length.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
863.50	11.44	19.53	5.50

Table 4.30: *Copy of first order modelling results from duration modelling experiments for frame shift of 2 ms and frame length of 10 ms.*

model order	maximum error [ms]	mean error [ms]	standard deviation [ms]
1	685.50	11.00	18.98

790.50 ms. This is a decrease of 8.45%. According to the mean and median errors, the high frequency components do not make much difference to the results. The maximum error has decreased, but the results for it have been very variable and so they should not be used for making deductions.

Table 4.32 shows the results for the alignment on NTIMIT. The same setup was used as in the previous experiment. The effect of the channel noise is very evident from these results. The mean error has increased from 11.61 ms to 15.10 ms, the median has increased from 5.50 ms to 6.31 ms and the maximum error has increased from 790.50 ms to 1429.00 ms.

Table 4.31: *Alignment results on lowpass filtered and subsampled TIMIT files.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
790.50	11.61	21.04	5.50

Table 4.32: *Alignment results on lowpass filtered and subsampled NTIMIT files.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
1429.00	15.10	33.24	6.31

4.6 Segmenting the African Speech Technology Databases

4.6.1 Method

The segmentation programs described above have been used on several of the AST databases. The procedure used is as follows:

1. A list is made of the phone occurrences in the label files and it is ordered according to phone occurrence count. The least frequent phones are then mapped to more frequent ones so that the least frequent phone occurs at least 100 times. This may also involve splitting diphthongs or triphthongs. The purpose of this step is to have enough training data for each phone model.

2. A model is created for each remaining phone label. The models we use are three-state left-to-right models with no skips and diagonal covariance Gaussian mixture model pdfs. All pdfs have eight mixture components.

3. A boundary is placed between each phone in the label files. These boundaries are equally spaced between any boundaries placed by the transcribers, or the start or end of the file.

4. Feature files are created from the audio files. The feature type is thirteen dimensional MFCCs with delta and acceleration elements. The frame length is 10 ms and the frame shift is 2 ms. The audio files are preemphasized before features are extracted.

5. Each phone model is trained on the sections of the feature file between the equally spaced boundaries around labels corresponding to the phone model. This is used as an initialisation for the model.

6. The phone models are trained using embedded training. The boundaries placed by the transcribers act as fixed boundaries in the embedded training and the equally spaced boundaries are ignored.

7. The trained models are used in a forced alignment of the database. The manually placed boundaries are again fixed in this step.

4.6.2 Results

In order to evaluate this method, 5% of the EE database was manually aligned on the phone level. The EE database contains 271 calls in South-African English by mother-tongue speakers. Table 4.33 gives the call distribution between male and female and between mobile phone and land-line.

Table 4.33: *Statistics for calls in the EE database.*

	training set	test set
mobile phone	27	2
land-line	230	12
male	91	5
female	166	9
total	257	14

The calls in the test set were divided into 552 utterances before being manually transcribed. There are 9645 segments in these files, so this gives $n = 9645 - 552 = 9093$ for the denominator in equation 4.4. The right hand side of the inequality becomes 0.013444.

As a benchmark, the NTIMIT models were used to align the EE test set. The same feature extraction methods and normalizer were used on the EE test set as for the NTIMIT data. Table 4.34 gives the results. The mean has increased from 15.10 ms to 67.90 ms and the median from 6.31 ms to 20.61 ms. This is a very significant degradation in performance. Although both the AST and NTIMIT databases are telephone speech, there are some differences which contribute to this bad result:

- The accent difference (United States vs South African) means that a segment label in the EE set and a model trained on NTIMIT might represent quite different sounds.

Table 4.34: *Alignment results for EE test set using models trained on NTIMIT.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
2099.50	67.90	144.74	20.61

Table 4.35: *Comparison of median durations for some phones in NTIMIT and EE. Durations are in ms.*

phone	NTIMIT	EE
s	110.31	144.15
sh	115.00	165.82
z	78.12	74.49
f	102.19	129.25
m	60.06	80.17
n	50.06	93.22
l	57.44	66.00
r	58.38	57.18
w	64.13	87.56
y	60.03	87.91
iy	82.12	22.48
ax	46.25	63.38

- There is a lot more background noise in the AST databases than in NTIMIT.
- The phone durations between the two sets differ (see table 4.35). Phones like `r` and `z` have similar durations, but others like `w` and `sh` have very different durations. The mean duration of `iy` in the NTIMIT set is almost four times larger than the value in the EE set.

Table 4.36 gives the results using the segmentation procedure described in section 4.6.1. The mean of 67.90 ms has decreased to 40.58 ms and the median has decreased from 20.61 ms to 17.76 ms.

In order to better initialise the models, the EE training set was aligned using the EE

Table 4.36: *Results for EE test set alignment using models not initialised with alignments.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
994.01	40.58	73.19	17.76

Table 4.37: *Results for EE test set alignment using models trained on 14 manually aligned calls.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
953.50	29.07	52.11	13.22

models just described. I then manually checked 14 calls for boundary placement errors and corrected any that I found. These 14 calls were then used to train new models. The results of aligning the test set using these models are given in table 4.37. The mean error has decreased from 40.58 ms to 29.07 ms and the median error has decreased from 17.76 ms to 13.22 ms. The advantage of training from aligned data over only using embedded training is evident. Also, the previous results are on models trained on 257 calls whereas these models were only trained on 14 calls. This makes the improvement even more impressive.

The next step was to try using the rest of the training data to improve the models trained on the 14 calls. Two cycles of embedded training were performed. The results are given in table 4.38. The mean has decreased from 29.07 ms to 27.73 ms and the median from 13.22 to 12.91 ms. This is a smaller improvement than those obtained previously. More cycles of embedded training could possibly have improved the results further, but I was concerned about overtraining the models.

Instead, because the frame shift of 5 ms gave better results with the TIMIT experiments, a set of models was trained for this frame shift. The models described in the previous paragraph were used to align the EE training set. The aligned data was then used to train the 5 ms frame shift models. The results are given in table 4.39. The mean and the median have decreased but the maximum has increased.

Table 4.38: *Results for EE test set alignment using models initialised using 14 manually aligned calls and trained on the entire EE training set.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
724.98	27.73	48.58	12.91

Table 4.39: *Results for EE test set aligned using models with a 5 ms frame shift trained from the EE training set aligned using 2 ms frame shift models.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
824.97	24.28	41.07	12.12

Table 4.40 gives results for the previous models after two embedded training cycles on the entire EE training set. All the measures have improved slightly.

Finally, variable minimum duration models were used. The best results obtained for TIMIT were 5 ms frame shift using variable minimum duration models. For the EE database, the minimum duration values were obtained from the 14 hand aligned calls in the training set. Variable minimum duration models were created using the same pdfs as used for the experiment of table 4.39. They were trained using 2 cycles of embedded training. The results are given in table 4.41. All the measures have improved except the median which has increased by 1.17%. Overall, there is a slight improvement, but not as significant as for the TIMIT case. One reason could be that the 14 calls are not representative enough of the data in the test set.

Table 4.40: *Results for EE test set aligned using models with a 5 ms frame shift after 2 cycles of embedded training on the EE training set.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
794.97	24.02	40.07	11.99

Table 4.41: *Results for EE test set aligned using variable minimum duration models with a 5 ms frame shift.*

Maximum Abs Error [ms]	Mean Abs Error [ms]	Std Dev of Abs Error [ms]	Median Error [ms]
714.97	23.51	38.56	12.13

Table 4.42: *Largest errors for experiment in table 4.41.*

Error [s]	label
0.714971	[int]
0.646984	v
0.537387	[spk]
0.528207	m
0.51359	@
0.504405	[sta]
0.485653	n
0.446447	[spk]
0.44475	n
0.442749	[spk]

Table 4.42 lists the largest errors and gives the segment labels. Half of the segments are noise. The first error in the list is a long (800 ms) `[int]` segment. It is followed by an `f` segment which is similar to the `[int]` segment. The second error is caused by a related problem. The `v` segment is followed by a `[sta]` segment. Again they are similar. The third error is at a `[spk]` segment followed by a `t`. Part of the duration of the `[spk]` is recognised as being part of the closure portion of the `t`.

4.6.3 Other Databases

Other AST databases have been aligned using the above method, but results cannot be reported because there are no hand aligned reference labels available. One example was the alignment of databases for the lexicon check. This check involved a transcriber reading through the pronunciation lexicon and looking for words that had suspicious pronunciations. The transcriber would then open the audio and transcription files and listen to the section of speech to determine whether the pronunciation was correctly transcribed. Some of the files were very long (e.g. the longest file in the EE database is 6 minutes). To assist the transcribers, the database was aligned and the checking program modified to open the file at the correct location.

4.7 Summary

Manual segmentation is complicated and tedious. A computer is better suited for the task. The method chosen for aligning speech files with label sequences was to use hidden Markov models and forced alignment. A program to perform embedded training was written to allow training from speech files for which boundaries between phone segments were not available. Experiments were performed on the TIMIT database to determine a good model structure and feature set for segmentation. These included checking the effect of grouping phones together, splitting data into male and female speakers, modelling duration, enforcing larger than normal minimum duration in the models and trying some special features like zero crossing density. Experiments were performed on down-sampled TIMIT files and NTIMIT to study the effects of lower bandwidth and channel noise. Finally, the EE database was aligned using the techniques learnt from the TIMIT experiments. The results improved from 67.90 ms mean error when aligning with mapped NTIMIT models to 23.51 ms. This result is also better than the 27 ms mean error reported in [2]. Also most of the boundaries fall within the 45 ms interval allowed in [20].

Chapter 5

Validation

This chapter describes methods to:

1. search for labelling errors in the label files produced by the transcribers,
2. search for labelling errors in transcription files once the alignment has been performed, and
3. measure the accuracy of the boundary placement.

5.1 Introduction

There are three types of labelling errors:

insertion A part of the given label sequence has an extra label not present in the correct label sequence.

deletion A label in the correct label sequence is missing in the given label sequence.

substitution A label in the correct label sequence has been replaced with a different label in the given label sequence.

Removing labelling errors should improve the accuracy of the boundary placement. The ease of determining a labelling error depends on the difference between the correct label sequence and the given label sequence in the region of the error. The effect of the error on automatic boundary placement should increase with the severity of the error and therefore the errors that are worst for boundary placement should be eliminated first.

5.2 Validation Methods

This section lists the checks performed to validate the databases.

5.2.1 Checks on Audio Files Only

1. Check for files with zero size. Such files are of no use in the set and should be deleted. This is done by listing the files together with their sizes after sorting the files in order of size.

2. Check for files that contain a constant signal. This is a recording error. These files must be removed from the set. Files only containing silence should be retained for possible use in training silence models. What is termed silence is actually low amplitude random noise (in the telephone line, the microphone, the recording device etc.).

3. The amplitude values of the signal should be centred about zero. Some recording devices introduce an offset which can produce erroneous results when the signal power is calculated. This problem can be corrected by highpass filtering the files.

5.2.2 Checks on Transcription Files Only

1. Use a list of allowed labels to check all the label files for incorrect labels. This check is to detect typing errors made by the transcribers.

2. For each language or accent group, use a list of phones that are expected to occur. Check through the label files for phones not in these lists. Such phones may be errors, but it is possible for phones of other languages to be used, especially in names. One can avoid making an explicit list by automatically producing a list of phones in the label files sorted according to frequency. If one works from the bottom of the list (rarest phones), the number of occurrences to check for each phone will be few and the error percentage should be high (if no similar checks have been performed yet).

3. Check for extremely short segments. These are probably errors that cannot be seen in the graphical transcription tool without zooming in several times.

4. Check whether the time indicated for each segment in a transcription file falls between a minimum and maximum predicted duration for a segment of that type. See section 5.6 for details of this check.

5. Check that the number of orthographic strings and the number of phonetic strings are equal. Because the phonetic strings are in X-SAMPA (see Appendix A), they do not need to contain spaces to be uniquely parsed. Spaces are therefore used to separate words in both the orthographic and phonetic transcriptions. This implies that the orthographic and phonetic transcriptions of an audio file must contain the same number of spaces.

6. Produce a list of the orthographic and phonetic transcriptions of assimilations so that this list can be checked for inconsistencies. If two orthographic strings are connected to indicate an assimilation, their corresponding phonetic strings must also be connected. A not uncommon error is for an orthographic string to be connected to the string on its right but for its phonetic string to be connected to the string on its left (or vice versa). This list must be checked by the transcribers, but it is less work than checking entire transcriptions and because they are concentrating on one type of error, an error is more likely to be detected. See section 5.7 for some examples.

7. Produce a pronunciation lexicon from the transcriptions and extract the rarest pronunciations for each word. These can then be checked. If a transcriber types an incorrect label, it is more likely to transform the pronunciation string for the word from a more common to a less common (or even unique) string.

5.2.3 Checks on the Combination of Audio and Transcription Files

1. Check for a difference between the length of the acoustic file and the end time of last label in the transcription file. The length of the file in seconds is

$$t = b/s/r$$

where

- t is the length of the file in seconds
- b is the length of the file in bytes
- s is the sample size in bytes per sample
- r is the sampling rate in samples per second

2. Check the power of segments marked as silence and flag any that have an average power value above some threshold value. See section 5.3.

3. Check the power in segments marked as non-silence and flag any that have an average power value below some threshold value. See section 5.4.

4. Check the number of values above a specified amplitude value in the silence segments and flag any segments in which the number of these large values is too great. This can detect short sections of non-silence in long silence sections that are not detected by the power check because the power is an average of the whole segment. Another way to do this is to calculate the power in a short window that is moved over the silence section. The method chosen is more efficient because each sample is used only once instead of every time it falls within the window (the number of times a sample that is not at the edge of the segment will be used in a calculation is equal to the length of the window in samples).

5. Check for unmarked silence sections on the edges of sections marked as speech. See section 5.5.

5.3 Silence Power Test

This test checks the power in sections marked as silence and indicates an error if the power of a segment is above a threshold. Figures 5.1 and 5.2 show errors found using this method. In figure 5.1, the boundary between the initial silence and the first utterance has been placed too far to the right, so that some of the utterance is included in the silence section. Figure 5.2 shows the end of a file. The file finishes with a long section of noise, but a silence segment has been marked at the end of this noise even though there is no change in amplitude of the signal.

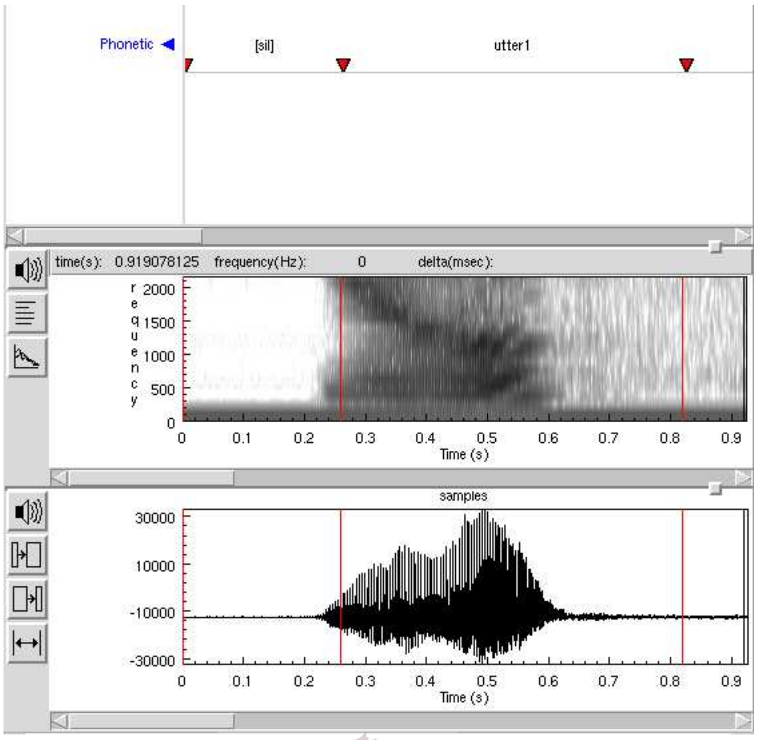

Figure 5.1: *An incorrectly labelled silence interval.*

Table 5.1 gives results for some of the AST databases. The database name is given in the column heading and the iteration number (number of times the transcribers checked the files for this type of error) is given in the first column. The checks were run until the number of flagged intervals became low enough (typically fewer than ten). Notice that the number of flagged intervals drops quite rapidly initially indicating that most of the flagged intervals are actually errors. The final entry should indicate the number of incorrectly flagged intervals. The interval count can increase if boundaries are moved instead of the interval being relabelled. This can lead to two intervals being flagged instead of one on the next iteration.

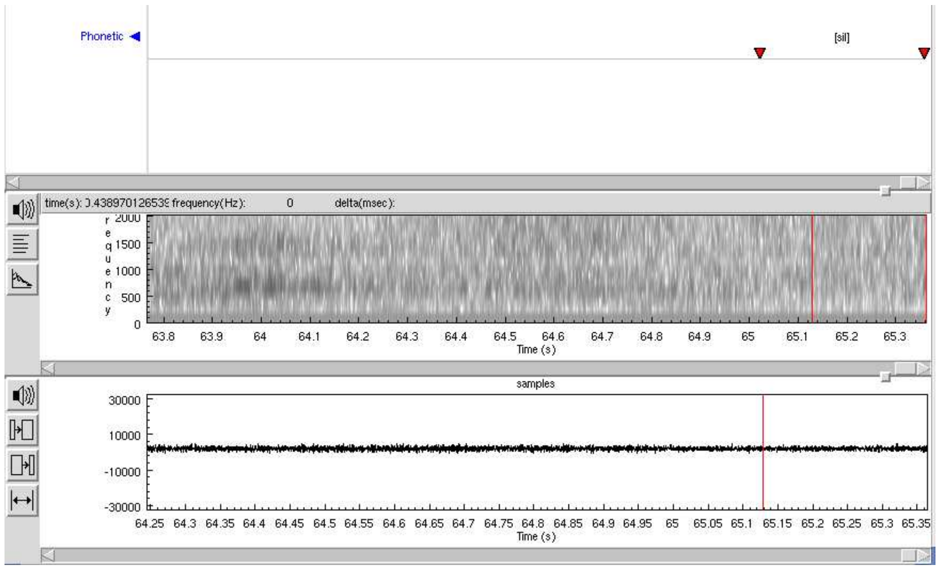

Figure 5.2: *Another incorrectly labelled silence interval.*

Table 5.1: *Number of intervals flagged as possible errors for each iteration of the silence power check. Intervals marked as silence are flagged if the power is too large.*

iteration	AE	BE	CA	CE	EE	IE
1	550	812	1178	573	269	354
2	38	941	31	481	9	6
3	3	34	3	6	4	5
4		37	2	3	1	5
5		4				

5.4 Non-Silence Power Test

This test checks the power in sections not marked as silence and indicates an error if the power of a segment is below a threshold. Figures 5.3 and 5.4 show errors found using this method. In figure 5.3, there is a silence interval on the left and a noise interval on the right, even though the amplitude of the signal is the same for the two intervals. The noise interval should have been labelled as silence. Figure 5.4 also contains two intervals. The one on the left is clearly a silence interval, but has been incorrectly labelled as noise.

Table 5.2 gives some results for this check. The databases shown all underwent four rounds of this check. As for the silence power check, the number of flagged intervals drops

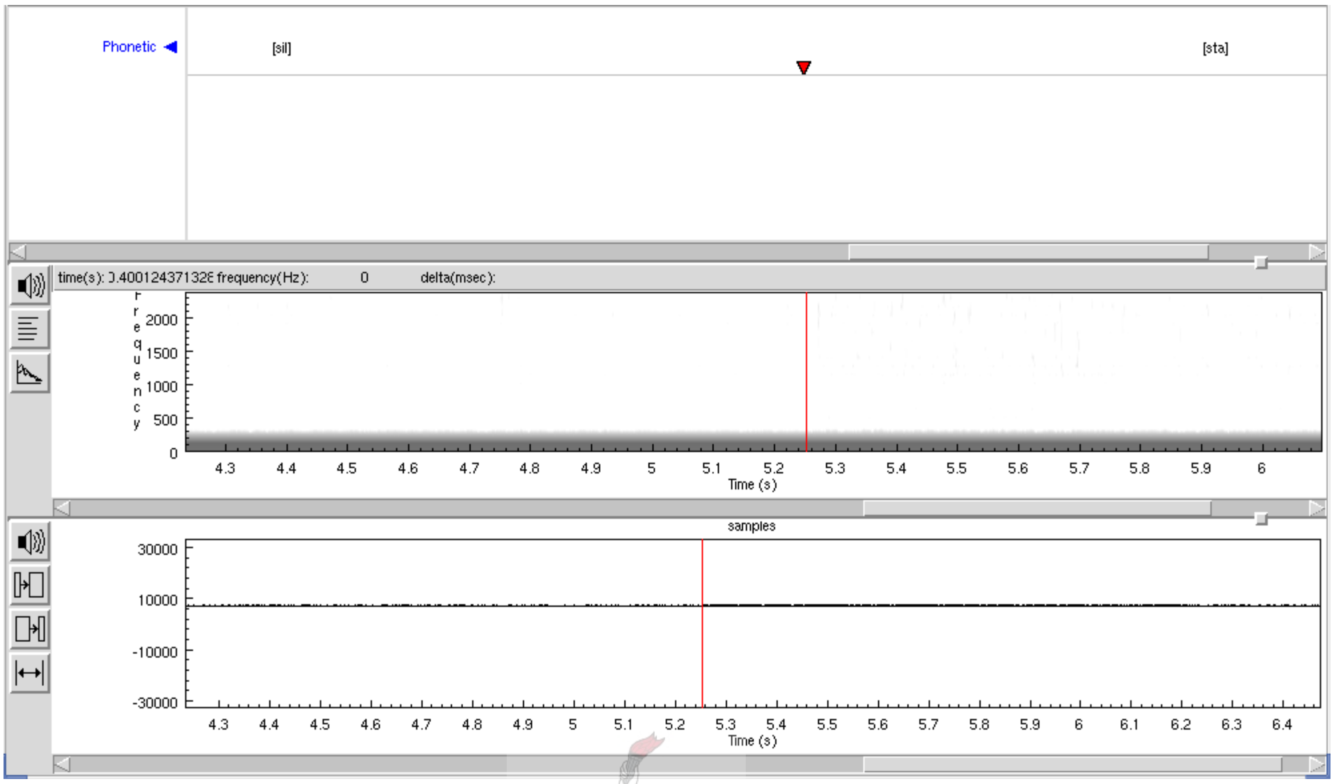

Figure 5.3: *An incorrectly labelled non-silence interval.*

quite rapidly and finally there are fewer than ten incorrectly flagged intervals remaining in the list.

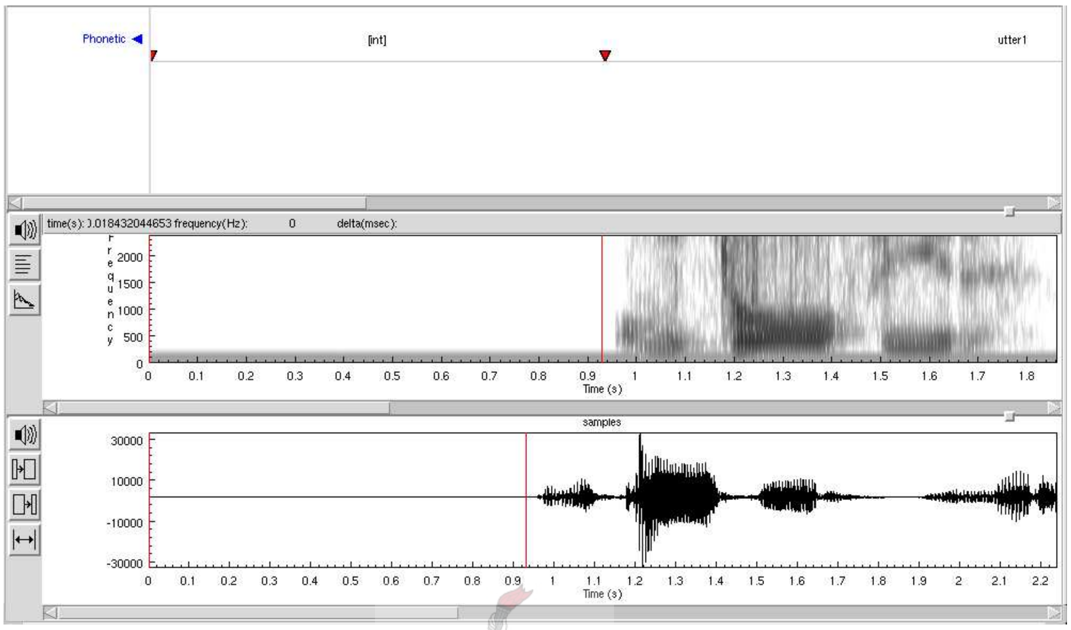

Figure 5.4: *Another incorrectly labelled non-silence interval.*

Table 5.2: *Number of intervals flagged as possible errors for each iteration of the non-silence power check. Intervals not marked as silence are flagged if the power is too low.*

iteration	AE	CA	CE	IE
1	376	263	567	925
2	96	109	320	32
3	25	16	45	5
4	9	9	8	5

5.5 Speech Segment Bounds Check

This test checks sections that have been labelled as containing speech. The program starts from the left boundary and moves right (into the speech segment) until the amplitude of the signal (the amplitude value of a few consecutive samples) rises above some threshold. If the distance to this point exceeds some allowed maximum value, an error is indicated. This distance was chosen to be 40 ms because it was believed that a transcriber can place a boundary accurately within this error margin. Figure 5.5 shows part of a file in which the boundaries have been correctly placed, whereas in figure 5.6, the left boundary of the speech interval has been placed about one second too far to the left. If the left boundary is found to be in an acceptable position, the program starts from the right boundary and moves left using the same procedure as for the left boundary.

Table 5.3 gives results for the boundary check. The initial number of boundaries flagged is very large. One contributing factor to this is that the boundary specification was made much tighter after a lot of the transcription work had been done. The tighter boundaries were necessary for the duration check (see section 5.6).

Table 5.3: *Number of boundaries flagged as possible errors for each iteration of the speech segment bounds check. A boundary is flagged if it there is too much low amplitude signal between it and the high amplitude signal in the interval.*

iteration	AA	AE	BE	CA	CE	IE
1	4925	2096	2526	2982	2012	1394
2	98	489	3199	187	1167	429
3	28	39	155	19	150	33
4		13	168			34

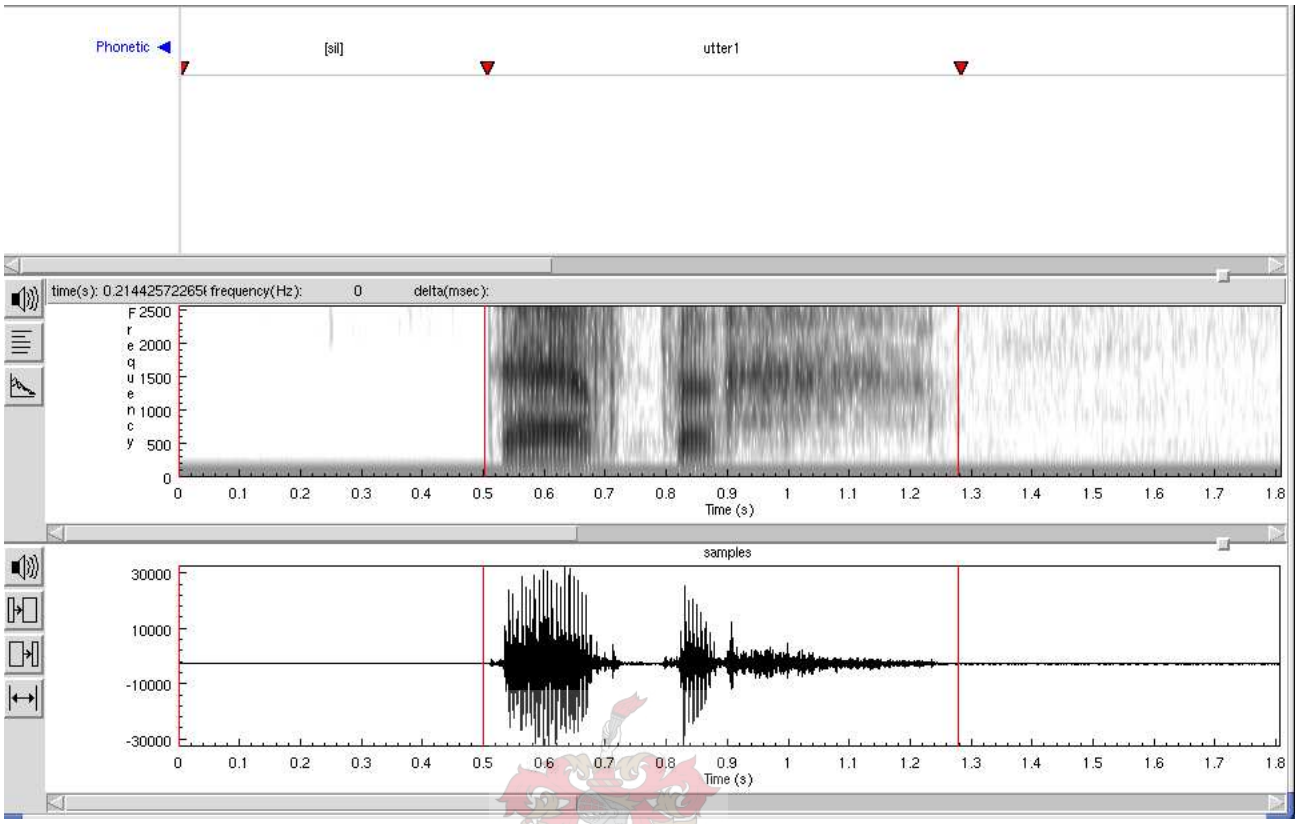

Figure 5.5: *Accurately placed boundaries around a speech interval.*

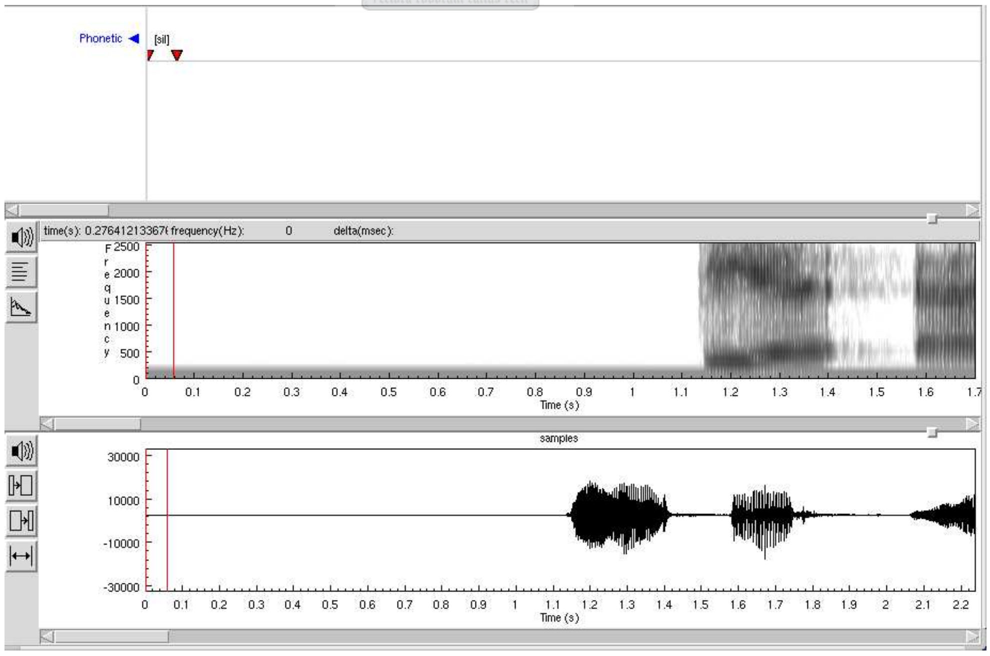

Figure 5.6: *A badly placed boundary.*

5.6 Duration Check

As described in section 3.2, each transcription file consists of a sequence of segments separated by manually placed time boundaries. A segment can contain speech, noise, silence, or any combination of these. In the phonetic transcription files, the speech segments consist of a sequence of phones.

The duration check is only done on speech segments because the durations of segments containing only noise, only silence or only noise and silence are very unpredictable. Duration models are trained for each unique phone in the label files, for each type of noise, and for silence. Only silence and noise that occur in a segment containing phones are modelled. Silence and noise between phones have more predictable durations.

Histograms are used for the duration models. The histograms are trained on phonetic transcription files in which a boundary is placed between each phone. Segmented phonetic transcriptions were not available for the AST data, so a mapping between the TIMIT phone set and the AST phone set was produced and the TIMIT database was used to train the histograms. The values in the bins of the histograms are normalized by dividing by the sum of the values in all the bins.

For each segment in a transcription file that contains at least one phone, the time of the left bound for that segment is subtracted from the time for the right bound to give the stated duration. A segment duration falling outside its permitted range indicates a probable labelling error or misplaced boundary.

One can consider the duration of a particular phone to be a random variable (the pdf of which we model with a histogram). If we assume that the phone durations are independent of each other, the histogram for the labels $l_1 \ldots l_n$ is given by $h_{sum} = h_1 * h_2 * \cdots * h_n$ [18, page 118] where h_i is the histogram corresponding to the label l_i.

The user of the error checking program enters a threshold value which is used to determine the minimum and maximum durations that will be regarded as correct and not flagged as an error. The minimum allowed duration is determined by starting from the left of the histogram and adding the heights of the bins until the threshold value is reached. The index of the bin for which this occurs gives the minimum duration. The maximum duration is determined by starting from the rightmost bin of the histogram and going left.

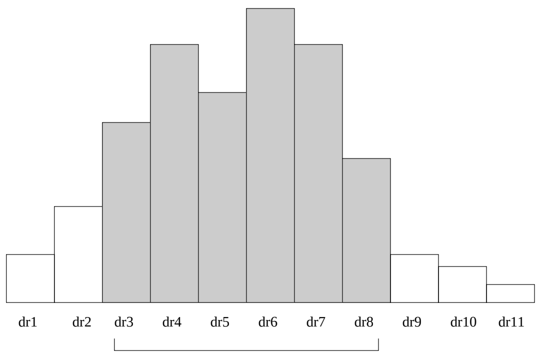

Figure 5.7: *Estimating duration using a histogram.*

Figure 5.7 gives a representation of a histogram that could be produced by the above convolution procedure. The heights of the bins sum to one. In this case the threshold results in durations from d3 to d8 being accepted (in grey) and durations from d1 to d2 and from d9 to d11 being rejected. The minimum duration for the segment is d3 and the maximum is d8. If the duration obtained from the transcription file falls outside this range, the segment is added to a list of possible errors.

When the algorithm was first developed, the transcription files contained labels that indicated optional silence. The above procedure had to be modified to handle the optional silences. When determining the minimum allowable duration, all optional silence labels were removed from the transcription and they were all converted to ordinary silence markers for determining the maximum allowable duration. Therefore, the histogram for the maximum segment duration was produced from the convolution of a sequence of histograms which included multiple copies of the histogram representing interword silence whereas the histogram for the minimum segment duration excluded these interword silence histograms in its convolution. The maximum segment duration histogram will predict a larger duration than the minimum segment duration histogram because of the extra histograms being convolved. Also the maximum duration histogram will be broader (i.e. have more bins) than the minimum duration histogram because each successive histogram in the convolution adds one fewer bins than the number of bins it has to the resulting histogram.

Table 5.4 gives some errors that were discovered in the Xhosa database using this check. The phone set used is X-SAMPA (see Appendix A).

It would have been useful to rerun the duration checks once the alignments of the AST

Table 5.4: *Some errors found using the convolution check.*

phone sequence	reported duration [s]
i n t	1.97
E m p c	1.93
O n E	1.45
i O j i l E	2.95
h a j i	1.77

databases were available. Histograms trained from phone segments in the database under consideration will be much more accurate than mapped histograms trained on TIMIT. Unfortunately, by the time the alignments were available, the checking of the databases had ended (and many of the transcribers were no longer employed by the project).

5.7 List of Assimilations

The term *assimilation* is used in this work for when the pronunciations of adjacent words are different from any of the forms that would be used if the words were spoken in isolation. The pronunciation of the group of words forming the assimilation must then be given as a unit so that the context is maintained.

Some examples are given in table 5.5. The phonetic transcriptions are in the X-SAMPA phone set. A semicolon is used to indicate an assimilation.

Table 5.5: *Examples of assimilations.*

orthographic transcription	phonetic transcription
eat;your raisins	`i:t-\SO r\e-\Iz@ns`
hundred;and;ninety	`hVndr\@nV-\IntI`
got;a	`gQ4@`

5.8 Summary

This chapter explains methods for checking for errors in speech files, transcription files and alignments. There are many automatic methods of error detection that can be implemented to improve the quality of speech databases. Using these methods can save a lot of transcriber time by directing their attention to problem areas instead of having them check all the files themselves.

Chapter 6

Conclusions

The embedded training and forced alignment algorithms were used to produce aligned files in the AST databases. Two uses for the aligned files were *zooming* the transcription tool to a specific word for the transcribers to check a pronunciation, and scoring a program that finds words in untranscribed speech files.

Duration modelling was found to improve the boundary accuracy, but not as much as minimum duration modelling customised for each phone. This produced the best result of 9.34 ms mean error. Features like zero crossing density and the ratio of low frequency energy to high frequency energy did not improve results. Both grouping TIMIT phones and separating files by gender caused the results to deteriorate.

When the EE database was aligned using models trained on NTIMIT, a mean boundary error of 67.90 ms was obtained. This was reduced to 23.51 ms by training on a portion of the EE database. This mean is well within the 45 ms error margin allowed in the literature for alignment of telephone speech. It even compares well with the 27 ms and 14 ms mean errors reported in the literature for high quality speech.

Several error checks were implemented which uncovered numerous errors or inaccuracies in the transcription files. Four transcribers spent over a year of the project just correcting errors found by these checks. The resulting improved transcriptions allowed for more accurate alignments.

6.1 Future Work

- If portions of the other AST databases are hand aligned, alignment experiments could be performed on them. This would be particularly interesting for the African languages.

- It would be interesting to experiment on aligning high quality South-African English speech to compare results with TIMIT.

Bibliography

[1] BLACK, A. W. and CAMPBELL, N., "Optimising selection of units from speech databases for concatenative synthesis." in *Proceedings of Eurospeech*, (Madrid, Spain), pp. 581–584, September 1995.

[2] COSI, P., FALAVIGNA, D., and OMOLOGO, M., "A preliminary statistical evaluation of manual and automatic segmentation discrepancies." in *Proceedings of Eurospeech*, pp. 693–696, 1991.

[3] DALSGAARD, P., ANDERSEN, O., and BARRY, W., "Multi-lingual label alignment using acoustic-phonetic features derived by neural-network technique." in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 197–200, 1991.

[4] DELLER, J. R., JR, HANSEN, J. H. L., and PROAKIS, J. G., *Discrete-Time Processing of Speech Signals*. New York: IEEE Press, 2000.

[5] DU PREEZ, J. A. and WEBER, D. M., "The integration and training of arbitrary-order HMMs." *Australian Journal on Intelligent Information Processing Systems (ICSLP98 Special Issue)*, July 1999, Vol. 5, No. 4, pp. 261–268.

[6] GOLD, B. and MORGAN, N., *Speech and Audio Signal Processing*. New York: John Wiley and Sons, Inc., 2000.

[7] HöHNE, H. D., COKER, C., LEVINSON, S. E., and RABINER, L. R., "On temporal alignment of sentences of natural and synthetic speech." *IEEE Transactions on Acoustics, Speech and Signal Processing*, August 1983, Vol. 31, pp. 807–813.

[8] JELINEK, F., *Statistical Methods for Speech Recognition.* Cambridge, Massachusetts: The MIT Press, 1999.

[9] KIPP, A., WESENICK, M.-B., and SCHIEL, F., "Automatic detection and segmentation of pronunciation variants in German speech corpora." in *Proceedings of the International Conference on Spoken Language Processing*, pp. 106–109, 1996.

[10] KVALE, K., *Segmentation and Labelling of Speech.* PhD thesis, Institutt for Teleteknikk, Trondheim, Norway, 1993.

[11] LADEFOGED, P., *Vowels and Consonants: An Introduction to the Sounds of Languages.* Blackwell Publishers, 2000.

[12] LADEFOGED, P., *A Course in Phonetics.* Fourth edition. Thomson Learning, 2001.

[13] LAMEL, L., KASSEL, R., and SENEFF, S., "Speech Database Development: Design and Analysis of the Acoustic-Phonetic Corpus." in *Proceedings of the DARPA Speech Recognition Workshop*, pp. 26–32, 1987.

[14] LEUNG, H. C. and ZUE, V. W., "A procedure for automatic alignment of phonetic transcriptions with continuous speech." in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 2.7.1–2.7.4, 1984.

[15] LJOLJE, A. and RILEY, M. D., "Automatic segmentation and labelling of speech." in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 473–476, 1991.

[16] MALFRèRE, F., DEROO, O., and DUTOIT, T., "Phonetic alignment: speech synthesis based vs. hybrid HMM/ANN." in *Proceedings of the International Conference on Spoken Language Processing*, vol. 4, pp. 1571–1574, 1998.

[17] NOLAN, F. and ESLING, J. (Eds), *International Phonetic Association Handbook: A Guide to the Use of the International Phonetic Alphabet.* Cambridge University Press, June 1999.

[18] PEEBLES, P. Z., JR, *Probability, Random Variables, and Random Signal Principles.* Third edition. Singapore: McGraw-Hill, Inc., 1993.

[19] RABINER, L. R. and JUANG, B. H., "An Introduction to Hidden Markov Models." *IEEE ASSP Magazine*, January 1986, Vol. 3, pp. 4–16.

[20] SVENDSEN, T. and SOONG, F. K., "On the automatic segmentation of speech signals." in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 77–80, 1987.

[21] TORKKOLA, K., "Automatic alignment of speech with phonetic transcriptions in real time." in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 611–614, 1988.

[22] VAN DEN HEUVEL, M., "Integrated IPA, SAMPA, X-SAMPA, SAMPROSA, ASTbet and PRAAT chart." AST project document, unpublished, June 2002.

[23] VAN HEMERT, J. P., "Automatic segmentation of speech." *IEEE Transactions on Signal Processing*, April 1991, Vol. 39, No. 4, pp. 1008–1012.

[24] WALPOLE, R. E., *Introduction to Statistics.* Third edition. New York: Macmillan Publishing Co., Inc., 1982.

[25] WELLS, J. C., "Computer-coding the IPA: a proposed extension of SAMPA." revised draft, April 1995.

[26] WELLS, J., "SAMPA computer readable phonetic alphabet." in *Handbook of Standards and Resources for Spoken Language Systems* (GIBBON, D., MOORE, R., and WINSKI, R. (Eds)), Berlin and New York: Mouton de Gruyter, 1997.

[27] WESENICK, M.-B. and KIPP, A., "Estimating the quality of phonetic transcriptions and segmentations of speech signals." in *Proceedings of the International Conference on Spoken Language Processing*, vol. 1, pp. 129–132, 1996.

[28] YOUNG, S., KERSHAW, D., ODELL, J., OLLASON, D., VALTCHEV, V., and WOODLAND, P., *The HTK Book.* Entropic, Ltd., Cambridge, U.K., Htk version 3.0 edition edition, 2000.

Appendix A

Phone Sets

This chapter list phonetic transcription symbols in three phone sets: IPA [17], X-SAMPA [25] and ASTbet [22]. X-SAMPA provides an encoding for each IPA symbol using the graphical subset of the 7-bit ASCII character set (i.e. ASCII characters 33–127). It is based on SAMPA [26] which was limited to encoding western European languages. X-SAMPA transcriptions can be stored in a text file and entered using a standard (U.S.) keyboard. ASTbet was developed as part of the AST project to provide an encoding for all the IPA symbols that are used to transcribe South-African languages. The encodings comprise only lowercase letters, digits and the underscore character (and always begin with a letter) so that they can be used as filenames on older operating systems and for programs such as HTK [28].

Table A.1: *Stops.*

IPA	X-SAMPA	ASTbet	description
p	p	p	voiceless bilabial plosive
b	b	b	voiced bilabial plosive
t	t	t	voiceless alveolar plosive
d	d	d	voiced alveolar plosive
ʈ	t\`	t_rt	voiceless retroflex plosive
ɖ	d\`	d_rt	voiced retroflex plosive
c	c	c	voiceless palatal plosive
ɟ	J\	jb	voiced palatal plosive
k	k	k	voiceless velar plosive
g	g	g	voiced velar plosive
q	q	q	voiceless uvular plosive
ɢ	G\	gc	voiced uvular plosive
ʔ	?	gl	glottal plosive

Table A.2: *Nasals, trills and taps.*

IPA	X-SAMPA	ASTbet	description
m	m	m	bilabial nasal
ɱ	F	mj	labiodental nasal
n	n	n	alveolar nasal
ɳ	N\`	nj_rt	retroflex nasal
ɲ	J	nlt	palatal nasal
ŋ	N	nj	velar nasal
ɴ	N\	nc	uvular nasal
ʙ	B\	bc	bilabial trill
r	r	r	alveolar trill
ʀ	R\	rc	uvular trill
ɾ	4	fh	alveolar tap or flap
ɽ	R\`	ri_rt	retroflex tap or flap

Table A.3: *Fricatives.*

IPA	X-SAMPA	ASTbet	description
ɸ	P\	fi	voiceless bilabial fricative
β	B	be	voiced bilabial fricative
f	f	f	voiceless labiodental fricative
v	v	v	voiced labiodental fricative
θ	T	th	voiceless dental fricative
ð	D	dh	voiced dental fricative
s	s	s	voiceless alveolar fricative
z	z	z	voiced alveolar fricative
ʃ	S	sh	voiceless post-alveolar fricative
ʒ	Z	zh	voiced post-alveolar fricative
ʂ	s‘	s_rt	voiceless retroflex fricative
ʐ	z‘	z_rt	voiced retroflex fricative
ç	C	clt	voiceless palatal fricative
ʝ	j\	jc	voiced palatal fricative
x	x	x	voiceless velar fricative
ɣ	G	vr	voiced velar fricative
χ	X	ci	voiceless uvular fricative
ʁ	R	ri	voiced uvular fricative
ħ	X\		voiceless pharyngeal fricative
ʕ	?\		voiced pharyngeal fricative
h	h	h	voiceless glottal fricative
ɦ	h\	hht	voiced glottal fricative
ɬ	K	lb	voiceless alveolar lateral fricative
ɮ	K\	lz	voiced alveolar lateral fricative

Table A.4: *Approximants.*

IPA	X-SAMPA	ASTbet	description
ʋ	v\	up	voiced labiodental approximant
ɹ	r\	rt	voiced alveolar approximant
ɻ	r\`	rt_rt	voiced retroflex approximant
j	j	j	voiced palatal approximant
ɰ	M\	ml	voiced velar approximant
l	l	l	voiced alveolar lateral approximant
ɭ	l`	l_rt	retroflex lateral approximant
ʎ	L	yt	palatal lateral approximant
ʟ	L\	lc	velar lateral approximant

Table A.5: *Non-pulmonic consonants.*

IPA	X-SAMPA	ASTbet	description
ʘ	O\	cl1	bilabial click
|	|\	cl2	dental click
!	!\	cl3	(post)alveolar click
ǂ	=\	cl4	palatoalveolar click
‖	|\\|\	cl5	alveolar lateral click
ɓ	b_<	b_imp	voiced bilabial implosive
ɗ	d_<	d_imp	voiced dental or alveolar implosive
ʄ	J_<	jb_imp	voiced palatal implosive
ɠ	g_<	g_imp	voiced velar implosive
ʛ	G_<	gc_imp	voiced uvular implosive

Table A.6: *Other consonants.*

IPA	X-SAMPA	ASTbet	description
ʍ	W	wt	voiceless labio-velar fricative
w	w	w	voiced labio-velar approximant
ɥ	H	ht	voiced labio-palatal approximant
ʜ	H\	hc	voiceless epiglottal fricative
ʢ	<\	gtb	voiced epiglottal fricative
ʡ	>\	gb	epiglottal plosive
ɕ	s\	cc	voiceless alveolo-palatal fricative
ʑ	z\	zc	voiced alveolo-palatal fricative
ɺ	l\	rlt	alveolar lateral flap
ɧ	x\	hhj	simultaneous ʃ and x

Table A.7: *Vowels.*

IPA	X-SAMPA	ASTbet	description
i	i	i	close front unrounded vowel
y	y	y	close front rounded vowel
ɨ	1	ib	close central unrounded vowel
ʉ	}	ub	close central rounded vowel
ɯ	M	mt	close back unrounded vowel
u	u	u	close back rounded vowel
ɪ	I	ic	close front lax unrounded vowel
ʏ	Y	yc	close front lax rounded vowel
ʊ	U	hs	close back lax rounded vowel
e	e	e	close-mid front unrounded vowel
ø	2	phi	close-mid front rounded vowel
ɘ	@\		close-mid central unrounded vowel
ɵ	8		close-mid central rounded vowel
ɤ	7		close-mid back unrounded vowel
o	o	o	close-mid back rounded vowel
ə	@	sw	schwa
ɛ	E	ep	open-mid front unrounded vowel
œ	9	oe	open-mid front rounded vowel
ɜ	3	epr	open-mid central unrounded vowel
ɞ	3\		open-mid central rounded vowel
ʌ	V	vt	open-mid back unrounded vowel
ɔ	O	ct	open-mid back rounded vowel
æ	{	ae	near-open front unrounded vowel
ɐ	6	at	near-open central vowel
a	a	a	open front unrounded vowel
ɶ	&		open front rounded vowel
ɑ	A	as	open back unrounded vowel
ɒ	Q	ab	open back rounded vowel

Table A.8: *Diacritics part 1.*

IPA	EXAMPLE	X-SAMPA	ASTbet	description
̥	n̥	_0	_vcls	voiceless
̬	s̬	_v	_vcd	voiced
^{h}	t^{h}	_h	_asp	aspirated
̹	ɔ̹	_O	_mrnd	more rounded
̜	ɔ̜	_c	_lrnd	less rounded
̟	u̟	_+	_adv	advanced
̠	e̠	_-	_rtr	retracted
̈	ë	_”	_cnt	centralized
̽	e̽	_x	_mcen	mid-centralized
̩	n̩	=	_syl	syllabic
̯	e̯	_^	_nsyl	non-syllabic
˞	e˞	‘	_rt	rhoticity
̤	b̤	_t	_brvd	breathy voiced
̰	b̰	_k	_ckvd	creaky voiced
̼	t̼	_N	_lglb	linguolabial
^{w}	t^{w}	_w	_lab	labialized
^{j}	t^{j}	’	_plt	palatalized
^{ɣ}	$t^{ɣ}$	_G	_vel	velarized
^{ʕ}	$t^{ʕ}$	_?\	_phr	pharyngealized
̴	ɫ	_e	_vlph	velarized or pharyngealized
̝	e̝	_r	_rzd	raised
̞	e̞	_o	_lwrd	lowered
̘	e̘	_A	_atg	advanced tongue root
̙	e̙	_q	_rtg	retracted tongue root

Table A.9: *Diacritics part 2.*

IPA	EXAMPLE	X-SAMPA	ASTbet	description
◌̪	t̪	_d	_dnt	dental
◌̺	t̺	_a	_apc	apical
◌̻	t̻	_m	_lam	laminal
◌̃	ẽ	~	_nas	nasalized
ⁿ	dⁿ	_n	_nsrl	nasal release
ˡ	dˡ	_l	_ltrl	lateral release
◌̚	d̚	_}	_norl	no audible release
ː	eː	:	_long	long
◌͡◌	d͡ʒ	-\	_lnk	for affricates and diphthongs

Appendix B

TIMIT Information

This appendix gives some useful information about the TIMIT database. Tables B.1 to B.7 describe the symbols used in the phonetic transcriptions. The tables for the consonants give descriptions for the symbols rather than example words because some phones can be used to transcribe the same word depending on the particular pronunciation e.g. `t` and the flap `dx` or the sequence `sw n` and `en`. The tables for the vowels use example words because precise description of vowels can get very complicated. The remaining tables give duration statistics for each phone symbol; first for the training set and then for the test set. The statistic in the final column (the range) is the difference between the minimum and maximum durations for the particular phone.

Table B.1: *TIMIT Plosives.*

phone	description
p	burst of voiceless bilabial plosive
b	burst of voiced bilabial plosive
t	burst of voiceless alveolar plosive
d	burst of voiced alveolar plosive
k	burst of voiceless velar plosive
g	burst of voiced velar plosive
pcl	closure of voiceless bilabial plosive
bcl	closure of voiced bilabial plosive
tcl	closure of voiceless alveolar plosive
dcl	closure of voiced alveolar plosive
kcl	closure of voiceless velar plosive
gcl	closure of voiced velar plosive
q	glottal plosive

Table B.2: *TIMIT Nasals and Flaps.*

phone	description
m	bilabial nasal
em	syllabic bilabial nasal
n	alveolar nasal
en	syllabic alveolar nasal
ng	velar nasal
eng	syllabic velar nasal
dx	alveolar flap
nx	nasal flap

Table B.3: *TIMIT Fricatives.*

phone	description
f	voiceless labiodental fricative
v	voiced labiodental fricative
th	voiceless dental fricative
dh	voiced dental fricative
s	voiceless alveolar fricative
z	voiced alveolar fricative
sh	voiceless post-alveolar fricative
zh	voiced post-alveolar fricative
hh	voiceless glottal fricative
hv	voiced glottal fricative

Table B.4: *TIMIT Affricates.*

phone	description
ch	voiceless post-alveolar affricate
jh	voiced post-alveolar affricate

Table B.5: *TIMIT Approximants.*

phone	description
r	voiced alveolar approximant
y	voiced palatal approximant
l	voiced alveolar lateral approximant
el	syllabic voiced alveolar lateral approximant
w	voiced labio-velar approximant

Table B.6: *TIMIT Monophthongs.*

phone	example
iy	b**ee**t
ih	b**i**t
eh	b**e**t
ae	b**a**t
aa	b**o**mb
ah	b**u**t
ao	s**aw**
uh	b**oo**k
uw	b**oo**t
ux	t**oo**t
er	b**ir**d
ax	**a**bout
ix	deb**i**t
axr	butt**er**
ax-h	s**u**spect

Table B.7: *TIMIT Diphthongs.*

phone	example
ey	b**ai**t
aw	b**ou**t
ay	b**i**te
oy	b**oy**
ow	b**oa**t

Table B.8: *Duration statistics for phones in the TIMIT training set. Part 1.*

phone	number of occurrences	median duration [ms]	mean duration [ms]	standard deviation [ms]	minimum duration [ms]	maximum duration [ms]	range [ms]
aa	3064	117.50	122.39	38.30	23.81	366.25	342.44
ae	3997	145.00	148.21	44.18	35.00	425.00	390.00
ah	2306	83.44	88.88	35.22	18.75	340.13	321.38
ao	2940	118.44	123.87	43.99	24.44	396.63	372.19
aw	729	154.00	160.76	51.15	62.69	415.00	352.31
ax	3610	45.56	48.36	17.69	4.62	151.94	147.31
ax-h	375	30.00	33.04	17.16	10.44	174.94	164.50
axr	3407	78.06	81.44	30.23	19.06	259.81	240.75
ay	2390	133.28	143.99	53.96	38.50	424.38	385.88
b	2181	15.88	17.45	7.10	2.94	98.62	95.69
bcl	1909	60.69	62.75	24.31	8.81	229.19	220.38
ch	822	83.25	86.31	27.51	29.94	345.63	315.69
d	3548	20.00	23.01	12.48	4.25	120.00	115.75
dcl	4942	45.00	51.02	25.79	8.50	206.25	197.75
dh	2826	33.19	37.64	19.51	4.50	123.25	118.75
dx	2709	27.50	28.70	7.77	9.50	72.88	63.38
eh	3853	85.62	91.20	33.39	21.44	281.13	259.69
el	951	85.00	90.36	31.50	26.31	223.69	197.38
em	124	75.44	81.32	34.42	27.63	260.63	233.00
en	723	75.44	79.12	30.06	12.75	195.50	182.75

Table B.9: *Duration statistics for phones in the TIMIT training set. Part 2.*

phone	number of occurrences	median duration [ms]	mean duration [ms]	standard deviation [ms]	minimum duration [ms]	maximum duration [ms]	range [ms]
eng	38	79.25	82.36	23.63	43.38	128.69	85.31
epi	1464	39.59	43.15	22.77	3.19	290.00	286.81
er	2046	112.31	117.30	38.69	30.00	338.81	308.81
ey	2282	120.13	126.80	40.11	31.69	322.56	290.88
f	2216	102.50	103.01	34.75	13.31	305.00	291.69
g	2017	28.06	29.82	12.78	7.50	106.06	98.56
gcl	2223	45.63	49.68	22.31	9.81	193.19	183.38
h#	9240	142.50	194.14	167.07	2.00	2996.87	2994.87
hh	957	59.81	64.98	30.82	13.75	250.00	236.25
hv	1154	66.91	68.53	22.82	19.62	210.31	190.69
ih	5051	73.81	78.73	28.63	17.50	399.13	381.63
ix	8642	48.47	51.53	19.72	12.00	199.31	187.31
iy	6953	83.50	90.42	35.71	23.25	428.19	404.94
jh	1209	52.62	58.30	25.49	14.87	192.06	177.19
k	4874	49.59	51.36	25.55	6.13	193.81	187.69
kcl	5859	55.00	59.78	25.94	8.88	243.88	235.00
l	5801	57.44	60.21	22.35	12.56	229.81	217.25
m	3903	60.00	61.17	24.85	8.12	229.75	221.63
n	7068	50.00	54.43	22.61	7.88	212.31	204.44
ng	1330	59.13	61.68	23.40	13.50	240.25	226.75

Table B.10: *Duration statistics for phones in the TIMIT training set. Part 3.*

phone	number of occurrences	median duration [ms]	mean duration [ms]	standard deviation [ms]	minimum duration [ms]	maximum duration [ms]	range [ms]
nx	971	29.88	28.81	7.10	7.63	67.88	60.25
ow	2136	120.13	125.82	40.63	34.94	438.63	403.69
oy	684	157.06	161.05	38.02	62.63	309.50	246.88
p	2588	41.25	44.27	21.65	6.25	150.06	143.812
pau	952	150.09	184.03	123.15	22.56	1225.19	1202.62
pcl	2644	67.50	69.62	24.83	12.50	349.38	336.88
q	3590	60.50	64.90	31.07	8.37	286.81	278.44
r	6539	56.69	60.60	24.66	2.25	177.50	175.25
s	7475	108.88	112.94	38.00	23.38	381.25	357.88
sh	2238	112.44	115.40	29.86	24.06	299.75	275.69
t	4364	44.66	48.08	23.99	5.56	145.00	139.44
tcl	6644	48.13	54.59	30.54	5.19	336.25	331.06
th	751	90.00	91.55	37.83	15.25	310.00	294.75
uh	535	70.00	76.23	30.76	18.25	253.69	235.44
uw	555	98.50	106.61	46.22	24.38	334.19	309.81
ux	1908	98.72	107.69	52.80	19.00	373.38	354.38
v	1994	57.47	60.33	21.83	12.00	190.00	178.00
w	3140	60.81	66.61	32.35	9.50	216.38	206.88
y	1715	57.44	66.06	34.87	11.44	221.75	210.31
z	3773	77.44	83.56	30.92	18.19	276.44	258.25
zh	151	79.38	81.25	26.02	22.19	170.06	147.88

Table B.11: *Duration statistics for phones in the TIMIT test set. Part 1.*

phone	number of occurrences	median duration [ms]	mean duration [ms]	standard deviation [ms]	minimum duration [ms]	maximum duration [ms]	range [ms]
aa	1133	116.00	124.77	47.60	20.00	483.44	463.44
ae	1407	146.25	149.64	46.18	57.75	354.50	296.75
ah	879	85.00	90.22	34.77	30.69	307.13	276.44
ao	1156	116.97	123.05	43.97	32.50	318.87	286.37
aw	216	168.13	173.26	51.92	67.31	378.56	311.25
ax	1346	46.25	48.95	17.48	15.19	173.62	158.44
ax-h	118	32.78	35.56	17.20	12.44	129.94	117.50
axr	1383	78.75	82.99	33.13	22.75	234.50	211.75
ay	852	140.00	148.01	53.01	50.00	388.63	338.63
b	886	15.69	17.63	7.31	3.94	62.69	58.75
bcl	776	64.47	66.04	26.36	13.06	194.62	181.56
ch	259	83.75	86.43	27.12	20.00	206.81	186.81
d	1245	20.12	23.14	11.72	2.37	113.94	111.56
dcl	1643	45.62	50.78	25.49	7.94	197.88	189.94
dh	1053	34.63	38.58	19.50	8.13	128.69	120.56
dx	940	28.47	28.83	7.68	10.44	65.44	55.00
eh	1440	84.56	88.88	29.97	27.06	260.00	232.94
el	343	88.81	92.88	32.11	26.00	238.12	212.12
em	47	74.06	73.72	18.28	30.87	109.69	78.81
en	251	75.81	78.82	29.64	22.50	181.19	158.69

Table B.12: *Duration statistics for phones in the TIMIT test set. Part 2.*

phone	number of occurrences	median duration [ms]	mean duration [ms]	standard deviation [ms]	minimum duration [ms]	maximum duration [ms]	range [ms]
eng	5	59.19	69.69	21.43	44.56	105.06	60.50
epi	536	41.22	43.95	19.00	10.06	222.31	212.25
er	800	115.00	120.93	39.76	35.00	366.37	331.37
ey	806	122.97	132.47	48.27	42.31	390.62	348.31
f	912	102.19	103.09	33.39	16.25	300.00	283.75
g	755	29.38	31.76	14.05	7.50	103.81	96.31
gcl	808	46.34	51.25	25.36	10.00	344.38	334.38
h#	3360	140.62	188.17	166.08	17.50	4642.81	4625.31
hh	356	64.22	67.11	28.58	17.25	177.50	160.25
hv	369	67.31	68.56	23.86	16.75	152.44	135.69
ih	1709	74.38	78.59	26.97	20.00	235.50	215.50
ix	2945	49.63	52.25	20.22	11.62	190.00	178.37
iy	2710	82.12	88.38	32.72	25.00	310.00	285.00
jh	372	53.13	59.97	27.43	19.00	231.00	212.00
k	1614	50.00	51.30	24.92	9.38	126.75	117.37
kcl	1964	55.37	60.20	24.44	12.00	189.37	177.37
l	2356	57.44	61.38	23.19	11.88	213.44	201.56
m	1526	60.06	61.26	24.43	4.31	198.06	193.75
n	2501	50.06	54.01	21.90	7.81	190.63	182.81
ng	414	61.75	65.72	24.59	15.31	215.00	199.69

Table B.13: *Duration statistics for phones in the TIMIT test set. Part 3.*

phone	number of occurrences	median duration [ms]	mean duration [ms]	standard deviation [ms]	minimum duration [ms]	maximum duration [ms]	range [ms]
nx	360	28.69	28.77	6.59	13.06	55.75	42.69
ow	777	122.50	128.97	40.77	49.56	389.06	339.50
oy	263	163.75	167.74	44.11	67.87	318.50	250.63
p	957	40.81	44.10	21.83	8.12	110.00	101.88
pau	391	156.00	193.13	133.17	21.69	904.38	882.69
pcl	965	66.25	69.61	24.17	17.12	210.00	192.88
q	1244	63.78	66.44	32.69	6.81	305.44	298.63
r	2525	58.38	62.11	25.38	7.25	180.19	172.94
s	2639	110.31	113.99	37.62	32.50	330.00	297.50
sh	796	115.00	118.03	29.05	49.06	257.50	208.44
t	1535	44.38	48.12	23.88	4.62	152.50	147.88
tcl	2334	45.72	53.71	29.76	8.81	265.12	256.31
th	267	82.38	86.74	35.30	12.56	215.31	202.75
uh	221	70.00	77.06	28.39	28.87	176.00	147.13
uw	170	100.91	111.57	51.77	32.50	314.63	282.12
ux	580	96.47	104.88	50.91	20.00	317.75	297.75
v	710	57.44	59.42	21.18	12.06	157.81	145.75
w	1239	64.13	69.40	31.11	12.50	201.06	188.56
y	634	60.03	68.52	36.10	16.63	231.31	214.69
z	1273	78.12	85.72	32.23	28.13	265.00	236.87
zh	74	79.91	86.71	36.06	41.87	263.63	221.75