Jarle Bauck Hamar

Using Sub-Phonemic Units for HMM Based Phone Recognition

Thesis for the degree of Philosophiae Doctor

Trondheim, June 2013

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Electronics and Telecommunications

NTNU – Trondheim
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Electronics and Telecommunications

ISBN 978-82-471-4483-1 (printed ver.)
ISBN 978-82-471-4484-8 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2013:185

Printed by NTNU-trykk

Abstract

A common way to construct a large vocabulary continuous speech recogniser LVCSR is to use 3 state HMMs to model phonemic units. In this dissertation the focus is to improve this standard phone model. To this end three alternative phone recognition systems will be proposed. Central in the first two systems is a set of Acoustic SubWord Units (ASWUs), which are used in order to train phone models with an extended state topology. This extended topology contains several parallel paths and allows the model to vary the amount of states that are employed for each realisation of the phones.

In the first system this topology is fixed with four parallel paths which contains one, two, three or four states. A novel training algorithm is developed in order to train each of the states properly. In the second system the number of paths and the number of states in each of the states are derived in a data driven manner using an algorithm for pronunciation variation modelling (PVM). This algorithm is applied to the set of ASWUs in order to find variations for each phones, variations which are used to decide the topologies.

The final system is a hybrid system that employs non-negative matrix factorisation (NMF), an algorithm capable of extracting latent units in a data driven manner to model the acoustic observations. This hybrid was

proposed before in the literature for modelling audio mixtures. In this dissertation modifications to this original hybrid, the non-negative HMM (N-HMM), are suggested for it to be used on the speech recognition task. The main contribution is to introduce dependency on state duration for the output probability distribution functions. This modified structure is referred to as the non-negative durational HMM (NdHMM).

Preface

This dissertation is submitted in partial fulfilment of the requirements for the degree of philosophiae doctor at the Norwegian University of Science and Technology (NTNU).

The work, including the compulsory courses corresponding to full-time studies of one semester and one year of teaching assistant duties, has taken place in the period of August 2007 to February 2013. In the period from August 2007 to June 2009, the work was performed part time while the final year of a master degree was completed. The research was conducted at the Department of Electronics and Telecommunications, NTNU.

The work was a part of the SIRKUS project which was funded by the Norwegian Research Council.

Professor Torbjørn Svendsen at the Department of Telecommunications, NTNU, has been the supervisor of the work.

Acknowledgments

First of all, I would like to express my sincere gratitude to my supervisor Professor Torbjørn Svendsen, first for giving me the chance of pursuing a PhD and then for his invaluable insights and support throughout this work. Moreover, I would like to thank Professor Thippur Sreenivas from the Indian Institute of Science in Bangalore for his collaboration on some of the work described in this dissertation. His vast knowledge has been important for the progress of the work. In addition a special thanks to Dr Rama Sanand Doddipatla for the close collaboration on the work with the NdHMM and for reviewing this dissertation.

I am also grateful for all my colleagues at the Speech group at NTNU. They have all created a good work environment, and I have enjoyed the discussions during meetings, lunch and other occasions. Moreover I would like to thank all the colleagues at the Signal Processing group at NTNU, specially Kirsten Marie Ekseth who has always been accommodating when I needed help or practical information.

Further, I would like to thank my family and friends for supporting me through my education and life. My parents Bjørn Reidar Hamar and Grete Bauck Hamar has provided a great childhood and fantastic support all my life. A special thanks also to Knut and Marit Reksten for their enthusiasm. Finally but certainly not least, a special thanks to my dear fiancée Anita Reksten for enduring with me through this process, for her understanding and for her encouragement.

Trondheim, December 2012
Jarle Bauck Hamar

Contents

List of Abbreviations

ASR	Automatic Speech Recognition.
ASWU	Acoustic Subword Unit.
CD-HMM	Context Dependent HMM.
CI-HMM	Context Independent HMM.
DCT	Discrete Cosine Transform.
DP	Dynamic Programming.
HMM	Hidden Markov Model.
LVCSR	Large Vocabulary Continuous Speech Recognition.
N-HMM	Non-negative Hidden Markov Model.
NdHMM	Non-negative durational Hidden Markov Model.
NMF	Non-negative Matrix Factorisation.
PVM	Pronunciation Variation Modelling.

Notation and Symbols

$p(\cdot)$	Probability distribution function (pdf).
$P(\cdot)$	Probability mass function (pmf).
μ	Mean.
σ^2	Variance.
$E[\cdot]$	Expectation.
$\mathcal{A}$	Set.
$\lvert\mathcal{A}\rvert$	Cardinality of set.
$\mathcal{A} \bigcap \mathcal{B}$	Intersection of two sets.
$\underline{x}$	Vector.
$\bar{\mathbf{x}}$	Time sequence of vector or scalar.
$\bar{\mathbf{x}}_a^b$	Time sequence of vector or scalar from $t = a$ to $t = b$.

Chapter 1

Introduction

Automatic Speech Recognition (ASR) has been an area of research for some decades and the performance of the systems has improved considerably. The technology is being more and more introduced into the daily life of the humans, for instance in cell phones, customer service systems, TV sets etc. Unfortunately, as many have discovered, the performance is still significantly below the human auditory system. Humans have an impressive ability to recognise speech even in very noisy conditions. Human word recognition itself is far from perfect, but combined with extensive knowledge about language redundancy, context and history, the level of performance has proven difficult to match by a computer. Today the state of the art ASR systems still perform one order of magnitude below humans, and therefore more research is a necessity.

One of the most important factors for the advances that have been made in the field of speech recognition, is the Hidden Markov Model (HMM). It has shown to be a strong and efficient model with low complexity, and it is a key part of many state of the art ASR system. Hence, improvements to the HMM framework could have great impact on ASR systems. In the

search for improved speech recognition performance, the objective of this dissertation is to investigate the HMM and see if enhancements can be achieved by introducing some modification to the HMM.

1.1 HMM Based Speech Recognition

The design of an ASR system is dependent on the task of the recogniser. A system that is to recognise and discriminate between a few keywords only is significantly simpler than a system for transcribing spontaneous speech. One of the first decisions to make is which basic units the recogniser should employ. A recogniser can be designed to recognise words directly, or smaller units, like syllables or phones. This choice strongly depends on the task the recogniser should solve. In a keyword ASR system, using words as the smallest unit may be the best choice. The number of words the system encounters is restricted, which makes it feasible to train models for all the words. For a general purpose Large Vocabulary Continuous Speech Recognition (LVCSR), however, the situation is different: using words as units would result in a high number of models to be trained, and in any given training database, there would be a significant number of words with a small number of occurrences, making it difficult to train the corresponding word models. However, the major issue would be how to handle words that do not occur in the training database at all. This is a problem that must be handled, because it is impossible to construct a database with all the existing words. Even if such a database was created, the set of words in a language is not constant. New words appear continuously which guarantees that an LVCSR system will be exposed to words which were not encountered during training. All words (both existing and future) can be described using an appropriate - finite and constant - set of subword units, which is much easier to train once and for all. Unfortunately, smaller units are more affected by the context.

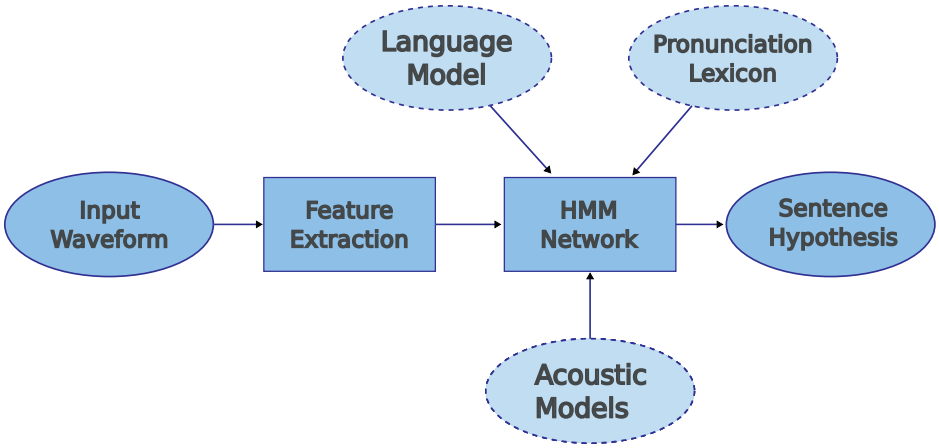


Figure 1.1: Example of an HMM based LVCSR system.

A phone can be realised quite differently in various contexts. For instance the "t" in "data" is placed in a voiced context and is often pronounced as a "d". Because of this, it is common to use context dependent (CD) phone models, as for instance the *triphone* models, which models a phone in the immediate left and right phonemic context.

An example of a conventional HMM based LVCSR system is shown in Figure 1.1. The input waveform is first presented to a feature extractor which produces feature vectors at fixed time intervals. Each feature vector contains relevant information about the input signal that can be used by the recognition unit to make a hypothesised text representation. The sequence of feature vectors is then sent to the recogniser which consists of a large network of HMMs. This large HMM network is made up of several small HMMs that models the phones (or the unit of choice). The pronunciation dictionary dictates how these small HMMs are combined into a model for all the words in the library, before the language model further combines the words into sentences.

The main focus of this dissertation will be to improve the phone modelling capability of the HMM. Improving the phone recognition capability of the ASR system, should lead to better conditions for also improving the word recognition. Although context dependent (CD) models performs better CI models are simpler which makes it more convenient to explore novel systems. Hence, the systems presented in this dissertation are all context independent.

1.2 This Dissertation

In this dissertation three HMM based phone recognition systems will be proposed. Two of the systems will utilise a set of Acoustic Sub-Word Units (ASWUs) in order to compose an extended phone model topology, designed in an effort to increase the modelling of the variation of phone pronunciation. In the final system the sub-phonetic units were extracted by the use of non-negative matrix factorisation (NMF) which was combined with the HMM. Thereby allowing the capability of the NMF for extracting latent units to be used with the modelling abilities of the HMM. This has been done before and resulted in the non-negative HMM (N-HMM). However, in this dissertation significant modifications are proposed in order to get a model that can be used for speech recognition.

1.2.1 Contributions

The contributions of this dissertation are:

- A definition of a set of Acoustic Sub-Word Units (ASWUs) based on clustering of acoustic segments (Chapter 3)
- A proposal of how to use the ASWUs to train a phone HMM with

parallel paths for realisations with different number of acoustic events (Chapter 3)

- A suggestion of how to use the ASWUs with a Pronunciation Variation Modelling (PVM) approach to compose the phone HMM topology in a data driven manner (Chapter 4).
- A set of modifications to the non-negative HMM (N-HMM) in order for it to generalise to unseen data, and thereby be a viable option for ASR (Chapter 5).

1.2.2 Outline

The dissertation will provide a description of three different HMM based phone recognition systems. However, first an introduction to the HMM in the context of ASR will be given in Chapter 2. In Chapter 3 a set of Acoustic Sub-Word Units (ASWUs) will be defined. These are given a phonetic interpretation before they are used for training a phone HMM with several parallel paths, each with a number of states ranging from 1 to 4. In Chapter 4, a system with a similar extended topology is proposed. A Pronunciation Variation Modelling (PVM) approach is employed to find the optimal phone model topology using the inventory of ASWUs defined in Chapter 3. In Chapter 5 the N-HMM is explored and modifications to make the system feasible for ASR are proposed. Finally, Chapter 6 provides a summary along with some concluding remarks.

Chapter 2

HMM Based Automatic Speech Recognition

The Hidden Markov Model (HMM) is a simple, yet powerful model widely used in the ASR community as a statistical model of speech generation. The model has a low complexity and, as will be described, the parameters can be effectively estimated with the Baum-Welch algorithm. Before the HMM is presented in Section 2.2, a brief overview of the input to the model, the feature vectors, will be given in Section 2.1. Then, in Section 2.3, the HMM based phone models are explained. Sections 2.4, 2.5 and 2.6 describe how to evaluate the HMM, how to use the HMM for decoding (Viterbi algorithm) and how to train the HMM parameters (Baum-Welch), respectively.

2.1 Feature Extraction - The Mel-Frequency Cepstral Coefficients

The first step in the ASR system is to extract features from the input waveform signal. This process is performed in order to extract the properties that are important to the recognition task, and might also include techniques to make the feature extraction robust to background noise.

One of the most used feature representation of speech in ASR, is Mel-Frequency Cepstral Coefficients (MFCCs), which were proven to be beneficial for speech recognition by Davis and Mermelstein in [1]. The MFCCs are extracted using a filterbank of triangular bandpass filters separated uniformly on the Mel-Frequency Scale; a non-linear perceptually motivated frequency scale.

Figure 2.1 summarises the steps in extracting MFCC features from a speech signal. Input to the extractor is a short part of the speech signal, isolated from the complete input signal by a window. A sequence of contiguous feature vectors are extracted by repeating the process after shifting the window with a small distance. Usually the shift is shorter than the window length, resulting in overlapping frames. A common choice is to use 25 ms wide Hamming windows shifted by 10 ms.

The first block uses the FFT to find the DFT of a short part of the input signal, followed by the aforementioned filterbank with triangular bandpass filters uniformly distributed on the mel-frequency scale. The output from the filterbank block is a vector of the energy output from each filter, which

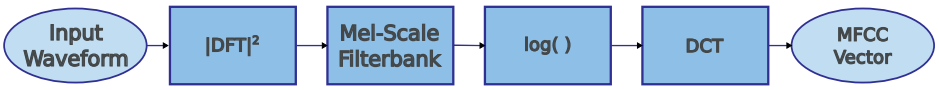


Figure 2.1: Block Diagram for MFCC Feature Extraction.

is converted to the log-energy in the next block. Finally, the discrete cosine transform (DCT) is applied to the vector, a process which reduces the dimensionality of the vector and decorrelates the vector elements. Usually, in speech with 8-10 kHz bandwidth the filterbank consists of 24 filters of which the outputs are reduced to 13 MFCCs.

Energy carries important information for the ASR system. After the DCT the 0^{th}coefficient in the MFCC vector equals the sum of the log-energy filterbank outputs, and thus represent the log-energy of the extracted frame. It is also possible to use other energy measures, as for instance the log energy of the input signal.

Temporal changes in the speech signal play an important role in human perception ([2]). The MFCCs contain information of the current analysis frame only, so to include information about surrounding frames, the delta coefficients, also called dynamic features, are usually appended. The dynamic features are estimates of the time derivative of the feature vectors, and are calculated using the surrounding frames. Adding both first and second order derivatives to the MFCC vectors greatly enhances the performance of the HMM based ASR system [3, 4, 5, 6].

2.2 The Hidden Markov Model (HMM)

The foundation of the HMM is the Markov Chain, a discrete stochastic process that takes on a finite or countable number of possible states, $\{S = i; i = 1, 2, \ldots, N\}$ [7]. In ASR the first order Markov model is usually employed. This has the important property that it is *memoryless*: the conditional distribution of any future state given the past and the present state depends only on the present state. This is also called the Markov property and can be expressed by:

$$P(S_{t+1} = j | S_t = i, S_{t-1} = i_{t-1}, \ldots) = P(S_{t+1} = j | S_t = i) = a_{ij} \quad (2.1)$$

where a_{ij} is the transition probability from state i to state j. Assuming that the number of states in the Markov chain is N, the transition probabilities may be described by an $N \times N$ matrix, the *transition matrix*. In addition to the transition matrix, the Markov chain also has a set of initial state probabilities. Hence the parameters of the Markov chain are:

$$a_{ij} = P(S_t = i | S_{t-1} = j) \quad 1 \leq i \leq N, 1 \leq j \leq N \tag{2.2}$$

$$\pi_i = P(S_1 = i) \quad 1 \leq i \leq N \tag{2.3}$$

In an HMM, the observations (feature vectors) are assumed to be dependent on an underlying unobservable (hidden) state sequence. For each time frame, the model makes a state transition according to the hidden Markov chain, and an output feature vector is generated from the output probability density function associated with the current state. The model is well suited for speech as the hidden state sequence can be associated with the various shapes the vocal tract takes during speech production. The ASR system is thus searching for the optimal state sequence $\bar{\mathbf{S}} = \{S_1, S_2, \ldots, S_T\}$ to describe the input sequence $\bar{\mathbf{x}} = \{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_T\}$. The state sequence can be mapped to a sequence of phones (or another unit of choice) and subsequently to words or sentences.

In addition to the state transition probabilities and the initial state probabilities in Equations (2.2) and (2.3), the HMM also needs an output probability density function for each state:

$$b_i(\underline{x}_t) = P(\underline{x}_t = \underline{x} | S_t = i) \tag{2.4}$$

which is commonly modelled by a Gaussian Mixture Model (GMM).

2.3 The Phone HMM

The model topology, that is the number of states and the transitions allowed to be performed in the underlying Markov chain, is an important

property for the modelling capability of the HMM. For modelling phones a popular choice is to use a left-to-right topology with about 3 - 5 states. The 3 state topology is shown in Figure 2.2:

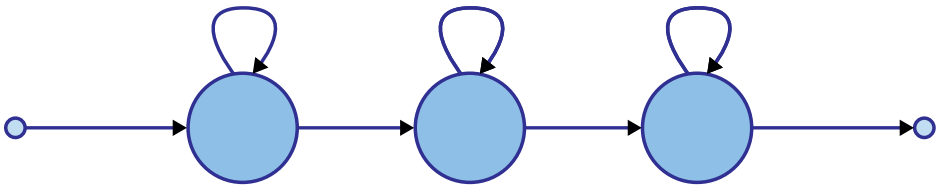


Figure 2.2: A standard 3 state left to right topology.

All the states have a transition to itself and to the next state and are thus modelling contiguous feature vectors that describes the same quasi-stationary segment. In a 3-state HMM, the three states can be viewed as modelling the beginning, middle and end of a phone realisation. It is also possible to allow the model to skip states by introducing more transitions, at the cost of more parameters. Skips are especially important if the model employs many states. Consider for instance that the features are extracted at 10 ms intervals. 5 states without skips would then require all phones to last at least 50 ms. Considering that the realisation of some phones may have duration shorter than 10 ms, it is clear that this might lead to a model mismatch resulting in improper modelling of short phone realisations.

Another crucial decision, is choosing the type of probability distribution $b_i(\underline{x}_t)$ for modelling the observation vectors. A common choice is to use the Gaussian Mixture Model (GMM), which is versatile and yields good performance. A drawback of the GMM is the number of parameters, which is a factor that needs attention in ASR due to limited access to training data. Employing a full covariance matrix for all Gaussians in an HMM system requires a large number of parameters to be estimated. An often used approximation is to assume that the feature vector components

are uncorrelated and use non-zero elements only on the diagonal. This gives a considerable reduction in the amount of parameters, at the cost of introducing a possible error source. The DCT used in the last step of the extraction process of the MFCCs has decorrelation properties, which means that vector elements have low correlation, reducing error introduced by using diagonal covariance matrices.

2.4 Evaluation of the HMM

Given an HMM Φ and an observation sequence $\bar{\mathbf{X}} = \underline{x}_1, \underline{x}_2, \ldots, \underline{x}_T$, the probability that the model generates the observations, $P(\bar{\mathbf{X}}|\Phi)$, describes how well the model fits the observations. The brute force solution to this problem is to enumerate all possible state sequences $\bar{\mathbf{S}} = S_1, S_2, \cdots, S_T$ and sum the probabilities:

$$P(\bar{\mathbf{X}}|\Phi) = \sum_{\bar{\mathbf{S}}} P(\bar{\mathbf{S}}|\Phi) P(\bar{\mathbf{X}}|\bar{\mathbf{S}}, \Phi) \tag{2.5}$$

Unfortunately, there are in the order of $O(N^T)$ state sequences, N being the number of states, yielding an exponential computational complexity. The *Forward Algorithm*, however, utilises the independence properties and the similarities between possible state sequences. This is done by defining a set of forward probabilities:

$$\alpha_t(i) = P(\bar{\mathbf{X}}_1^t, S_t = i|\Phi) \tag{2.6}$$

where $\bar{\mathbf{X}}_1^t = \underline{x}_1, \underline{x}_2, \ldots, \underline{x}_t$. The forward probability, $\alpha_t(i)$, is the probability that the HMM is in state i at time t and have produced the output vectors X_1^t. This probability can be calculated iteratively:

$$\alpha_t(j) = \left(\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} \right) b_j(\underline{x}_t), \quad t = 2, \ldots, T \tag{2.7}$$

with the initialisation:

$$\alpha_1(i) = \pi_i b_i(\underline{x}_1). \tag{2.8}$$

In ASR it is common to require the model to end in a special final state (which is non-emitting) at time $T+1$. The likelihood $P(\bar{\mathbf{X}}|\Phi)$ in Equation (2.5) can then be found by using the calculated forward probabilities:

$$P(\bar{\mathbf{X}}|\Phi) = \alpha_{T+1}(N) \tag{2.9}$$

The forward algorithm thus calculates the probability in Equation (2.5) with a complexity of $O(T \cdot N^2)$, which is a significant improvement compared to the brute-force solution.

2.5 Decoding the Input Sequence

The main problem in an HMM based speech recogniser is to find the best state sequence for a given utterance. The state sequence corresponds to the sequence of recognised units subsequently translated into phonetic labels. The most widely used criterion for the best state sequence is the sequence that has the highest probability of being taken while producing the observation sequence. In other words: given an HMM Φ and a set of input vectors $\bar{\mathbf{X}} = \underline{x}_1, \underline{x}_2, \ldots, \underline{x}_T$, the problem is to find the state sequence $\bar{\mathbf{S}} = S_1, S_2, \ldots, S_T$ that maximises $P(\bar{\mathbf{S}}, \bar{\mathbf{X}}|\Phi)$.

The problem is solved efficiently using the Viterbi Algorithm which is based on dynamic programming. The Viterbi algorithm is similar to the forward algorithm used for evaluating the HMM, but searches for the best state sequence rather than summing over all the possible paths. Hence, the summation in Equation (2.7) is replaced with a max operator:

$$V_t(j) = \max_{1 \le i \le N} \left[V_{t-1}(i) a_{ij} \right] b_j(\underline{x}_t) \tag{2.10}$$

where $V_t(i) = P(X_1^t, S_1^{t-1}, s_t = i|\Phi)$ is the probability of the most likely state sequence that generated the observations $\bar{\mathbf{X}}_1^t = \underline{x}_1, \underline{x}_2, \ldots, \underline{x}_t$ and ends in state i. The initialisation is the same as the initialisation of the forward algorithm:

$$V_1(i) = \pi_i b_i(x_1) \tag{2.11}$$

The most likely state sequence is found by saving the argument yielding the maximum value in Equation (2.10) for each state at each time instance, and then perform back tracking from the state with the highest value at the final time frame.

Instead of remembering only the best state sequence for every state at every time instance, it is possible to remember more alternatives, at the cost of a significant increase in memory usage. The advantage is that a list of the most promising sentence hypotheses can be given, instead of only the one in a normal decoding. This is referred to as N-Best decoding, N being the number of hypotheses.

2.6 Training of the HMM Parameters - Baum-Welch

In order to use the HMM for ASR, the parameters of the model have to be estimated. The most common method for training an HMM is the Baum-Welch algorithm [8], which is based on the Expectation-Maximisation algorithm (EM algorithm). A tutorial of the EM algorithm is given in [9]. The Baum-Welch algorithm is an iterative method, which for each iteration maximises the likelihood of the data given the model $P(\bar{\mathbf{X}}|\Phi)$. The new set of parameters $\hat{\Phi}$ is found according to the EM algorithm, by first

taking the conditional expectation of the complete data log likelihood:

$$E_{\Phi}\left[\log(\bar{\mathbf{X}}, \bar{\mathbf{S}}|\hat{\Phi})\right]_{\bar{\mathbf{S}}|\bar{\mathbf{X}}} = \sum_{\bar{\mathbf{S}}} P(\bar{\mathbf{S}}|\bar{\mathbf{X}}, \Phi) \log(P(\bar{\mathbf{X}}, \bar{\mathbf{S}}|\hat{\Phi})) \tag{2.12}$$

with the current parameters Φ, before finding the $\hat{\Phi}$ that yields the maximum.

To avoid summing over all possible state sequences the Baum-Welch algorithm utilises the forward probabilities defined in Section 2.4 and a set of backward probabilities:

$$\beta_t(i) = P(\bar{\mathbf{X}}_{t+1}^{T}|s_t = i, \Phi). \tag{2.13}$$

Similar to the forward probabilities, these are computed inductively over t with initialisation:

$$\beta_{T+1}(N) = 1 \tag{2.14}$$

where the HMM is required to end in the final state, and then the induction:

$$\beta_t(i) = \left[\sum_{j=1}^{N} a_{ij} b_j(X_{t+1}) \beta_{t+1}(j)\right], \quad t = T, \ldots, 1 \tag{2.15}$$

which is performed in the time-reversed direction. Hence, the summation in Equation (2.12) may be replaced by a summation over time, previous and current state. The complete data log likelihood can then be expressed

in terms of the α and β:

$$
\begin{aligned}
E_\Phi\left[\log(\bar{\mathbf{X}},\bar{\mathbf{S}}|\hat{\Phi})\right]_{\bar{\mathbf{S}}|\bar{\mathbf{X}}} &= \sum_{\bar{\mathbf{S}}} P(\bar{\mathbf{S}}|\bar{\mathbf{X}},\Phi)\log(P(\bar{\mathbf{X}},\bar{\mathbf{S}}|\hat{\Phi})) \\
&= \sum_{\bar{\mathbf{S}}} \frac{P(\bar{\mathbf{S}},\bar{\mathbf{X}}|\Phi)}{P(\bar{\mathbf{X}}|\Phi)}\log(P(\bar{\mathbf{X}},\bar{\mathbf{S}}|\hat{\Phi})) \\
&= \sum_i \sum_j \sum_t \frac{P(S_{t-1}=i,S_t=j,\bar{\mathbf{X}}_1^T|\Phi)}{P(\bar{\mathbf{X}}|\Phi)}\log(P(\bar{\mathbf{X}},\bar{\mathbf{S}}|\hat{\Phi})) \\
&= \sum_i \sum_j \sum_t \frac{\alpha_{t-1}(i)a_{ij}b_j(\underline{x}_t)\beta_t(j)}{\sum_{k=1}^N \alpha_T(k)}\log(P(\bar{\mathbf{X}},\bar{\mathbf{S}}|\hat{\Phi})) \\
&= \sum_i \sum_j \sum_t \gamma_t(i,j)\log(P(\bar{\mathbf{X}},\bar{\mathbf{S}}|\hat{\Phi})) \qquad (2.16)
\end{aligned}
$$

where $\gamma_t(i,j)$ is the probability of taking the transition from state i to state j at time t given the model and the observation sequence. The optimal parameters are then found by adding Lagrange multipliers and maximising with respect to each parameter individually.

The Baum-Welch algorithm guarantees a non decreasing likelihood. However, this is not the same as non decreasing performance as the discriminative capabilities of the HMM is not trained explicitly. Discriminative training of the HMM can be done using methods like maximum mutual information (MMI), minimum classification error (MCE), minimum word error (MWE) and minimum phone error (MPE) (see e.g. [10]). However, these require significantly more computational power. In this dissertation discriminative training was not used.

2.7 Performance Evaluation

The performance of a given ASR system can be evaluated by using a labelled test set and compare the recognised words with the spoken words. In the comparison there are three types of word recognition errors that can occur:

Substitution (S): A correct word was substituted by an incorrect word

Insertion (I): A new incorrect word was inserted

Deletion (D): A correct word was removed

In order to find these errors the two word sequences are aligned by dynamic time warping (DTW). The word error rate (WER) is defined as:

$$\text{WER} = \frac{\text{S} + \text{I} + \text{D}}{N} \cdot 100\% \tag{2.17}$$

where S, I and D is the number of substitutions, insertions and deletions respectively, while N is the number of spoken words. The performance of the recogniser can also be expressed by the accuracy, defined as:

$$\text{Accuracy} = 100\% - WER \tag{2.18}$$

The performance of a large vocabulary speech recogniser LVCSR is dependent on the accuracy of the recognised units. Although some errors made by the phone recognition step in an LVCSR are corrected by the dictionary and language models, it is a fair assumption that improving the phone recognition performance also improves the LVCSR.

Evaluating an ASR system on the phone level can be done in exactly the same manner as for words. This requires a labelling of the test utterances on the phone level.

2.8 HMM Variations

As mentioned, the HMM has been widely used in the ASR community. Naturally, several modifications and variations of the HMM have been proposed in the literature, some more successful than others. An overview of many of the variations is given in [11] and [12]. Because of its popularity, the research performed on the HMM is extensive. In this section some of the models derived from the HMM are presented.

Today, the computational time for training HMMs with continuous emission densities is low. However, a couple of decades ago, the computational complexity of such models was a concern. The discrete HMM, where a vector quantisation (VQ) was applied to the input in order to use discrete output probability functions, offered faster training and decoding times at the cost of introducing VQ distortion. In [13, 14] a compromise between the continuous HMM and the discrete model, the semi-continuous HMM was proposed. The semi-continuous HMM was trained using a discrete HMM, but included the training of a probability density function for each entry in the VQ codebook. During decoding weighted sums of the pdfs in the VQ codebook were used as emission densities.

The emission densities of the HMM are dependent on the current state only, and are independent of the previous (or succeeding) output vectors. Since speech is created by a relatively slowly moving vocal tract, this is clearly an incorrect assumption, specially when dynamic features are used, as they are directly dependent on the surrounding frames. In [11] an overview of different approaches for segmental HMM are given. In segmental HMM each state outputs a sequence of feature vectors, called a segment. The length of the segment is a random variable. This way it is possible to model the dependencies between the observation vectors in one segment. Similar to the segmental HMM the trajectory HMM (e.g.[15, 16, 17]) allows the the gaussian components in the GMMs to

change during a state, and adds dependencies on neighbouring states as well.

Another shortcoming of the conventional HMM is that the state durations are modelled implicitly by a geometric distribution. In [18, 19, 20, 21] approaches to implement more appropriate distributions are proposed. In the Hidden Semi-Markov Model (HSMM), the duration is explicitly modelled. Unfortunately, this leads to the Markov assumption becoming invalid, an assumption which the Baum-Welch and Viterbi algorithms are dependent on. The problem is investigated in [20], where the computational burden is reduced by imposing a limit to the number of frames allowed in each state. In the Expanded State HMM (ESHMM) [22], the duration is modelled by a Markov chain.

Several efforts have been made to improve the acoustic models of the HMM. The standard is to employ GMMs as emission densities. However, other approaches have also been tried. For instance the Linear predictive HMM [23] or the AR HMM [8, 24], use output probability distributions derived from the souce-filter model which assumes that speech can be modelled as noise filtered by an all-pole filter. In the source-filter model the filter models the vocal tract, which shapes the airflow from the lungs into speech sounds. Another approach has been to combine the HMM with Artificial Neural Networks (ANNs), known as Hybrid HMM/ANN approaches [25, 26, 27], where the ANNs represents the emission densities of the states. Recently, hybrids with Deep Belief Networks (DBNs), which is a layered network composed of stochastic variables, have proven to be successful [28, 29, 30].

Chapter 3

HMM Based on Phonetically Interpreted Acoustic Subword Units (ASWUs)

Conventional HMM-based speech recognisers normally employ 3-state phone models to handle the acoustic modelling. When training the phone models, speech data and an associated phonemic transcription are used. The model topology is normally identical for all phone units. If there is a need for reducing the number of model parameters, e.g. for context dependent phone modelling, or if there is insufficient training data available, data sharing strategies such as state tying are employed.

Training phonetically defined units like phones puts restrictions on the use of the available training data. Even though similar acoustic events may occur in different phonetic units, conventional training strategies are

unable to exploit this. Recognition strategies based on sub-phonemic units may be able to better utilise similar acoustic events regardless of phonetic membership.

In the conventional phone models, the uniform 3-state left-to-right model topology can limit the capability of modelling diverse acoustic realisations of the phones. Depending on factors like speaking style and rate, phonetic and syllabic context as well as the prosodic structure, phone realisations may exhibit variation in the number of distinct acoustic events. A richer phone model topology may be able to provide a more accurate representation of natural pronunciation variation at the phone level.

In this chapter a phone recognition system that utilises a set of acoustic subword units (ASWUs) to model the phones is proposed. The ASWUs are defined as a set of short units with stable acoustic properties and are used as building blocks in the phone models. A phone is assumed to be produced as a sequence of one or more acoustical events which in turn is modelled by the ASWUs. In the next section, an overview of the proposed system will be given, before the acoustic subword units are introduced in Section 3.2. Further, in Section 3.3 the training scheme for the proposed system is described. Finally, the experiments conducted with the system are presented in Section 3.4, followed by a discussion of the results in Section 3.5 and a concluding summary in Section 3.6

3.1 System Overview

The phone recognition system proposed in this chapter differs from the conventional 3 state left-to-right HMM by extending the topology of the phone HMM. Instead of only using a single path with 3 states, it is offering alternatives for the number of states to use for a phone realisation. This is done by introducing several parallel paths, where each path is a left to

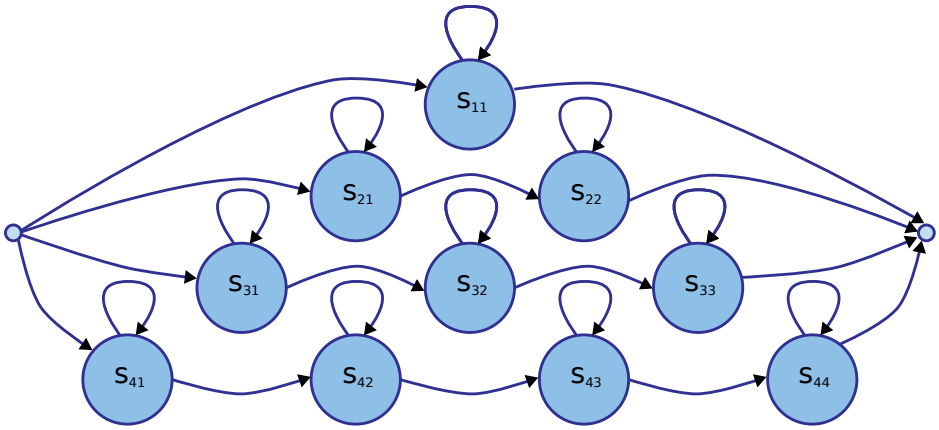


Figure 3.1: An example of the proposed phone HMM topology.

right model with different number of states. An example of this is shown in Figure 3.1 where 4 paths with 1, 2, 3 and 4 states are available to the model. The hypothesis is that this will better cope with the great variation in which a phone may be realised, dependent on e.g. speaker, context and stress. A phone may be realised using a variable amount of acoustic events and extending the phone model in this manner will give better terms for modelling each acoustic event accurately by a state. Variability in phone pronunciation can thus be better modeled with this topology than the standard left-right topology which relies on a rich formulation of the state emission densities to accommodate a reasonable degree of variability.

A standard training procedure using the Baum-Welch algorithm described in Chapter 2.6, can lead to data sparsity problems and result in poor performance because of the increase in parameters used by each phone model. Therefore, an alternate approach is required to train the models robustly. As indicated, the training procedure will define and utilise a set of ASWUs to alleviate the data sparsity problems. The ASWUs are, as will be discussed in detail in Section 3.2, found by first dividing the input

utterances into segments with stable acoustic properties, before clustering the segments. Each cluster then corresponds to an ASWU which can be linked to the states in the HMM topology by a linguistic interpretation. In the next section the ASWUs will be presented before a more detailed description of the training scheme is given in Section 3.3.

3.2 Acoustic Subword Units

In a traditional speech recogniser the input signal is segmented by a linguistic interpretation of the acoustic signal. The HMM both performs segmentation of the signal and links each segment to a unit during the Viterbi decoding described in Section 2.5. The result is a sequence of units (e.g. phones) that can be converted to words.

Another approach is to define units purely based on the acoustic properties of the signal. In [31, 32] acoustic subword units, or *fenones*, were defined by first performing a vector quantisation of the input sequence. The VQ codebook is then used to label the input sequence, where each label corresponds to a fenone. The fenones describes very short acoustic events, often as short as one frame, and they are modelled by 1 state HMMs. Finally, word models were constructed as a sequence of fenones. In [33, 34] an ASR system based on segment models or acoustic segment units (ASU) was proposed. The segment models were created by a maximum likelihood segmentation, where the frames of a segment were assumed to be generated from a single AR model. These segments were then clustered or quantised into groups of acoustically similar clusters, which were modelled by simple HMMs. Finally, word models were created by concatenating the units through an acoustic lexicon.

In this chapter a set of acoustic subword units (ASWUs) similar to the ASUs in [33, 34, 35] will be used. The input speech signal is first divided

into short segments by a constrained clustering approach described in Section 3.2.1, before each segment is clustered or quantised by the approach presented in Section 3.2.2.

3.2.1 Automatic Acoustic Segmentation

Automatic acoustic segmentation of speech is the task of finding a set of boundaries $\{b_1, b_2, \ldots, b_J\}$ based on the acoustic properties of the signal. When recognising speech using HMMs, the HMM is performing segmentation as part of the recognition process. However some research has also been conducted on the task of segmenting the speech signal separately. In [36, 37, 38] the authors proposed a procedure where each frame was compared to its near neighbours by calculating the Euclidean distance between the feature vectors. The procedure moves from left to right and associates each frame with either the past or the future, whichever has the greatest similarity. Boundaries are inserted when the direction of the current frames association changes from past to future.

A segmentation approach based on dynamic programming (DP) was proposed in [39], and used later in [35, 40, 41, 42, 43]. The approach approximates each segment with a polynomial, and finds the segmentation that yields the best approximations. In [44] each feature dimension of the acoustic segments was modelled with a trajectory approximation. Each dimension was assumed to be a noisy representation of a quadratic polynomial. In [41, 42, 45] the DP approach and the trajectory models were combined.

In this dissertation, the DP approach will be employed with a constant approximation of the segments. The DP approach is a procedure which iteratively increases the number of segments until a stopping criterium is met. For a given number of segments J and the input utterance $\{\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_T\}$, the algorithm finds the segmentation $\{S_1, S_2, \cdots, S_J\}$

with boundaries $\{b_0, b_1, \ldots, b_J\}$ that minimises:

$$D(J) = \frac{1}{T} \sum_{j=1}^{J} \sum_{\forall i \in S_j} d(\underline{x}_i, \underline{c}_j(i)) \tag{3.1}$$

where $d(\cdot)$ is a distortion measure, S_j is the jth segment of consecutive speech feature vectors $\{\underline{x}_i; b_{j-1} < i \leq b_j\}$ and $\underline{c}_j(i)$ is the vector that represents the ith frame of the jth segment. In this dissertation a constant segment representation has been used, and $c_j(i)$ is replaced by c_j. Note that $b_0 = 0$, while $b_J = T$.

The algorithm initiates with only two segments, and iteratively increases the number of segments until one of the terminating conditions:

$$J = J_{\max} \vee D(J) \leq \theta \tag{3.2}$$

is fulfilled. These constraints limit the algorithm to maximally produce a predetermined number of segments $J_{\max}$, or to stop when the segmentation yields a piecewise constant approximation which deviates less than the given threshold θ from the original signal. As seen in Equation (3.1) the distortion of a segmentation is scaled with the number of frames T. This way the distortion threshold θ is not dependent of the length of the utterance, and may be fixed for all utterances. When the phone sequence of the input is available it is possible to disable the distortion criterion ($\theta = 0$) and instead fix the number of segments based on the number of phones. A simple choice is to multiply the number of phones in the utterance with a fixed factor, ρ:

$$J_{\max} = \rho * N_{\text{phones}} \tag{3.3}$$

Since there are generally more acoustic incidents than phones in an utterance, this factor should be larger than 1, i.e an *over-segmentation.*

The distortion measure $d(\cdot)$ in Equation (3.1) and the segment representation function $\underline{c}_j(i)$ controls the properties of the segments produced by the

segmentation. In [44] segments were modelled using polynomial trajectories, which is a reasonable representation function for this segmentation algorithm. However, since segments with approximately constant acoustic properties are desired, only the constant term in the polynomial is used in this dissertation. In other words, the frames are compared to the centroid of the segment and thus ensures that the intra-segment variation is kept at a minimum. The distance measure that has been chosen is simply the Euclidean distance of the two argument vectors.

3.2.2 Segment Clustering

After the automatic acoustic segmentation, where a set of boundaries has been assigned to the input data, the acoustic segments are clustered. By collecting acoustically similar segments together, the result will be a set of clusters with low acoustic variability that covers the output space. It is therefore a fair assumption that these clusters are describing various acoustic events that occur in the speech data set. Each of these clusters defines one Acoustic Subword Unit (ASWU).

Since the choice of a 0^{th}order segment approximation in the acoustic segmentation algorithm results in segments with low intra-segment variability, the centroid is a reasonable representation of each segment. The clustering of the segments can then be performed by clustering the centroid vectors. This can be done using the HMM Toolkit HTK [46] which performs the vector quantisation "by a top-down clustering process where clusters are iteratively split until the desired number of clusters are found". Also for the clustering the Euclidean distance metric was chosen.

3.3 Parameter Estimation

Since the number of states in the phone models has been increased compared to the standard topology for context independent phone HMMs (10 vs 3 states) standard training procedures can be sensitive to data sparsity. In addition, similar acoustic events may occur in different phone realisations, or even in different phones, something that could be utilised to share training data. Hence, a training scheme inspired by the procedure described in [43] is proposed in this section. The scheme consists of several steps:

1. Automatic acoustic segmentation
2. Create the ASWUs by clustering the acoustic segments
3. Training of a GMM for each ASWU
4. Linguistic interpretation of the acoustic segments
5. Combining ASWU GMMs into state GMMs
6. Final embedded re-estimation

The first two steps, which have already been explained, results in an inventory of ASWUs based on acoustic segments. Each ASWU consists of a collection of acoustic segments which can be used to train GMMs. The linguistic interpretation, which will be described in the next section, assigns each of the acoustic segments to one of the states in the HMM. Each acoustic segment is then assigned both to an ASWU and an HMM state. This creates a link from the states of the HMM to the ASWUs through the acoustic segments. In Section 3.3.2 a description of how this link can be used to construct the state GMMs as a linear combination of the ASWU GMMs will be given.

3.3.1 Linguistic Interpretation of Acoustic Segments

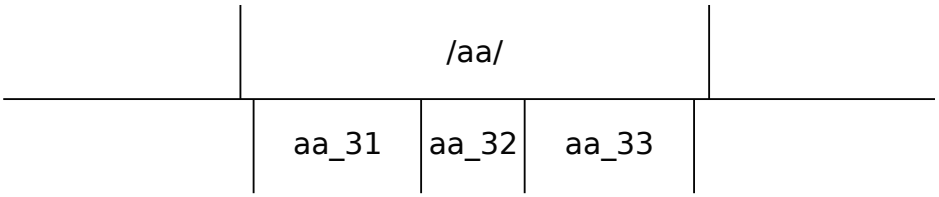
After the acoustic segmentation, the segments are aligned with a phonemic labelling. This labelling may be produced manually as in e.g. the TIMIT database, or automatically with a standard speech recogniser and forced alignment. The aligned labels are then compared and the acoustic segments are assigned to a manual segment. Based on this assignment, the acoustic segments are given a linguistic label, which creates a link between the acoustic segments and the states in the HMM.

In Figure 3.2 four examples of possible alignments are shown. The outcome of the comparison between an acoustic segment with the manual labels can be one of the three cases:

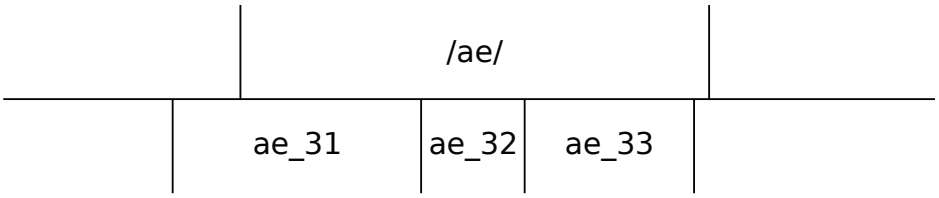
1. the acoustic segment is fully contained in a manual segment (Figure 3.2(a))
2. the acoustic segment spans one manual boundary (segment "ae_31" in Figure 3.2(b))
3. the acoustic segment spans multiple manual boundaries (Figure 3.2(c) and Figure 3.2(d))

For the first case, the phonemic assignment is straightforward as Figure 3.2(a) shows. In all other cases, a *membership function* is employed: for an acoustic segment "$\mathcal{A}$" with boundaries at $t = a_b$, $t = a_e$ and a manual segment "$\mathcal{M}$" with boundaries at $t = m_b$, $t = m_e$, the membership function f is given by the portion of the phonemic segment covered by the acoustic segment. I.e. the length of the intersection between the manual and the acoustic segments divided by the length of the manual segment:

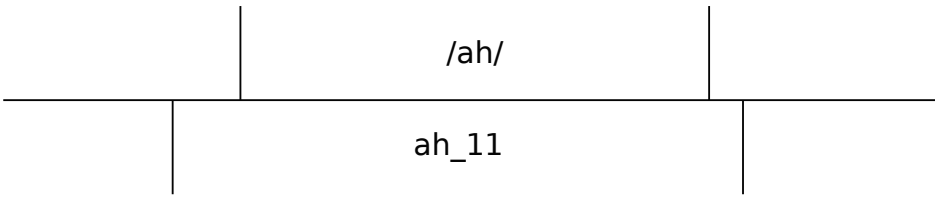
$$f(\mathcal{A},\mathcal{M}) = \begin{cases} \frac{m_e - \max(a_b, m_b)}{m_e - m_b} & \text{if } a_b < m_e < a_e \\ \frac{\min(a_e, m_e) - m_b}{m_e - m_b} & \text{if } a_b < m_b < a_e \end{cases} \tag{3.4}$$



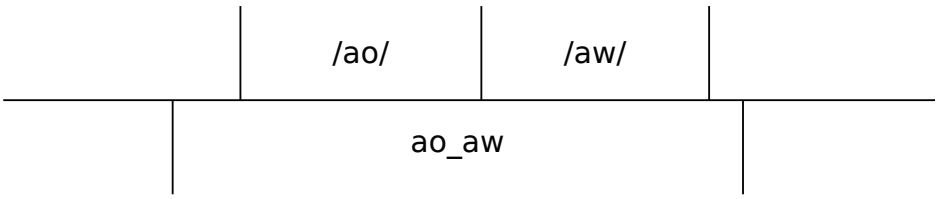
(a) Example with acoustic labels fully contained in a manual segment.



(b) Example with an acoustic label that covers a manual label.



(c) Example with an acoustic label that spans an entire manual segment



(d) Example with an acoustic label that covers several manual segments

Figure 3.2: Four example cases of labelling acoustic segments (lower) after comparison with phonetic labels (upper). In (a) the simple case where all the acoustic labels are fully contained in a manual segment. (b) shows the case where an acoustic segment (ae_31) contains a manual boundary. (c) shows an example where the acoustic segment covers two manual boundaries. Finally, in (d) is an example of the unfortunate case where the acoustic segment covers more than one manual segment.

which will be equal to 1 if both conditions are fulfilled, i.e. the phonemic segment is fully contained within the acoustic segment.

For the second of the three cases above, the acoustic segment will simply be assigned to the manual segment yielding the highest membership measure. For the segment "ae_31" in Figure 3.2(b) the membership measure will be highest for "/ae/" compared to the preceding manual segment. The relative overlap of "/ae/" is higher.

In the third case the acoustic segment is assigned to all manual segments with membership measure higher than 40%. This implies that some of these segments will be assigned to more than one manual segment. The third case can thus be divided into two subcases: one where the acoustic segment is assigned to one manual label and one where the acoustic segment is assigned to multiple manual labels. In Figure 3.2(c) an example of an acoustic segment that spans two manual boundaries are shown. Since the membership measure will be less than 40% for both the preceding and succeeding manual segments, the acoustic segment is assigned to "/ah/" only. If the segment had covered more of either neighbour of "/ah/", the segment had been assigned to both. This is the case in Figure 3.2(d), where the acoustic segment covers two entire phone segments, and is assigned to both.

It is desired to use the acoustic segments to provide a systematic description of the variability of the phone realisations. The number of acoustic segments associated with a specific phone segment is an indication of the manner in which the phone is realised. Also, the position of an acoustic segment in a multi-segment realisation of the phone points to a specific acoustic event in the phone realisation. Thus, information about the number of acoustic segments and the position of the segment is included in the linguistic label. All the acoustic segments are labelled according to three properties: name of the phone the segment is assigned to, the number of acoustic segments assigned to that specific phone instance and the

position of the acoustic segment. In Figure 3.2, the labelling of the previous mentioned cases is shown. Note that for the case where the acoustic segment is assigned to multiple phones, as in Figure 3.2(d), the proposed labelling convention is inadequate. Instead the segment is named based on the phones it is assigned to. These segments are not considered in the next step, where the linguistic labels are used to map the acoustic segments to a state in the HMM. Since there is no limit to how many acoustic segments that can be assigned to a manual segment, it is possible that too many acoustic segments are assigned to a manual segment. Each segment corresponds to a state in the HMM, and as the topology in Figure 3.1 shows, there is a maximum number of states a given realisation can be modelled with. In this dissertation the maximum number of states was 4, hence all phones assigned more than 4 acoustic segments yield a problem. A simple solution is to merge the segments down to 4 states. This was done by merging the two consecutive segments with lowest Euclidean distance between the centroids, until the number of segments is 4. The solution is not optimal as the other boundaries are not moved and the resulting segmentation is not likely to be the best in terms of maximum likelihood. Hence, the automatic acoustic segmentation algorithm needs to be adjusted carefully in terms of how many segments it produces. This will be investigated further in Section 3.4.1.

3.3.2 HMM Training

The ASWUs, which are modelled by GMMs, are created by clustering the acoustic segments. Therefore, there is a link from each acoustic segment to an ASWU. During the linguistic interpretation of the acoustic segments, the acoustic segments were also linked with HMM states. Hence, through the acoustic segments, each HMM state is linked to several ASWUs. An example of the situation is shown in Figure 3.3. In this example the state "S_1" is assigned to only one ASWU, whereas the state "S_2" has one link

to each ASWU. The links from an HMM state to the different ASWUs can be used to represent the emission density of the state as a weighted sum of the ASWU GMMs. The weights in the sum can be determined by counting the number of links from the HMM state to each ASWU. Let the set of acoustic segments assigned to the state "s" be denoted by $\mathcal{U}_s$ and the set of acoustic segments assigned to ASWU number i be denoted $\mathcal{C}_i$, then the weight w_i^s of the cluster GMM i for the state "s" is given by:

$$w_i^s = \frac{|\mathcal{U}_s \bigcap \mathcal{C}_i|}{|\mathcal{U}_s|} \tag{3.5}$$

where $|\cdot|$ denotes cardinality. The emission density of state "s" is thus given by:

$$p(\underline{x}|s) = \sum_{i=1}^{C} w_i^s \sum_{j=1}^{M} c_{ij} \mathcal{N}(\underline{x}|\mu_{ij}, \Sigma_{ij}) \tag{3.6}$$

where C is the total number of clusters and M is the number of mixture components in each cluster GMM which is represented by the last summation.

In Equation (3.6), the state output density functions are expressed by a linear combination of the ASWU GMMs. Only the weights of the combination differ between the states, while all the Gaussian components are shared. The situation is similar to the semi-continuous HMM (SC-HMM) where the state output probability functions was created as a weighted sum of pdfs from a VQ codebook (see [13, 14]). This makes it possible to have larger GMMs without the increase in parameters. In addition to using the GMMs defined by Equation (3.6) as state emission densities, there are two other alternatives that will be tested in this dissertation. First, it is possible to collect all the shared Gaussian components in a shared pool, and allow a Baum-Welch re-estimation to update the weights for each state. I.e a tied Gaussians system. The other is to treat the GMMs from Equation (3.6) separately for all the states, and allow a Baum-Welch

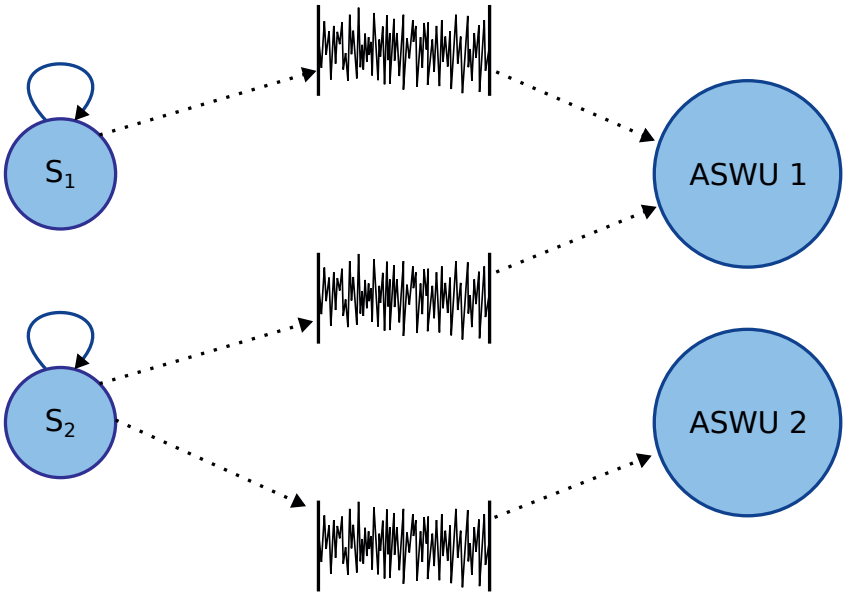


Figure 3.3: Example of links from HMM states to ASWUs through acoustic segments.

re-estimation to update both the weights and Gaussian components. This increases the freedom for the state to change its GMM during a succeeding Baum-Welch re-estimation. Unfortunately, this freedom comes at the cost of a considerable growth in the number of distinct parameters in the HMM system. To alleviate the problem, the number of mixture components in the state GMMs is reduced by a pruning approach. The pruning is conducted in two steps:

1. All GMM components with a weight lower than a threshold given by the largest mixture weight are removed. For instance remove components with weights lower than 15% of the largest weights, which is the threshold used in this dissertation.

2. Iteratively locate and merge the two components that results in the lowest log-likelihood cost when replaced with one component [47], until the a target GMM size is reached.

The first step is a fast pruning of the mixture components, while the second step is considerably slower. Hence, the number of mixture components after the first step should be small to avoid using too much time in step two. On the other hand: since the pruning does not consider coverage of the output space, too coarse pruning may result in poor modelling of some parts of the output space.

To summarise, three different approaches have been suggested:

1. Use the GMMs defined by Equation (3.6) directly, and only update the transition probabilities, including the probability of each path given the phone (the leftmost transitions in Figure 3.1).
2. Put the Gaussian components from Equation (3.6) into a shared pool of mixtures and employ a tied mixture system. The final Baum-Welch re-estimation process then updates the weights in the GMMs, in addition to the transition probabilities as before.
3. Allow the states to have individual GMMs which can be fully updated by a Baum-Welch re-estimation procedure. This enforces pruning of the GMMs to avoid a blow up in the parameter count.

3.4 Experiments

In this section, experiments conducted with the segmentation algorithm and the proposed HMM system will be presented. The experiments have been performed with the TIMIT database described in more detail in Appendix A. The training set consists of 3296 utterances, and the test set contains 1344 utterances. The models were trained with a 48 phone set

which was mapped down to a 39 phone set during testing (see Appendix A). Two different set of features were used; one for the segmentation procedure and one for training the HMMs. The segmentation was performed with 13 MFCCs, including C_0, extracted with a 15 ms window and a 5 ms shift. The HMMs were trained using 13 MFCCs including C_0, and the delta and acceleration coefficients, making a feature vector dimension of 39.

There are several aspects to the proposed system and training scheme that needs investigation. First, the impact of the segmentation algorithm will be explored, before the parameters of the training scheme and the model itself will be tested.

3.4.1 Adjusting the Segmentation Algorithm

When manual labels are present, the number of segments can be controlled by either the over-segmentation factor or the distortion threshold. The over-segmentation factor limits the number of acoustic segments to a constant factor multiplied with the number of manual labels. The result is a predictable number of segments, but the over-segmentation factor cannot be used when the manual labels are not present. In such cases one has to use a comparable transcribed data set, and find a threshold that yields the desired over-segmentation on average. That threshold can then be used for the untranscribed data. Anyway, as TIMIT is manually labelled, the over-segmentation factor was used, and the distortion threshold was set to zero in all the experiments.

In Figure 3.4 and Figure 3.5 the distribution of the number of segments per phone is shown for a selection of phones when the over-segmentation factor is varied between 2 and 5. A complete overview of all the phones is given in Appendix B. The figures have a separate category for the case where a phone is assigned more than 4 segments. However, in later steps

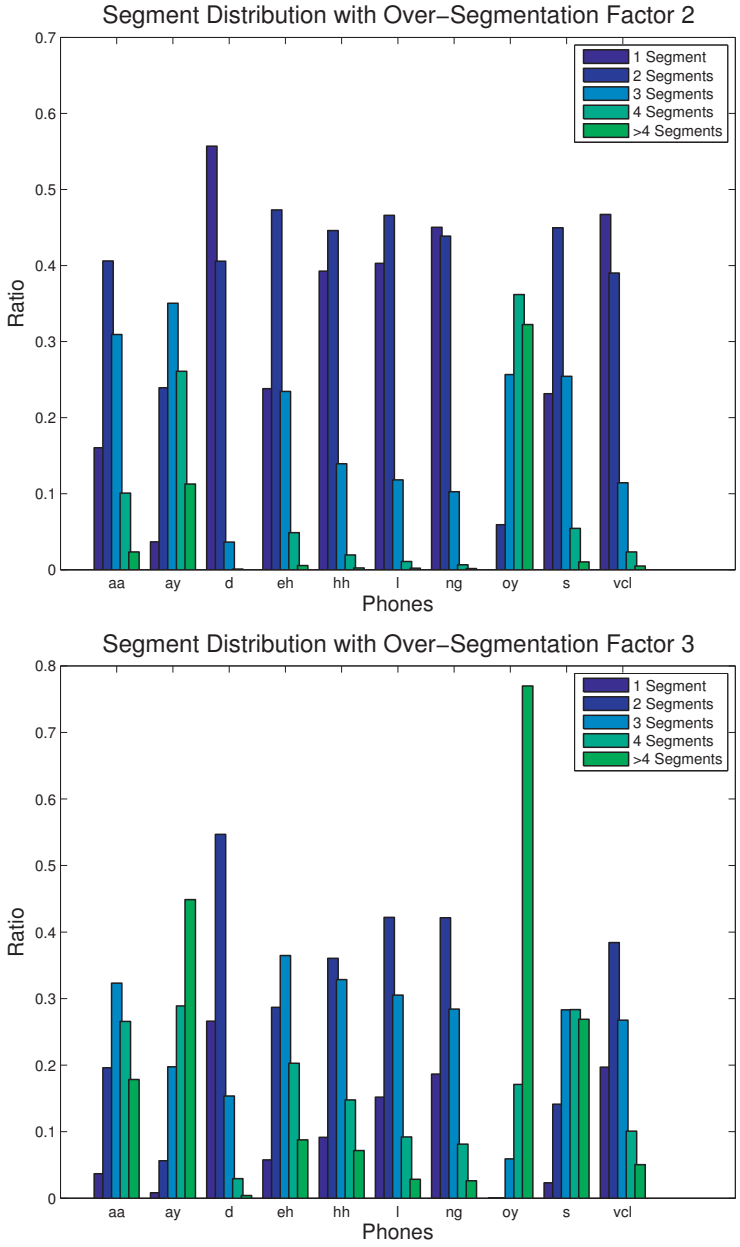


Figure 3.4: Distribution of number of segments per phone when the over-segmentation factor is 2 (upper figure) and 3 (lower figure).

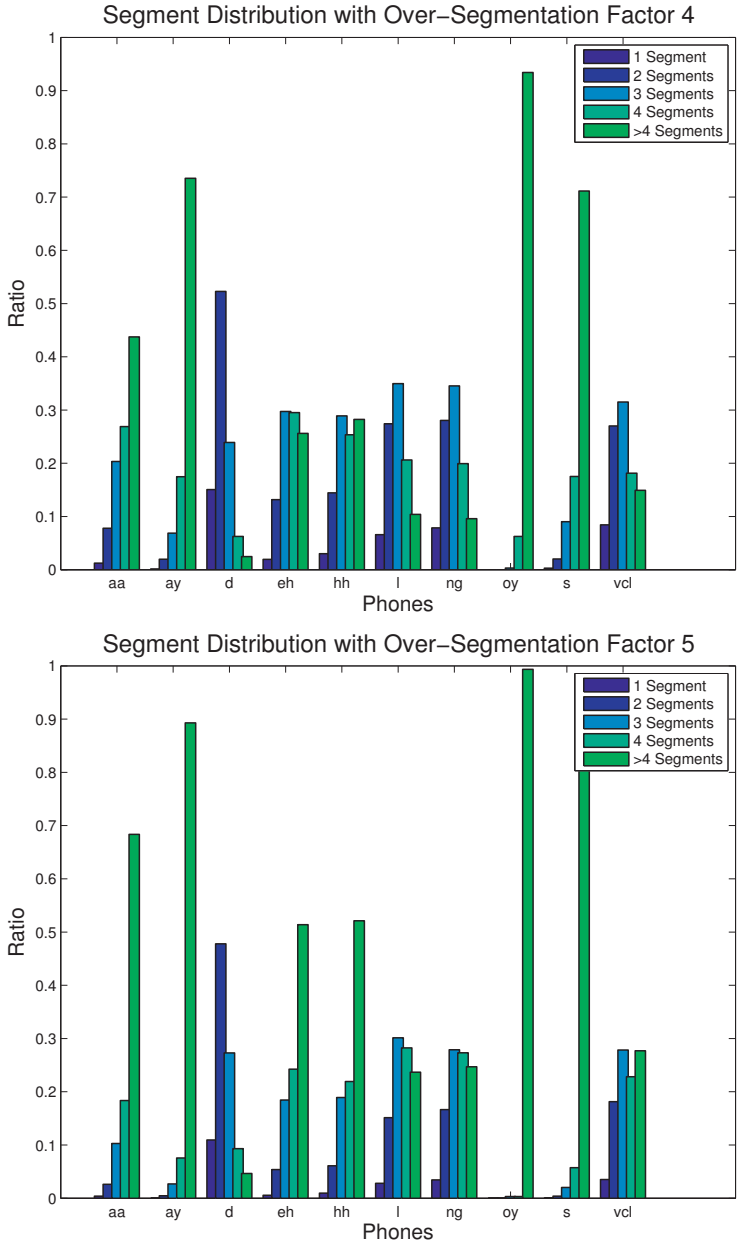


Figure 3.5: Distribution of number of segments per phone when the over-segmentation factor is 4 (upper figure) and 5 (lower figure).

these instances will be merged down to 4 segments. Acoustic segments covering more than one phone are not contributing to the figure nor in the later training steps (except the final Baum-Welch re-estimation procedure). Thus, a low over-segmentation factor will result in less data to train the cluster GMMs, whereas a high over-segmentation factor will naturally lead to an increased number of segments being merged. The figures show this expected relation: low over-segmentation factor moves the ratios to the left, while higher over-segmentation moves them to the right. Another observation that can be made from the figures is the difference between the phone classes: short phones like stops tend to have more instances with few segments compared to longer phones like diphthongs. In Table 3.1 the ratios of segments that cover more than one phone (multisegments) for the various over-segmentation factors are shown. The ratio of multisegments decreases from 5.16% to 0.30% when the over-segmentation factor is increased from 2 to 5. This means that there is a significant higher number of segments not being exploited in the ASWU creation when the over-segmentation factor is 2.

Table 3.1: Ratio of segments that cover more than one phone (Multisegments).

Over-segmentation Factor	Ratio of multisegments
2	5.16%
3	1.73%
4	0.69%
5	0.30%

In Table 3.2, the phone recognition results using different over-segmentation factors are shown. The system was trained using 256 clusters, each modelled by 8 component GMMs, and the GMMs in the final HMM states were reduced to a size of 32 (see Section 3.3 for description of these parameters and Section3.4.2 for an overview of the system performance for different settings). The results show that the over-segmentation factor should be

Table 3.2: Experiments with different over-segmentation factors. The experiments were performed with cluster size of 256 each with 8 component GMMs, and the final HMM GMM size was reduced to 32.

Over-segmentation Factor	Recognition Accuracy
2	61.6 %
3	64.9 %
4	66.2 %
5	66.1 %

set fairly high. With an over-segmentation factor of 5, one sentence had to be removed from the training set because the ratio of number of segments to number of frames was too high. The resulting acoustic segments would have had an average length less than 2 frames, which is the minimum segment length.

During the training procedure, the acoustic segments are compared to the manual labelling in order to give the acoustic segments a linguistic interpretation. It is therefore possible to get an overview of which phones the segments in a cluster belongs to. By counting how many segments that are assigned to each phone for a given cluster or ASWU, histograms like the ones in Figure 3.6 and Figure 3.7 can be constructed. These histograms give some characteristics of the ASWU. For the ASWU in Figure 3.6 most of the segments originate from only a few of the phones: the /k/ has by far the largest frequency of coinciding with the cluster (note that the scale on the y-axis is different for the two histograms in the figure), while /g/ and /cl/ (unvoiced closure) are the two next on the list. The /g/ is the voiced variant of the /k/, however in unvoiced context it might be pronounced unvoiced and thus identical to the /k/. Hence, all the three phones with highest frequency of co-occurring with this ASWU can be related to the /k/. This is a strong indication that this

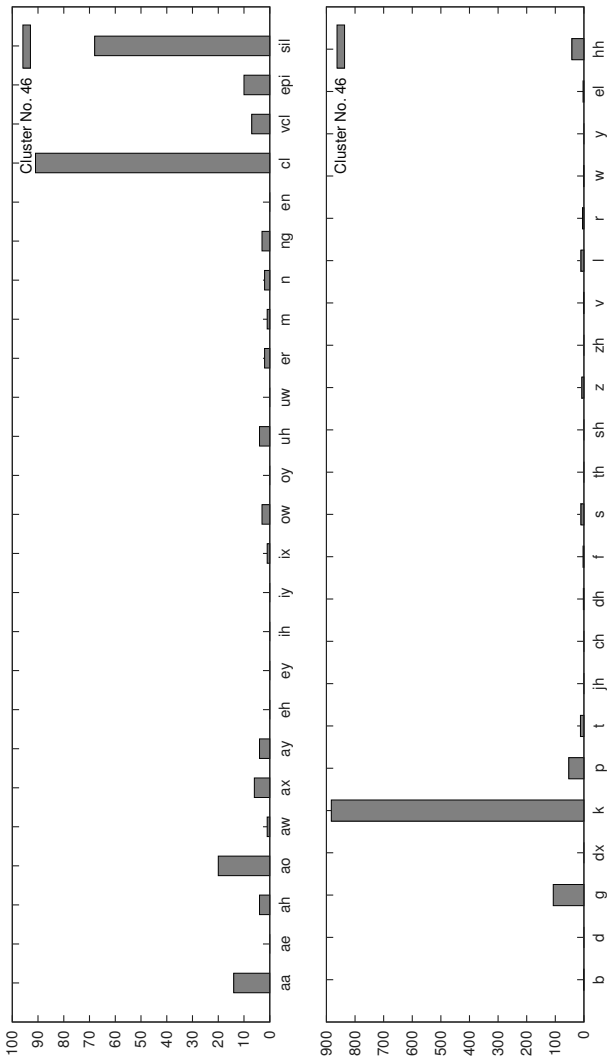


Figure 3.6: Example of phone assignment of ASWU

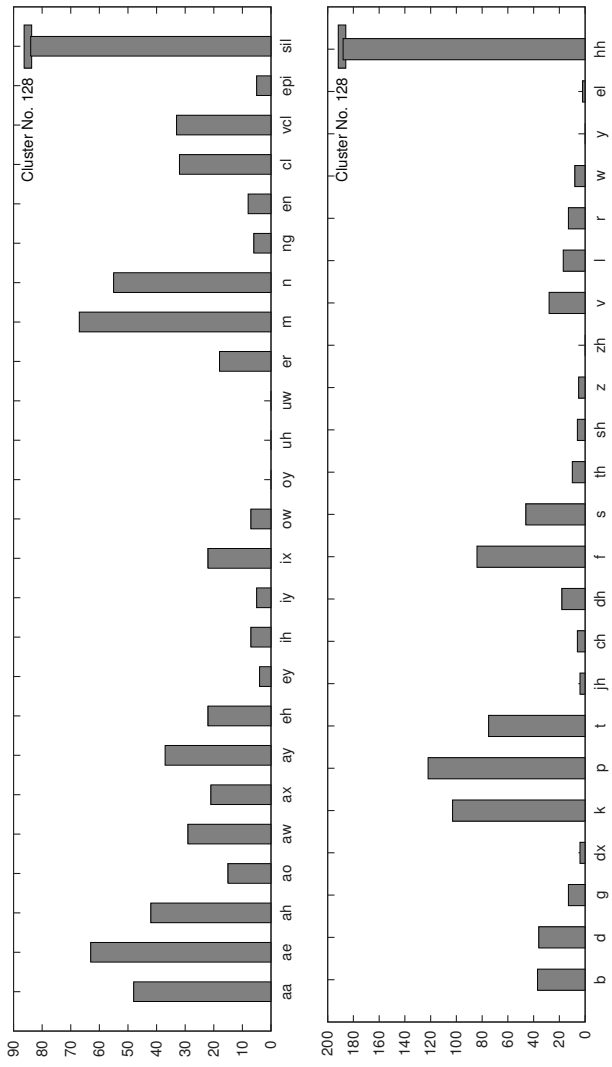


Figure 3.7: Example of phone assignment of ASWU

ASWU describes an acoustic event associated to the /k/. In Figure 3.7 the situation is a bit different: nearly all the phones contribute to the cluster. It is therefore reasonable to assume that the ASWU represents an acoustic event that has neutral characteristics and thus fits different types of sounds. Perhaps it often occurs between the phones. Nevertheless, since the ASWU describes parts of quite different phones, it is likely to have poor discriminative properties.

3.4.2 Varying Number of Clusters and GMM sizes

Table 3.3: ASWU HMM Results. Both the number of clusters and the cluster GMM size are varied. Alternative 1 is to use the GMMs from Equation (3.6) directly, while Alternative 2 employs Baum-Welch re-estimation to update the weights. For the third alternative also the final GMM size decided by the pruning was varied. All the experiments were conducted with an oversegmentation factor of 4.

				Alt. 3	
Number of	Size of			Size HMM GMM	
Clusters	Cluster GMM	Alt. 1	Alt. 2	16	32
Baseline HMM		- %	- %	65.0 %	66.7 %
128	8	23.3 %	31.6 %	64.9 %	66.3 %
128	16	23.2 %	31.6 %	64.6 %	65.9 %
128	32	22.6 %	32.0 %	64.8 %	66.0 %
256	8	17.8 %	22.9 %	64.7 %	66.2 %
256	16	17.6 %	23.2 %	64.7 %	66.0 %
256	32	17.5 %	23.6 %	64.6 %	65.6 %

In Table 3.3 the results of the experiments conducted with the three suggested alternatives are presented (see Section 3.3.2). For all the three alternatives the oversegmentation factor was set to 4 while both the number of clusters and size of cluster GMMs were varied. For the third alternative

also two different sizes of the GMMs resulting from the pruning step were tried.

The most obvious observation that can be made from the table is which of the three alternatives that is most successful: alternative 3. It is quite clear that allowing each of the states to update its GMM components individually in a Baum-Welch re-estimation procedure is superior in this case. Another observation is the relatively low variation in the performances for the alternative 3 systems. While varying the number of clusters and their GMM size for the alternative 1 and 2 systems yields significant changes in the performance, these parameters have small or no impact on the performance of the alternative 3 systems. Only the number of components in the GMMs after the pruning step seem to have influence on the results. This is expected as increasing the state GMM size from 16 to 32 improves the modelling capability of the state, something that can also be observed with the baseline HMM systems. The baseline results are achieved using standard 3-state HMMs trained with a conventional training scheme.

3.5 Discussion

In Section 3.4.1 different over-segmentation factors were tried for the segmentation algorithm. This factor controls how many segments the algorithm produces compared to the number of manual labels in the data. A reasonable hypothesis would be to adjust the algorithm to divide most of the phone realisations in about 2-3 segments, and have less realisations with 1, 4 or more segments. However, the experiments showed that the best value for the over-segmentation factor was about 4-5. This is a bit unexpected, but it indicates that the segments that cover more than one phone hurts the performance more than phones segmented into more than 4 segments. When considering how these two cases are handled by the training algorithm, it may seem as a more reasonable result: the segments

that cover more than one phone are not used further in the training, whereas more than 4 segments assigned to a phone results in the segments to be merged down to 4 segments. Hence, the first case results in unused training data (at least until the final Baum-Welch procedure), which is a loss that should be kept to a minimum.

In an effort to avoid the cases where an acoustic segment covers more than one phone, it could be tempting to consider segmenting each phone separately. This is possible to accomplish by using the manual labels and segment each phone. To get diversity in the number of segments to use for each phone, a distortion threshold should be used as stopping criterium. Unfortunately, this means that either the phones have approximately the same average number of segments, or this average needs to be decided individually for each phone. How to decide the average number of segments to use for a phone is not trivial, while having the same average number of segments is probably not optimal. As Figure 3.4 and Figure 3.5 (see Appendix B for full overview) show, the different phones naturally have quite different average number of segments assigned to them. Diphthongs like /ay/ and /oy/, for instance, have in general more segments assigned to them, compared to shorter phones like for instance /d/ (plosive).

The results of the phone recognition experiments show surprisingly small variations for the best alternative: alternative 3, where the state GMMs are pruned and re-estimated individually. The largest difference is less than 1 %, which makes it difficult to conclude which setting is the better. However, the results for alternative 1 and 2, where the Gaussian components were untouched after the construction of state GMMs according to Equation (3.6), show larger variations. In fact, the number of clusters seem to have a significant impact on the performance for these two alternatives. A difference which vanishes after the pruning and Baum-Welch re-estimation for the alternative 3 systems. It should be mentioned that there is another natural alternative to the three systems described in this chapter: instead of using a shared pool of Gaussian components

as in alternative 2, it is possible to employ a shared pool of GMMs. I.e. let c_{ij} and $\mathcal{N}(\underline{x}|\mu_{ij}, \Sigma_{ij})$ in Equation (3.6) on page 33 be shared among the states, and only allow the transition probabilities and the w_i^s weights to be updated in the Baum-Welch re-estimation. This will reduce the amount of parameters compared to alternative 2, but also reduce the flexibility. However, this alternative is expected to be close to alternative 2 in terms of performance, and considering the rather poor results achieved with alternative 2, this alternative was not implemented and tested.

A parameter that is influencing the result is the size of the GMMs in the final HMM states. Increasing from 16 to 32 components in the GMMs is consistently giving an increase in the performance of about 1-1.5%, which is about the same as for the baseline. Since the phones are modelled by 10 states, instead of 3 states like the baseline, the number of parameters is more than three times higher for the proposed system. In ASR the number of parameters is always a concern, as the size of the training data is limited. Fortunately, based on the presented results, the number of parameters does not seem to have a significant negative impact of the performance. Although, as mentioned earlier the number of parameters *might* have reached the limit for the setting with 256 clusters and 32 components in the GMMs.

A comparison between the proposed alternative 3 system and the baseline, shows that the two systems seem to perform at the same level: when using GMMs with 16 components both systems have an accuracy of about 65%, while for the case of 32 components in the GMMs, the baseline perform about 0.5% better. The baseline system thus performs marginally better than the proposed system in the experiments presented. In Appendix A.2 an approximation of confidence intervals for the TIMIT phone recognition experiments suggests that a 95% confidence interval for the PER has a size of about 0.7%, when the true PER is about 35% (i.e. an accuracy of about 65%). Hence, the difference is too small to be able to make any confident conclusions about which is the better system in terms of perfor-

mance, however the conventional HMM has a slight lead. Nevertheless, the number of components should also be considered. Each parameter needs a certain amount of training data to be sufficiently trained. A higher parameter count, thus means that a larger amount of training data is needed. In addition, the number of parameters is an indicator for the complexity of the system. Higher complexity leads to a slower system, which requires more resources for training and decoding. This cost needs to be justified with better performance. Unfortunately, this is not the case in the experiments reported in this chapter.

3.6 Concluding Summary

In this chapter an alternative to the conventional topology of the phone HMMs has been proposed. Instead of using 3-state in a left-to-right manner, the proposed system used several parallel left-to-right paths, each with a different number of states ranging from 1 to 4. The intention was to give the model more accurate modelling of the different acoustic events occurring within the phones. A special training algorithm utilising an automatic acoustic segmentation algorithm was designed to accommodate the increased number of states and parameters. The acoustic segmentation algorithm finds the segmentation that minimises the intra-segment variation, leading to a segmentation where each segment may be approximated by the centroid. A clustering procedure is then performed by the use of the segment centroids, yielding a set of ASWUs. By comparing the acoustic segments with a manual phonetic labelling of the data, links between the HMM states and the acoustic segments were created. These links were used to construct each state GMM by a linear combination of ASWUs, before a final embedded Baum-Welch re-estimation procedure was performed. The proposed system provides recognition performance at the same level as the baseline 3-state HMM system. However, the signifi-

cant increase of parameters and complexity makes the conventional HMM system seem like a better choice.

Chapter 4

Construction of HMM by Pronunciation Variation Modelling of ASWUs

In the previous chapter a phone recognition HMM system with an expanded topology compared to the standard 3 state HMM was proposed. Acoustic Subword Units (ASWUs) were derived in the training scheme which was needed to estimate the parameters of the enlarged system properly. In this chapter the expanded HMM topology will be revisited, but with a significant difference: now, the topology of the phone HMMs will be derived individually in a data driven manner, allowing the phone HMMs to be custom made for each phone automatically. The ASWUs defined in the previous chapter will be employed by a Pronunciation Variation Modelling (PVM) algorithm for this purpose. PVM is normally used in ASR to allow more than one pronunciation of words. In this chapter PVM will be operated on the phone level to produce pronunciation variations of phones.

The Acoustic Subword Units (ASWUs) were defined in Section 3.2 (see page 24) in the previous chapter. In this chapter an overview of the new HMM system will be given first, before Pronunciation Variation Modelling is discussed in Section 4.2. In Section 4.3 the final re-estimation procedure of the HMM system will be described. Finally, the experimental results will be presented in Section 4.4, followed by a discussion in Section 4.5 and a concluding summary in Section 4.6.

4.1 System Overview

In this chapter an HMM based phone recognition system, where the topology is derived in a data-driven manner, will be presented. The HMM system resembles the system presented in the previous chapter where the topology was expanded to consist of several parallel left-to-right HMM paths with different number of states. However, instead of using a fixed topology for all the phones, the number of paths and the number of states in each path will be automatically derived. Customising the topology of each phone allows the system to adjust the number of paths according to the variation seen in the training data of the phone in question. Also the number of states in each path will be adapted to better match the durations of the realisations in the training set. This is an advantage as some phones, like diphthongs, often have longer duration than other phones, like for instance stops. Thus such phones are probably better modelled using a higher number of states.

In Figure 4.1 an example of the proposed phone HMM is shown. The example shows a topology of four parallel paths with a varying number of states. Both the number of paths and the number of states are derived automatically and may therefore vary between phones. In the previous chapter a set of ASWUs were defined and used to train phone HMMs with similar topology. A linguistic interpretation of the ASWUs was made in

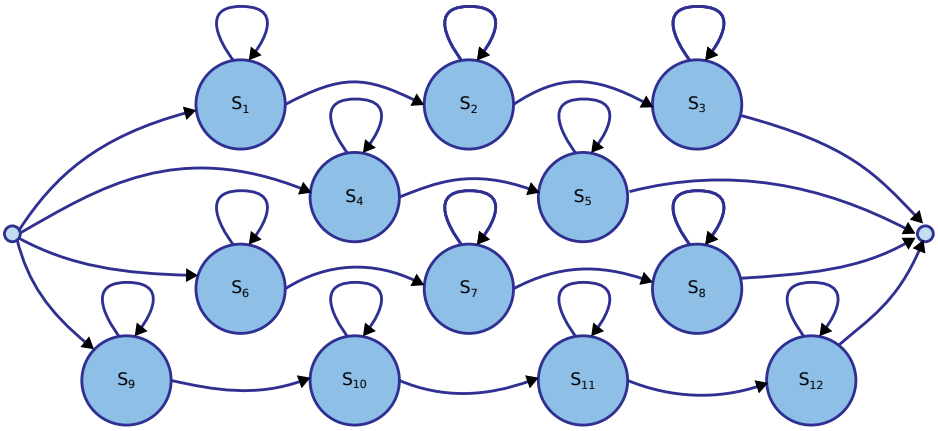


Figure 4.1: An example of the phone HMM topology.

order to construct the emission densities of the states as a linear combination of the ASWU GMMs. In this chapter the approach will be quite different. Instead of using linguistic knowledge, the ASWU GMMs will be used by a Pronunciation Variation Modelling (PVM) algorithm to find the optimal sequences of ASWUs for each phone. Each phone realisation is assumed to consist of one or more acoustic events described by the ASWUs. The PVM algorithm uses the training data to optimise the selection of such ASWU sequences to use for each phone. Finally, the result is a mapping between each state in the phone HMM and one ASWU. This is different from the system described in the previous chapter where each state GMM was defined as a linear combination of all the ASWU GMMs.

The next section will introduce PVM and describe the algorithm used, along with a description of modifications made to adjust the algorithm for phone level PVM.

4.2 Pronunciation Variation Modelling (PVM)

In speech recognition, modelling of pronunciation variations is performed to handle the differences in the pronunciation of words. Pronunciation variation modelling (PVM) is usually performed by including several pronunciation alternatives in the lexical dictionary. Each pronunciation alternative consists of a possible sequence of units, referred to as a *baseform*, to make up a word. Since the pronunciation of words can vary dependent on various factors like speaking style [48], degree of formality [49], environment [50], speech disability, accent or dialect [49], and emotional status [51], PVM is important in many ASR tasks. The amount of literature on the topic is large, however overviews may be found in for instance [52, 53, 54, 55, 56]. In [52] Strik and Cucchiarini divide the different approaches based on the source of information that the modelling is based on: in *knowledge based* pronunciation modelling it is assumed that the information on pronunciation variation is known in the literature, see e.g. [57, 58, 59]. Whereas in *data driven* pronunciation modelling this information is obtained from the data, see e.g. [60, 61, 62, 63, 64, 65, 66, 67, 68]. In this work PVM is used on acoustic units without a linguistic connection, and knowledge about the pronunciation of phones in terms of these acoustic units is not available. Hence, knowledge based PVM is not possible.

PVM is usually performed in two steps: baseform *generation* and baseform *selection*. In [53] the approaches for baseform generation is divided into several categories: rules [69, 64], artificial neural networks [70], grapheme-to-phoneme converters [71], phone recognisers [72] and decision trees [73, 74].

In [75] and [76] it was shown that when the average number of pronunciation variants per word exceeds about 2.5, the system performs worse than a system with only one variant per word. Hence, the selection process of

the candidate baseforms needs to be done carefully. The literature proposes several criteria for selecting which baseforms to use: frequency of occurrence of the variants [76], confidence measures [72], the degree of confusability between the variants [72, 62] and an ML criterion [77, 78, 79, 56].

In this dissertation, the data driven Maximum Likelihood approach presented in [77, 78, 79, 56] was chosen. The baseform candidates are found by performing an N-best decoding of a set of realisations, or *tokens* of each word. The ML approach, which will be described in Section 4.2.1, then performs the baseform selection in a two stage procedure. In the first stage, which is performed individually on each word, the tokens of the word are clustered by the use of a special iterative K-Means based scheme, where each cluster is represented by one of the candidate baseforms. Thus, each cluster represents one pronunciation variation. At each iteration of the K-Means clustering approach the optimal clusters are stored for use in the second stage which finds the baseform candidates that maximises the joint likelihood of all the words given an average number of baseforms per word.

The approach is completely data driven, which is a prerequisite for modelling the phone level pronunciation variations with the use of the ASWUs. Another advantage is the joint optimisation, which leads to a varying number of pronunciation variants for the different phones. Thus, phones with more variation in the realisations may be modelled with more pronunciation variants than phones with less variation. However, to apply the approach on the phone PVM task, some modifications had to be performed. These modifications will be discussed in Section 4.2.2, but first, the original approach from [77] will be presented.

4.2.1 The Pronunciation Variation Modelling Algorithm

The pronunciation variation modelling algorithm was used in [77] for word level PVM. It employs an ML criterion in order to find the optimal lexicon for all the words $\mathcal{W}$ in the training data. For each word w a set of K randomly drawn realisations or tokens are used to find the optimal baseforms. The number of baseforms J_w for the word w is found under the constraint that the total number of baseforms is J_T. For each word, a list of candidate baseforms is created by conducting an N-best Viterbi search on the K tokens. For each word w with baseform candidates $\{B_{w,j}|w \in \mathcal{W}, j \in [1 \dots J_w]\}$ the likelihood of the K tokens is calculated as:

$$L_w = \prod_{k=1}^{K} \max_{j \in [1 \dots J_w]} p(\bar{\mathbf{x}}_{w,k}|B_{w,j}, \theta), \tag{4.1}$$

where $\bar{\mathbf{x}}_{w,k}$ are the feature vectors for the kth token of the word w, and θ is the set of HMM parameters (which are kept constant for all tokens and words). The PVM searches for the baseform candidates that optimise the total likelihood L:

$$\begin{aligned} L &= \prod_{w \in \mathcal{W}} L_w \\ &= \prod_{w \in \mathcal{W}} \prod_{k=1}^{K} \max_{j \in [1 \dots J_w]} p(\bar{\mathbf{x}}_{w,k}|B_{w,j}, \theta) \end{aligned} \tag{4.2}$$

under the constraint

$$\sum_{w \in \mathcal{W}} J_w = J_t \leq J_T. \tag{4.3}$$

The process of finding the ML lexicon is divided into two stages: in Stage 1 the algorithm handles each word individually and finds the optimal

baseforms for different values of J_w, which is limited by the parameter $J_{\max}$. Stage 2 uses these results to find the optimal set of pronunciation variants which will comprise the lexicon.

The search for the optimal baseforms for each word can be viewed as a clustering problem, where the tokens of the word are clustered and each cluster is represented by a baseform. This is conducted by a clustering approach based on the K-Means algorithm. The clustering approach is *divisive*, i.e. it initialises with one large cluster and iteratively divides the clusters. Since each cluster is represented by the optimal baseform describing the cluster, the result in each iteration is the optimal baseforms for a given number of baseforms. In each iteration, the cluster with the lowest likelihood is located and divided into two clusters. The division is performed by comparing the baseform candidates in each cluster. Each baseform is a sequence of one or more units, and thus *Levenshtein distance* is possible to use as a measure for distance between two baseform candidates. The Levenshtein distance is a string metric for measuring the distance between two strings. It is the minimum number of insertions (I), deletions (D) and substitutions (S) needed to convert one string into the other. Stage 1 can be summarised as follows:

1. Set the number of clusters $J_w = 1$ and assign all the K tokens of word w to the initial cluster $\mathcal{K}_1$.

2. Find the ML-estimate that describes the initial cluster:

$$\hat{B}_w^{\mathcal{K}_1}(J_w = 1) = \underset{B_{w,i}\,\forall i}{\operatorname{argmax}} \prod_{k \in \mathcal{K}_1} p(\bar{\mathbf{x}}_{w,k} | B_{w,i}, \theta) \tag{4.4}$$

$$l_w^{\mathcal{K}_1}(J_w) = \prod_{k \in \mathcal{K}_1} p(\bar{\mathbf{x}}_{w,k} | \hat{B}_w^{\mathcal{K}_1}(J_w = 1), \theta) \tag{4.5}$$

$$\tag{4.6}$$

3. Find the cluster $\mathcal{K}_i$ with the lowest total likelihood $l_w^{\mathcal{K}_i}(J_w)$, and within the cluster identify the two baseforms spaced furthest apart

with regard to string distance. Let the corresponding baseforms be the initial baseforms for cluster $\mathcal{K}_i$ and a new cluster $\mathcal{K}_{J_w+1}$.

4. Increase the number of clusters $J_w = J_w + 1$.
5. For each token k: re-assign the token to the cluster $\mathcal{K}'_j$ which maximises the likelihood:

$$\mathcal{K}'_j = \underset{\mathcal{K}_j \in \{\mathcal{K}_1 \ldots \mathcal{K}_{J_w}\}}{\operatorname{argmax}} p(\bar{\mathbf{x}}_{w,k} | \hat{B}_w^{\mathcal{K}_j}(J_w)) \tag{4.7}$$

6. Find the new baseforms to represent each cluster:

$$\hat{B}_w^{\mathcal{K}_j}(J_w) = \underset{B_{w,i}\,\forall i}{\operatorname{argmax}} \prod_{k \in \mathcal{K}_j} p(\bar{\mathbf{x}}_{w,k} | B_{w,i}, \theta) \quad j \in [1 \ldots J_w] \tag{4.8}$$

7. Calculate the data likelihood for each cluster and the total likelihood:

$$l_w^{\mathcal{K}_j}(J_w) = \prod_{k \in \mathcal{K}_j} p(\bar{\mathbf{x}}_{w,k} | \hat{B}_w^{\mathcal{K}_j}(J_w), \theta) \tag{4.9}$$

$$l_w(J_w) = \prod_{j=1}^{J_w} l_w^{\mathcal{K}_j}(J_w) \tag{4.10}$$

8. If the token assignment has changed, go to step 5.
9. If $J_w < J_{\max}$, increment J_w and go to step 3.

Stage 1 results in a list of optimal baseforms for each value of $J_w \in [1 \ldots J_{\max}]$, and the corresponding likelihoods, for all the words. In Stage 2 this list is used to find the number of baseforms to use for each word by maximising the joint likelihood of the entire training set, while the total number of baseforms for all words combined is fixed to the limit J_T. The optimisation is performed in an iterative manner, where each word is initiated with 1 baseform, and for each step one baseform is added to the word yielding the maximum likelihood gain. Thus, Stage 2:

1. Initialise $J_w = 1$ for all words w, and set the counter for the total number of baseforms equal to the number of words: $J_t = W$.
2. Find the word that yields the highest gain in likelihood when increasing the number of baseforms by one:

$$w' = \underset{w \in [1...W]}{\operatorname{argmax}} \frac{l_w(J_w + 1)}{l_w(J_w)} \tag{4.11}$$

3. Increment $J_{w'}$ and J_t.
4. If $J_t < J_T$ go to step 2.

Stage 2 results in the desired lexicon with the given average number of baseforms per word.

4.2.2 Modifications to the PVM algorithm

The algorithm described in the previous Section was mainly used for word modeling using phonemic baseforms. In this dissertation, it is desired to use the approach on phone level modeling using ASWUs. Although the algorithm was successfully applied on ASWUs for constructing the word level lexicon with the best baseform for each word in [43], some modifications are necessary for the PVM algorithm to be suitable for phone level modelling. The phones are shorter than the words, leading to short baseforms. They are also more dependent on the context, which means that a higher number of tokens should be used in order to cover the variation in pronunciation. The baseform candidates are found by performing N-Best Viterbi decoding on all the K tokens. When K is high, this leads to a significant number of candidate baseforms. Processing all the candidates is time consuming and requires a lot of memory. To remedy this, the candidate baseform list was pruned by sorting on frequency of occurrence and likelihood, and keeping the most promising baseforms.

Another problem is related to the duration of the phones. Since the duration is generally quite short, the length of the baseforms is about 2-4 units. In Stage 1, the string distance is used as a distance measure between two baseforms. Because of the short baseform length, the string distance with unit penalty for insertion, deletion and substitution results in small distances with low variation. A way of improving the measure is to use a penalty dependent on the acoustical distance between the ASWUs. Remember that all the ASWUs are derived using clusters of acoustic segments, which makes it possible to assign a centroid to represent the acoustic properties of each ASWU. Hence, in this work the Euclidean distance between the centroids has been chosen as the cost of substituting one ASWU with another. Using such a penalty is better, because knowledge about the acoustic space is introduced. Substituting an ASWU with a nearby ASWU, will yield a low distance compared to substituting with an ASWU further apart in the acoustic space.

The insertion and deletion costs need to be comparable to the substitution cost. One possible cost that was tried is the average substitution cost of the ASWU to be inserted or deleted. I.e., the average distance to all the other ASWU centroids. A problem that appeared, was that ASWUs far away have a large influence on this cost. Experiments showed that using a nearest neighbour approach with only the 50% nearest ASWUs was a better measure and was therefore used in the experiments presented in this dissertation.

The PVM approach does not take into account that the same baseform might be a candidate for more than one word. This probably becomes a larger problem when modelling pronunciation variations for phones, since the baseforms are generally shorter. When a baseform candidate has been suggested as a candidate for more than one word, that baseform is likely to cause confusion if used by either of the words. Experiments have also shown that better performance is achieved by simply removing such baseforms from the candidate list before presenting them to the PVM algo-

rithm. There might be instances where this approach removes a good candidate from a phone because it is a (poor) candidate for another phone, a situation that might affect the performance. This can be avoided by finding a threshold for when to remove the candidate. However, such an approach introduces yet another parameter which needs to be adjusted. Hence, the solution of simply removing such baseforms was used in the experiments presented in this dissertation.

4.3 Final Re-estimation

After the PVM algorithm has found the optimal baseforms to use for each phone, the phone HMMs can be constructed by using the ASWU GMMs as states. By performing forced alignment on the training data using the manual labels and the full HMM system, a labelling on the state level can be produced. These state level labels contain the optimal path to use for each phone realisation, which makes it possible to run the Baum-Welch algorithm to re-estimate the HMM parameters. The two steps can be repeated several times for improved estimates.

4.4 Experiments

In this section the experiments performed with the proposed system will be presented. Again, the TIMIT database described in Appendix A was utilised. The ASWUs were trained in the same way as in the previous chapter, where the acoustic segmentation was performed with 13 static MFCC features (including C_0) extracted with a 15 ms window with a 5 ms shift. For the training of the ASWU GMMs, 13 MFCCs (including C_0) were extracted using a 25 ms window with a 10 ms shift, and in addition, the dynamic features ($\Delta + \Delta\Delta$) were appended. Experiments were run

with a varying number of ASWUs and components in the GMMs. The number of ASWUs were varied from 128 to 512, while both 16 and 32 GMM components were tested. All the training was performed with the 48 phone set, which was mapped to the 39 phone set during testing (see Appendix A).

For the PVM algorithm both the average and the maximum number of baseforms per phone can be adjusted. For the word level PVM task, the literature suggests to have the number of pronunciation variants per word less than 2.5 [75, 76], to avoid the confusability to become large. A similar effect is probably true for the case of phone level PVM, and thus the number of baseforms should be kept low. In the results presented in Table 4.1 the average number of baseforms were varied between 1.0 and 2.0, while the maximum number of baseforms were set to 3 (obviously, for experiments where the average number were 1.0 or 1.25 the maximum number of baseforms seen were 1 and 2, respectively). These results also indicate that the average number of baseforms needs to be low. However, that the best performance seems to be achieved with only one baseform per phone, seems odd. This intuitively contradicts to the observation that context dependent models improve the performance. In Table 4.2 the final re-estimation step, which is performed on the system proposed by the PVM algorithm, is skipped. I.e the ASWU GMMs are used directly without any further re-estimation, and thus some of the GMMs are shared. The table shows the performance for the system created with 512 clusters and 32 components in each ASWU GMM. First of all: it is clear that the re-estimation procedure has big impact on the performance, which is quite weak for this system. Secondly: the performance now has a positive trend when the number of baseforms is increased.

The baseline performance, which was also presented in the previous chapter, is given in Table 4.3. These results were derived using a conventional 3 state HMM with Gaussian mixtures. In Table 4.4 the number of states used by the final HMM system are presented. This shows that the best

Table 4.1: Phone recognition results (accuracy) on the TIMIT test set. The average number of baseforms per phone varies from 1.0 to 2.0, while the maximum number of baseforms is kept at 3.

Number of ASWUs	GMM Size	Average Number of Baseforms 1.0	1.25	1.5	1.75	2.0
128	16	64.3 %	63.2 %	62.7 %	62.9 %	62.6 %
128	32	65.1 %	64.7 %	64.4 %	64.5 %	64.0 %
256	16	63.6 %	62.9 %	62.8 %	63.1%	62.8 %
256	32	64.8 %	64.4 %	64.5 %	64.2 %	63.9 %
512	16	63.1 %	62.7 %	62.3 %	62.6 %	62.4 %
512	32	65.2 %	64.8 %	64.2 %	64.1 %	64.0 %

Table 4.2: Phone recognition accuracy when the final re-estimation step is skipped.

Number of ASWUs	GMM Size	Average Number of Baseforms 1.0	1.25	1.5	1.75	2.0
512	32	26.7 %	27.3 %	29.1 %	28.7 %	31.4 %

Table 4.3: Baseline phone recognition results on the TIMIT test set.

GMM Size	Accuracy
16	65.0 %
32	66.7 %

Table 4.4: Statistics for number of states used per phone.

Number of ASWUs	GMM Size	Average Number of Baseforms 1.0	1.25	1.5	1.75	2.0
128	16	2.7	3.3	3.8	4.4	4.9
128	32	2.6	3.1	3.7	4.3	4.9
256	16	2.6	3.4	3.9	4.5	5.0
256	32	2.5	3.3	3.8	4.5	5.0
512	16	2.6	3.4	3.8	4.4	4.9
512	32	2.7	3.4	3.8	4.4	5.1

Table 4.5: Average number of states per baseform for the different phone classes for the experiment with 512 ASWUs, GMM size of 32 and 1 baseform per phone.

Class	Phones	Min	Max	Avg
Vowels	aa ae ah ax eh er ih ix iy uh	2	4	2.7
Diphthongs	ao aw ay ey ow oy	2	5	3.7
Stops	b d dx g k p t	2	3	2.3
Closure	cl vcl	3	4	3.5
Affricative	ch jh	3	3	3.0
Fricative	dh f s sh th v z zh	2	4	2.8
Semivowel/Glide	el hh l r w y	2	3	2.2
Nasal	en m n ng	2	2	2.0
Other	epi sil	2	3	2.5

systems have about 2.7 states per phone, which is slightly less than the baseline. Finally, Table 4.5 present some statistics of the number of states per baseform for the setup with highest accuracy: the system with 512 ASWUs, 32 GMM components and 1.0 baseform per phone. The table shows that the different classes indeed have a variation in the baseform length. Diphthongs are phones which often is longer in duration and have large acoustic variations. In the table it is apparent that these require more states (average of 3.7 states) than phones with relatively short durations like for instance stops (average of 2.3 states).

4.5 Discussion

In the previous section the proposed system was tested on the TIMIT phone recognition task. Both the number of ASWUs, GMM size and the average number of baseforms were varied. There are several noteworthy tendencies in the performance results presented in Table 4.1. The most obvious is probably the impact of the average number of pronunciation variants for the phones: for all the settings the best performance was achieved with only one variant for every phone. Isolated, these results might suggest that adding pronunciation variation to the phones increases the confusion more than it benefits the modelling of the variations in the realisations of the phones. However, the results presented in Table 4.2 shows the opposite behaviour with the best performance with an average number of baseforms of 2.0, which was the highest number in these experiments. This suggests that the PVM algorithm works as intended and yields better systems when the average number of baseforms per phone is higher than one. Unfortunately, this advantage is not retained through the Baum Welch Re-estimation procedure, which according to Table 4.2 has a huge impact on the phone recognition performance.

One reason for the reduction in accuracy for systems with a higher number

of baseforms, might be the increase in the average number of states per phone shown in Table 4.4. Because the final re-estimation step removes the sharing of parameters between the states, a higher number of states leads to a higher number of parameters to train, which again requires more training data to be estimated properly. As the table shows the total number of states becomes almost twice as high when doubling the average number of baseforms. This is a significant increase in parameters, which might lead to the parameters in the system not being trained properly. It is also interesting that the best performance is achieved with approximately 3 states per phone, which is the same as the baseline.

Another property worth mentioning is the impact of the number of ASWUs. Although the differences are small, it seems there is an advantage with a low number of ASWUs when using a GMM size of 16. Whereas for the case of 32 Gaussian components the number of ASWUs seems to have no impact on the performance of the phone recogniser.

A somewhat strange behaviour is found by observing the tendency of the performance when increasing the number of baseforms for the case of 16 Gaussian components. The general trend is a lower performance when the number of baseforms is increased. However, for this case there is a small local maximum in the accuracy when the average number of baseforms is 1.75. This is consistent for all the three chosen values for the number of ASWUs. The increase in performance at this point is probably too small to be of any statistical significance, and thus might be a coincidence. However, the peculiar behaviour is present for three different settings, and should therefore be noted.

Compared to the baseline system, the performance of the proposed system is slightly worse, especially for the case of 32 Gaussian components where the difference is 1.5%. Although the number of parameters used in the best variants of the proposed system is lower than the baseline system, the difference is probably too small to have any impact when the training

data is sparse.

4.6 Concluding Summary

In this chapter the topology of the phone HMMs were adapted automatically by using a PVM algorithm on the ASWUs defined in the previous chapter. The phone realisations were modelled as a sequence of ASWUs, or baseforms, and several baseforms were found for each phone by N-Best Viterbi decoding. The PVM algorithm then employed an ML criterion to jointly optimise which and how many baseforms to use for each phone model. The result was an HMM topology similar to the one in the previous chapter, but with the main difference that the number and lengths of the parallel paths were automatically derived in a data driven manner. The intention was to allow the HMM to model phones with high variation in their realisations with more paths. In addition phones with generally more acoustic events in their realisations could be modelled using longer baseforms. Phone recognition experiments were carried out on the TIMIT database, showing several interesting properties of the proposed system. Unfortunately, the performance of the proposed system was slightly lower than the baseline system. In addition the best performance was achieved when only one single pronunciation variant was used for all the phones. Hence, based on the experiments presented in this chapter and the previous, it might seem that using several paths in the HMM topology does not improve the systems modelling capabilities. In any case the baseline system seems like a simpler and more attractive choice.

Chapter 5

The Non-Negative Durational HMM

In the two previous chapters the HMM systems have been based upon a set of ASWUs, defined in a data driven manner without the use of phonetic knowledge. In this chapter an HMM system based on another algorithm able to extract units in a data driven manner will be investigated. The system presented in this chapter combines Non-negative Matrix Factorisation (NMF) with the HMM framework. NMF is similar to Principal Component Analysis (PCA) and Singular Value Decomposition (SVD), a matrix factorisation technique, but it requires the input matrix to be non-negative. The result is an approximation of each column in the input matrix by a linear combination of latent units. These latent units are recurrent patterns and are extracted without any knowledge not contained in the matrix. However, they often turn out to have some sort of meaning, like representing a nose or an eye [80] when NMF is performed on images. Combining this technique with the HMM allows the modelling of the non-stationarity of the speech signal by means of the Markov chain, while the

NMF provides for discovery and modelling of the latent units. This was done in [81, 82, 83] which resulted in the non-negative HMM (N-HMM). Originally this model was used to approximate a specific input matrix. However in order for it to be useful for speech recognition it needs to generalise to unseen data. In this chapter a modified version of the N-HMM, the NdHMM, is presented. Unlike the N-HMM, the NdHMM can be used for recognition on unseen data.

5.1 NMF and the N-HMM

Non-negative matrix factorisation (NMF) has been shown to be useful in various disciplines in the recent years. One of the key properties of NMF is the ability to extract latent components from data, and represent the non-negative matrix as a linear combination of the latent components. In NMF a non-negative matrix $\mathbf{V}$ is approximated by two non-negative matrices $\mathbf{W}$ and $\mathbf{H}$:

$$\underset{N\times M}{\mathbf{V}} \approx \underset{N\times R}{\mathbf{W}} \cdot \underset{R\times M}{\mathbf{H}}, \qquad R \ll N, M. \tag{5.1}$$

Since the matrices are non-negative, the factorisation represents the columns of $\mathbf{V}$ as an additive linear combination of the latent components, where the latent components are in the columns of $\mathbf{H}$ and the linear combination weights are in the rows of $\mathbf{W}$.

Although the above decomposition is inexact, the reduced rank approximation has been shown to be useful in many applications. In [84], NMF has been successfully used to discover phone patterns by representing each utterance in the database using weighted phone lattice transition probabilities in the columns of $\mathbf{V}$. Further, in [85] convolutional NMF (cNMF) [86] was used on the spectrogram of the utterances to discover phone structures. The spectrogram is represented as a matrix where column dimension reflects frequency and row dimension reflects time. Unlike NMF

which restricts the latent units to describe one column in the spectrogram, the cNMF utilises a fixed temporal duration of the latent units. This allows the factorisation to discover latent units with a duration, like the phones, which is exactly what is done in[85].

Probabilistic extensions for NMF allow the use of sophisticated statistical techniques while still using the general ideas of NMF. In [87], a probabilistic extension of NMF was presented for modelling sound spectrograms. The probabilistic extension models the columns of a spectrogram $\mathbf{V}$ as histograms filled by "sound quanta". The amount of sound quanta in a given time-frequency bin is indicated by the Fourier magnitude of that bin. Once normalised, the spectrogram can be considered as a joint probability distribution $P_t(f)$ over time and frequency and is represented in the NMF framework as follows:

$$P_t(f) = \sum_z P(f|z)P_t(z). \tag{5.2}$$

The above equation states the distribution which is used to generate a quantised version of the spectrogram by performing multiple random draws. Each draw results in adding a sound energy quantum to the corresponding time-frequency bin. The distribution, $P_t(f)$, is defined as a linear combination of a set of time independent dictionary components ($P(f|z)$) weighted with time dependent weights ($P_t(z)$), and is represented using multinomial distributions[1]. Thus, each time frame of the spectrogram is generated by performing multiple draws. Each draw includes two steps: first a dictionary component is chosen according to $P_t(z)$, before the frequency is drawn according to $P(f|z)$. The draws are repeated until

[1]A multinomial distribution is the distribution of the outcome when N draws are made from a categorical distribution. When the number of outcomes of the categorical distribution is 2, these become the binomial and Bernoulli distributions. Since the categorical distribution is a multinomial distribution with one draw, "multinomial" will in this dissertation refer to both the multinomial and the categorical distribution.

the frame is filled with the total number of observed sound quanta. The mixture weights of the model therefore capture the temporal variation in the input signal. The formulation in Equation (5.2) is also referred to as probabilistic latent semantic analysis (pLSA) in the literature [88].

A major limitation in using the above formulation for modelling speech is that a single set of dictionary components, $P(f|z)$, are derived to represent the entire spectrogram. This limits the expressive power of the model as the speech spectrum is non-stationary. In [81, 82] and [83] it has been shown that HMMs can be combined with NMF to incorporate the non-stationary component as a Markov model and thereby changing the dictionary components with time. This model is called non-negative HMM (N-HMM) and employs the probabilistic NMF to model the emission densities of the states. Each state q of the N-HMM has a fixed set of dictionary components $P(f|z,q)$ with time varying weights $P_t(z|q)$. Thus an N-HMM is able to describe different parts of the input signal with different states. For a speech signal the states may correspond to different phones. In fact, in[81], the model was applied to a speech signal represented by the spectrogram, and the dictionary components were demonstrated to contain phone-like structures.

Although the N-HMM has been reported to be successful for separating mixture signals, such as music or speech and noise in [81], it is not suited for modelling components of a speech recognition system (ASR); this is because the weights $P_t(z|q)$ are dependent on the absolute time t of the utterance. The dependence on the weights $P_t(z|q)$ means that the model does not generalise to unseen data. In this chapter, a set of modifications to the N-HMM is proposed, in order for it to be viable for ASR. The main change to the N-HMM is to make the weights dependent on the state occupancy duration, i.e. the number of frames since the process entered the state. Thus the modified model is referred to as non-negative durational HMM (NdHMM).

5.2 Detailed Description

The N-HMM combines NMF and HMM, thereby allowing the speech signal to be expressed with time varying dictionary components. The N-HMM described in [81] is a hidden Markov model which models a matrix with the use of the probabilistic NMF. It was designed for use on a spectrogram represented as a matrix where column dimensions reflects frequency and row dimensions reflects time. Each column represents a time frame and is modelled by the multinomial probability distribution

$$P_t(f|q) = \sum_z P(f|z,q)P_t(z|q), \tag{5.3}$$

where f is the frequency dimension, q is the state, $P(f|z,q)$ is the dictionary components and $P_t(z|q)$ are the weights. This is similar to the model in Equation (5.2), except from the dependency on the current state q. Thus, each time frame can be generated by performing multiple draws from the distribution $P_t(f|q)$, where each draw results in adding a sound energy quantum to the corresponding time-frequency bin. The number of draws v_t is explicitly modelled using a Gaussian[2] distribution $P(v_t|q_t)$, also called the energy distribution of the state. This is because the number of draws intuitively corresponds to the energy of the spectrogram at that time frame. Each state contains a dictionary of probability distributions $P(f|z,q)$ which reflects a variation of the sound (e.g. phone) that is modelled by the state. The degree of which a given dictionary component $P(f|z,q)$ contributes to the time frame at time t is controlled by the time dependent weights $P_t(z|q)$.

In addition to the hidden state sequence $\bar{\mathbf{q}}$, the sequence of dictionary components $\bar{\mathbf{z}}$ is also hidden. Thus, the likelihood for the model to produce

[2]The use of a continuous distribution for the number of draws, which is discrete, is inherited from the original N-HMM formulation.

Algorithm 5.1 Generative process of N-HMM

Draw q_1 from $P(q_1)$
for $t = 1 \rightarrow T$ **do**
  Draw v_t from $P(v_t|q_t)$
  for $v = 1 \rightarrow v_t$ **do**
    Draw $z_{t,v}$ from $P_t(z|q_t)$
    Draw f from $P(f|q_t, z_{t,v})$
    $\underline{x}_t[f] = \underline{x}_t[f] + 1$ ▷ Add the sound quantum to the frequency bin
  end for
  Draw q_{t+1} from $P(q_{t+1}|q_t)$
end for

an observation sequence $\bar{\mathbf{x}}$ is given by:

$$P(\bar{\mathbf{x}}) = \sum_{\bar{\mathbf{q}}} \sum_{\bar{\mathbf{z}}} P(q_1) \left(\prod_{t=1}^{T-1} P(q_{t+1}|q_t) \right) \left(\prod_{t=1}^{T} P(v_t|q_t) \right) \left(\prod_{t=1}^{T} \prod_{v=1}^{v_t} P_t(z_{t,v}|q_t) P(f_{t,v}|v_t, z_{t,v}, q_t) \right), \tag{5.4}$$

where T is the number of observation vectors and $f_{t,v}$ is the vth frequency bin drawn to produce $\underline{x}_t$. The generative process for creating the observation sequence by filling each vector with sound quanta is shown in Algorithm 5.1. It is important to note that the weights $P_t(z|q)$ are time dependent and vary for each time frame. The variation of weights for each time frame captures the temporal variations in the input signal though the dictionary components are time invariant, only conditioned on the state. Hence, the weights trained on a particular utterance cannot be used for another one, which makes it impossible to decode an unknown utterance using Viterbi search. This greatly reduces the generalisation capacity of the model to unseen data. For the intended use of the N-HMM as a representation of the input matrix, where the capabilities of uncovering recurrent events were the focus, this is not a problem. However, for the

task of speech recognition, generalisation is important.

The time dependency of the weights cannot be removed as they capture the temporal dynamics. Further, having constant weights for all time frames, will collapse the multinomial mixture models to a single multinomial for each state and will result in a poor model. In order to overcome the above problem, a model where the weights are dependent on the state occupancy duration is proposed. I.e. the weights of the model are dependent on the number of frames encountered since the process entered the state. Thus, the same set of weights are used every time the state is visited. By denoting the duration of the current state q_t as d_t, the weight distribution for the state is changed to $P(z_t|q_t, d_t)$. This modification does not introduce any duration modelling, but the distribution of the latent variable z_t is now dependent on both the state and the contiguous occupancy of the state at time t. The number of self transitions is still only dependent on the transition probabilities as before and d_t is simply keeping track of how many self transitions that have occurred at any given time.

However, to estimate $P(z_t|q_t, d_t)$ as a discrete distribution, each state q is assigned a threshold D_q for the duration variable d. The weights for the state durations exceeding D_q are then assumed to be constant. It is important to note that this modification does not restrict the number of self transitions to a particular state. The process may be in a specific state for an infinite number of time frames, however $P(z_t|q_t, d_t)$ will not change when d_t exceeds D_q:

$$P(z_t|q_t, d_t + 1) = P(z_t|q_t, d_t) \quad \text{if} \quad d_t \geq D_q \tag{5.5}$$

Making this modification not only helps in removing the time dependency problem of the weights, but also significantly reduces the number of parameters to be estimated. This is because the original formulation requires the calculation of weights for every time instance for every state. The modified structure will be referred to as Non-negative durational HMM (NdHMM).

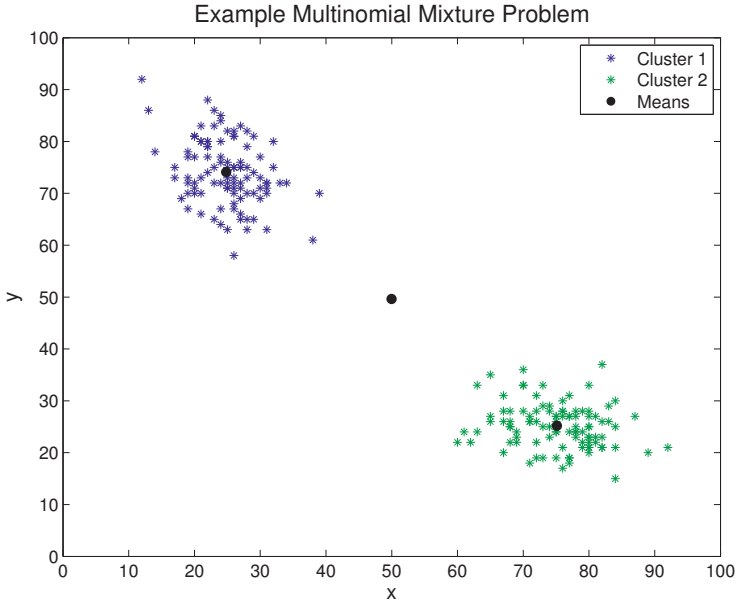


Figure 5.1: Example of multinomial mixture problem. The clusters are represented by blue and green marks, while the means of each cluster and the total mean are represented by black circles. The original formulation will result in a distribution describing the area around the total mean in the middle, whereas the new formulation will result in a distribution describing the area around both the cluster means.

The output space of a state in the model is defined by the multinomial distributions that constitute the dictionary. In the original N-HMM model the output probability density function for a given time frame was one single multinomial created by a combination of the dictionary components. For every time frame, the model was allowed to fine tune the output probability function by adjusting the weights, which were only used for that time frame. Thus, the probability function could be designed to describe

a freely chosen subspace of the space defined by the dictionary. Whereas in the new NdHMM formulation, the weights are reused for every visit to a state, and thus the model loses the freedom to adjust the output density function for each frame separately. Consider the example shown in Figure 5.1. The data to be modelled is distributed into two clusters, one with mean $[25, 75]^T$ and one with mean $[75, 25]^T$ which are represented by black circles in the figure. If the data is modelled by two dictionary components, it is fair to assume that the two components will be given by the distributions $P(f|z_1, q) = \mathcal{M}(\{0.25, 0.75\})$ and $P(f|z_2, q) = \mathcal{M}(\{0.75, 0.25\})$, where $\mathcal{M}(\{p, 1-p\})$ denotes the multinomial distribution with probabilities p and $1-p$. The mean of the energy distribution will be 100. Since the two clusters have the same quantity, the weights will be equal to 0.5 for both: $P(z_1|q, d) = 0.5$ and $P(z_2|q, d) = 0.5$. Hence, the output probability density function will be:

$$\begin{aligned} P(f|q) &= \sum_z P(z|q, d)P(f|z, q) \\ &= P(z_1|q, d)P(f|z, q) + P(z_2|q, d)P(f|z, q) \\ &= 0.5\mathcal{M}(\{0.25, 0.75\}) + 0.5\mathcal{M}(\{0.75, 0.25\}) \\ &= \mathcal{M}(\{0.5, 0.5\}) \end{aligned} \tag{5.6}$$

The model is therefore modelling the area around the middle black circle in the figure, and most of the training data points will be unlikely to occur according to the model. By forcing the model to draw all the frequency bins from one dictionary component when generating an output vector, the model would be able to handle such cases. The model in the example would then for every output vector either draw from $\mathcal{M}(\{0.25, 0.75\})$ or $\mathcal{M}(\{0.75, 0.25\})$, which would better fit the training data. The modification that is proposed is therefore to: for every output vector, first draw one of the dictionary components, and then fill the entire output vector with "sound quanta" by drawing frequency bins from only that dictionary component.

Algorithm 5.2 Generative process of NdHMM

Draw q_1 from $P(q_1)$
Set $d_1 = 0$
for $t = 1 \rightarrow T$ **do**
  Draw v_t from $P(v_t|q_t)$
  Draw z_t from $P(z_t|q_t, d_t)$
  for $v = 1 \rightarrow v_t$ **do**
    Draw f from $P(f|q_t, z_t)$
    $\underline{x}_t[f] = \underline{x}_t[f] + 1$ ▷ Add the sound quantum to the frequency bin
  end for
  Draw q_{t+1} from $P(q_{t+1}|q_t)$
  if $q_{t+1} = q_t$ **then**
    $d_{t+1} = \min(d_t + 1, D_{q_t})$
  else
    $d_{t+1} = 0$
  end if
end for

The likelihood for the NdHMM to produce an observation sequence is then given by:

$$P(\bar{\mathbf{x}}) = \sum_{\bar{\mathbf{q}}} \sum_{\bar{\mathbf{z}}} P(q_1) \left(\prod_{t=1}^{T-1} P(q_{t+1}|q_t) \right) \left(\prod_{t=1}^{T} P(v_t|q_t) \right) \left(\prod_{t=1}^{T} P(z_t|q_t, d_t) \right) \left(\prod_{t=1}^{T} P(\underline{x}_t|v_t, z_t, q_t) \right) \tag{5.7}$$

where T is the number of observation vectors. Note that d_t is given by the state sequence $\bar{\mathbf{q}}$, and therefore it is not necessary to sum over all possible values of d_t. The generative process of the NdHMM for creating the output sequence $\bar{\mathbf{x}} = \{\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_T\}$ is given in Algorithm 5.2. As an example of how to draw an output vector, consider the process entering the state shown in Figure 5.2. The "energy" v_t is first drawn from the energy distribution, and a dictionary component z_t is drawn by using the weights

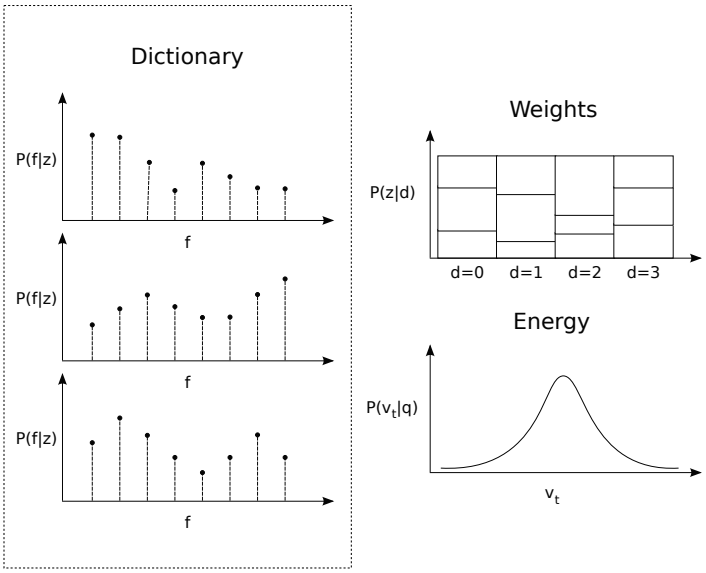


Figure 5.2: Content of a NdHMM state for an example with dictionary size of 3 and a maximum duration of 3.

$P(z|d)$ corresponding to $d = 0$. Then the output vector is generated by drawing v_t times from the multinomial distribution $P(f|z_t)$.

5.2.1 NdHMM vs HMM

The key difference between the conventional Gaussian HMM and the proposed NdHMM is the output probability distribution. Instead of Gaussian mixture models the NdHMM uses multinomial probability distributions. The output probability functions of the NdHMM are time varying, while they are fixed for the conventional HMM. Thus, the NdHMM is able to model the evolution of the unit to be recognised by a single state, while the conventional HMM has to use several states. The NdHMM also explicitly

models the "energy", which helps in discriminating between states. The original N-HMM shares these differences, but instead of reusing weights for each visit to a state, it has a separate set of weights for each time instance for each state. This enables it to adjust the weights for each time frame individually and thus gives better modelling capability. Unfortunately, this limits the N-HMM from describing unseen data, which is a problem for applications like ASR.

It follows from the independence between each draw that the observations are assumed to be independent in terms of both time and frequency. The assumption of independence between time frames is clearly not correct, and if the input is a spectrogram, the assumed independence between frequencies is also incorrect. An assumption of conditional independence between the time frames is also made for the conventional HMM, and often GMMs with a diagonal covariance matrix are used for the output observations leading to the same assumptions as for N-HMM. In order to alleviate the problem with the assumption of independence between feature components transforms with decorrelation properties are often used. For instance the cosine transform is used as the last step in the extraction process for the MFCCs.

At first glance it might seem that the N-HMM is using a lot of parameters because of all the multinomial distributions. However, the multinomials are replacing the GMMs in the standard HMM. A GMM consists of a set of weights, mean vectors and covariance matrices. Assuming that diagonal covariance matrices are used each mean vector and covariance matrix consists of a number of parameter equal to the dimension of the observation vectors. If the dimension of the input vectors for the two systems are the same, each mixture component in the GMM consists of twice the amount of parameters compared to a dictionary component from the N-HMM. The number of weights is higher for the N-HMM as each state has several sets. However for the conventional HMM each phone in a phone recognition system is represented by several states, usually 3-5

states, while the N-HMM is designed to use only one.

5.3 Parameter Estimation

In this section the equations used for estimating the parameters of the NdHMM will be derived. The procedure is similar to the one presented in [81], which in turn is based on the Baum-Welch algorithm which was discussed in Section 2.6. Although the basis of the derivation is similar to the one in [81], most of the equations have significant changes in order to accomodate the modifications applied to the N-HMM.

The complete set of parameters in the NdHMM are:

The dictionaries: multinomial distributions: $P(f|z,q)$

The weights: multinomial distributions $P(z|q,d)$

Energy distributions: gaussian distributions $P(v|q)$

Transition probabilities: Markov model $P(q_{t+1}|q_t)$

Initial state probabilities: $P(q_1)$

As in standard HMM, the input vectors are assumed to be conditionally independent, and the transition probabilities are only dependent on the current state.

Before the EM-algorithm can be applied the model needs to be initialised. This may be done by dividing the training data onto the states before conducting a simple K-Means clustering of the data for each state.

In order to find the new parameters of the NdHMM, the EM algorithm maximises the expected data log-likelihood $E_{\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}}\left[\log \hat{P}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{q}},\bar{\mathbf{v}})\right]$. The first step, the E-step, is therefore to calculate the expression for the expected log likelihood. In the Baum-Welch procedure for the conventional

HMM this step is conducted by introducing forward and backward probabilities (see Section 2.6). For the NdHMM the same approach will be used in order to compute the values of the expectation expression. The next step, the M-step, is to maximise the expectation expression with regard to the new parameters. This is done by partial differentiation of the expectation expression. To ensure that the probability distributions fulfil the requirement of summing to one, Lagrange multipliers are used in the differentiation.

5.3.1 E-Step

In the expectation step (E-Step) the expression for the expected data log-likelihood $E_{\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}}\left[\log \hat{P}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{q}},\bar{\mathbf{v}})\right]$ of the NdHMM is derived. Since each time frame of the input is assumed dependent on the current state only, the complete data log-likelihood is found by summing the log-likelihood of all the state transitions and the observation vectors:

$$\begin{aligned}\log P(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{q}},\bar{\mathbf{v}}) = \log P(q_1) &+ \sum_{t=1}^{T-1} \log P(q_{t+1}|q_t) \\ &+ \sum_{t=1}^{T} \log P(v_t|q_t) + \sum_{t=1}^{T} \log P(z_t|q_t,d_t) \\ &+ \sum_{t=1}^{T} \log P(\underline{x}_t|v_t,z_t,q_t), \end{aligned} \tag{5.8}$$

where $\bar{\mathbf{x}}$, $\bar{\mathbf{z}}$, $\bar{\mathbf{q}}$ and $\bar{\mathbf{v}}$ denotes the sequence of feature vectors, dictionary components, states and energy respectively. Let $\hat{P}(\cdot)$ denote the new probabilities, while $P(\cdot)$ denote the current probabilities. The EM-algorithm dictates that the new parameters are found by first using the current parameters to find the conditional expectation of the log likelihood which is calculated with the new parameters:

$$
\begin{aligned}
\mathcal{L} &= E_{\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}}\left[\log \hat{P}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{q}},\bar{\mathbf{v}})\right] \\
&= \sum_{\bar{\mathbf{q}}}\sum_{\bar{\mathbf{z}}} P(\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(q_1) \\
&+ \sum_{t=1}^{T-1}\sum_{\bar{\mathbf{q}}}\sum_{\bar{\mathbf{z}}} P(\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(q_{t+1}|q_t) \\
&+ \sum_{t=1}^{T}\sum_{\bar{\mathbf{q}}}\sum_{\bar{\mathbf{z}}} P(\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(v_t|q_t) \\
&+ \sum_{t=1}^{T}\sum_{\bar{\mathbf{q}}}\sum_{\bar{\mathbf{z}}} P(\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(z_t|q_t,d_t) \\
&+ \sum_{t=1}^{T}\sum_{\bar{\mathbf{q}}}\sum_{\bar{\mathbf{z}}} P(\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(\underline{x}_t|v_t,z_t,q_t). \qquad (5.9)
\end{aligned}
$$

The sum over all possible sequences can be changed to be a sum over all possible states and dictionary components for all time frames. Then realising that variables not present inside the log are marginalised, results

in:

$$
\begin{aligned}
\mathcal{L} &= E_{\bar{\mathbf{z}},\bar{\mathbf{q}}|\bar{\mathbf{x}},\bar{\mathbf{v}}}\left[\log \hat{P}(\bar{\mathbf{x}},\bar{\mathbf{z}},\bar{\mathbf{q}},\bar{\mathbf{v}})\right] \\
&= \sum_{q_1} P(q_1|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(q_1) \\
&+ \sum_{t=1}^{T-1} \sum_{q_t} \sum_{q_{t+1}} P(q_t, q_{t+1}|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(q_{t+1}|q_t) \\
&+ \sum_{t=1}^{T} \sum_{q_t} P(q_t|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(v_t|q_t) \\
&+ \sum_{t=1}^{T} \sum_{q_t} \sum_{d_t=0}^{D_{q_t}} \sum_{z_t} P(z_t, q_t, d_t|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(z_t|q_t, d_t) \\
&+ \sum_{t=1}^{T} \sum_{q_t} \sum_{z_t} P(z_t, q_t|\bar{\mathbf{x}},\bar{\mathbf{v}}) \log \hat{P}(\underline{x}_t|v_t, z_t, q_t) \\
&+ \kappa \left(1 - \sum_{q_1} \hat{P}(q_1)\right) + \sum_{q_t} \eta_{q_t} \left(1 - \sum_{q_{t+1}} \hat{P}(q_{t+1}|q_t)\right) \\
&+ \sum_{q} \sum_{d} \tau_{d,q} \left(1 - \sum_{z} \hat{P}(z|q,d)\right) \\
&+ \sum_{q} \sum_{z} \rho_{z,q} \left(1 - \sum_{f} \hat{P}(f|z,q)\right), \qquad (5.10)
\end{aligned}
$$

where also the Lagrange multipliers have been added in order for the probability distributions to have a sum equal to one. Note that each of the terms consist of only one of the updated distributions. Hence, the expression may be maximised with regard to each of the new distributions individually. In order to find the expression for the new probabilities that maximises the log likelihood, the posteriors in the equation have to be calculated. The posteriors are computed using the forward-backward

algorithm by defining the forward-backward probabilities:

$$\alpha(q_t, d_t) = P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t) \tag{5.11}$$

$$\beta(q_t, d_t) = P(\bar{\mathbf{x}}_{t+1}^T, \bar{\mathbf{v}}_{t+1}^T | q_t, d_t) \tag{5.12}$$

$$\gamma(q_t, d_t) = P(q_t, d_t | \bar{\mathbf{x}}, \bar{\mathbf{v}}) \tag{5.13}$$

where $\bar{\mathbf{x}}_1^t = \underline{x}_1, \underline{x}_2, \cdots, \underline{x}_t$.

In order to compute the forward-backward probabilities, the probability of a sample given the state and duration is needed:

$$\begin{aligned} P(\underline{x}_t, v_t | q_t, d_t) =& P(v_t | q_t, d_t) P(\underline{x}_t | q_t, d_t) \\ =& P(v_t | q_t, d_t) \sum_{z_t} P(\underline{x}_t, z_t | v_t, q_t, d_t) \\ =& P(v_t | q_t, d_t) \sum_{z_t} P(\underline{x}_t | v_t, z_t, q_t) P(z_t | v_t, q_t, d_t), \end{aligned} \tag{5.14}$$

where the probability $P(\underline{x}_t | v_t, z_t, q_t)$ is given by the definition of the multinomial probability density function:

$$P(\underline{x}_t | v_t, z_t, q_t) = \binom{v_t}{x_t[1], x_t[2], \cdots, x_t[F]} \prod_f P(f | z_t, q_t)^{x_t[f]} \tag{5.15}$$

thus:

$$\begin{aligned} P(\underline{x}_t, v_t | q_t, d_t) = P(v_t | q_t) \sum_{z_t} P(z_t | q_t, d_t) \\ \binom{v_t}{x_t[1], x_t[2], \cdots, x_t[F]} \prod_f P(f | z_t, q_t)^{x_t[f]} \end{aligned} \tag{5.16}$$

The α and β probabilities may then be calculated by a recursive procedure.

First, the α probabilities are initialised by:

$$\begin{aligned}\alpha_1(q_1, d_1 = 0) &= P(\underline{x}_1, v_1, q_1, d_1 = 0) \\ &= P(\underline{x}_1, v_1 | q_1, d_1 = 0) \cdot P(d_1 = 0 | q_1) \cdot P(q_1) \\ &= P(\underline{x}_1, v_1 | q_1) \cdot P(q_1), \end{aligned} \tag{5.17}$$

and the recursive computation is done by:

$$\begin{aligned}\alpha_{t+1}(q_{t+1}, d_{t+1}) =& P(\bar{\mathbf{x}}_1^{t+1}, \bar{\mathbf{v}}_1^{t+1}, q_{t+1}, d_{t+1}) \\ =& P(\underline{x}_{t+1}, v_{t+1} | \bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_{t+1}, d_{t+1}) P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_{t+1}, d_{t+1}) \\ =& P(\underline{x}_{t+1}, v_{t+1} | q_{t+1}, d_{t+1}) \\ & \sum_{q_t} \sum_{d_t} P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t, q_{t+1}, d_{t+1}) \\ =& P(\underline{x}_{t+1}, v_{t+1} | q_{t+1}, d_{t+1}) \\ & \sum_{q_t} \sum_{d_t} P(q_{t+1}, d_{t+1} | \bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t) P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t) \\ =& P(\underline{x}_{t+1}, v_{t+1} | q_{t+1}, d_{t+1}) \\ & \sum_{q_t} \sum_{d_t} P(d_{t+1} | q_{t+1}, q_t, d_t) P(q_{t+1} | q_t) \alpha_t(q_t, d_t), \end{aligned} \tag{5.18}$$

where $P(d_{t+1}|q_{t+1}, q_t, d_t)$ ensures that only valid combinations of q_{t+1}, d_{t+1}, q_t and d_t are included in the summation.

For the β probabilities the initialisation is:

$$\beta_T(q_T, d_T) = 1, \tag{5.19}$$

and the recursive computation becomes:

$$\begin{aligned}
\beta_t(q_t, d_t) =& P(\bar{\mathbf{x}}_{t+1}^T, \bar{\mathbf{v}}_{t+1}^T | q_t, d_t) \\
=& \sum_{q_{t+1}} \sum_{d_{t+1}} P(\bar{\mathbf{x}}_{t+1}^T, \bar{\mathbf{v}}_{t+1}^T, q_{t+1}, d_{t+1} | q_t, d_t) \\
=& \sum_{q_{t+1}} \sum_{d_{t+1}} P(\bar{\mathbf{x}}_{t+2}^T, \bar{\mathbf{v}}_{t+2}^T | \underline{x}_{t+1}, v_{t+1}, q_{t+1}, d_{t+1}, q_t, d_t) \\
& P(\underline{x}_{t+1}, v_{t+1}, q_{t+1}, d_{t+1} | q_t, d_t) \\
=& \sum_{q_{t+1}} \sum_{d_{t+1}} P(\bar{\mathbf{x}}_{t+2}^T, \bar{\mathbf{v}}_{t+2}^T | q_{t+1}, d_{t+1}) \\
& P(\underline{x}_{t+1}, v_{t+1} | q_{t+1}, d_{t+1}, q_t, d_t) P(q_{t+1}, d_{t+1} | q_t, d_t) \\
=& \sum_{q_{t+1}} \sum_{d_{t+1}} \beta_{t+1}(q_{t+1}, d_{t+1}) P(\underline{x}_{t+1}, v_{t+1} | q_{t+1}, d_{t+1}) \\
& P(d_{t+1} | q_{t+1}, q_t, d_t) P(q_{t+1} | q_t),
\end{aligned} \tag{5.20}$$

where, once again, $P(d_{t+1}|q_{t+1}, q_t, d_t)$ ensures that only valid combinations of q_{t+1}, d_{t+1}, q_t and d_t are included in the summation.

Now, $\gamma_t(q_t, d_t)$ can be calculated by:

$$\begin{aligned}
\gamma_t(q_t, d_t) &= P(q_t, d_t | \bar{\mathbf{x}}, \bar{\mathbf{v}}) \\
&= \frac{P(\bar{\mathbf{x}}, \bar{\mathbf{v}}, q_t, d_t)}{\sum_{q_t} \sum_{d_t} P(\bar{\mathbf{x}}, \bar{\mathbf{v}}, q_t, d_t)} \\
&= \frac{P(\bar{\mathbf{x}}_{t+1}^T, \bar{\mathbf{v}}_{t+1}^T | \bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t) P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t)}{\sum_{q_t} \sum_{d_t} P(\bar{\mathbf{x}}_{t+1}^T, \bar{\mathbf{v}}_{t+1}^T | \bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t) P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t)} \\
&= \frac{\alpha_t(q_t, d_t) \cdot \beta_t(q_t, d_t)}{\sum_{q_t} \sum_{d_t} \alpha_t(q_t, d_t) \cdot \beta_t(q_t, d_t)}
\end{aligned} \tag{5.21}$$

After computing the forward-backward probabilities, the posteriors in Equation (5.10) may be computed. The posterior for the initial transi-

tion probability is simply:

$$P(q_1|\bar{\mathbf{x}},\bar{\mathbf{v}}) = P(q_1, d_1 = 0|\bar{\mathbf{x}},\bar{\mathbf{v}}) = \gamma_1(q_1, d_1 = 0). \tag{5.22}$$

The distribution needed for calculating transition probabilities is somewhat more complex:

$$P(q_t, q_{t+1}|\bar{\mathbf{x}},\bar{\mathbf{v}}) = \frac{P(\bar{\mathbf{x}},\bar{\mathbf{v}}, q_t, q_{t+1})}{\sum_{q_t}\sum_{q_{t+1}} P(\bar{\mathbf{x}},\bar{\mathbf{v}}, q_t, q_{t+1})} \tag{5.23}$$

where

$$\begin{aligned}
P(\bar{\mathbf{x}},\bar{\mathbf{v}}, q_t, q_{t+1}) &= \sum_{d_{t+1}}\sum_{d_t} P(\bar{\mathbf{x}},\bar{\mathbf{v}}, q_t, d_t, q_{t+1}, d_{t+1}) \\
&= \sum_{d_t}\sum_{d_{t+1}} P(\bar{\mathbf{x}}_{t+1}^T, \bar{\mathbf{v}}_{t+1}^T|\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t, q_{t+1}, d_{t+1}) \\
&\quad P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t, q_t, d_t, q_{t+1}d_{t+1}) \\
&= \sum_{d_t}\sum_{d_{t+1}} P(\bar{\mathbf{x}}_{t+1}^T, \bar{\mathbf{v}}_{t+1}^T|q_{t+1}, d_{t+1}) \\
&\quad P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t|q_t, d_t, q_{t+1}, d_{t+1}) P(q_t, d_t, q_{t+1}, d_{t+1}) \\
&= \sum_{d_t}\sum_{d_{t+1}} P(\bar{\mathbf{x}}_{t+2}^T, \bar{\mathbf{v}}_{t+2}^T|\underline{x}_{t+1}, v_{t+1}, q_{t+1}, d_{t+1}) \\
&\quad P(\underline{x}_{t+1}, v_{t+1}|q_{t+1}, d_{t+1}) P(\bar{\mathbf{x}}_1^t, \bar{\mathbf{v}}_1^t|q_t, d_t) \\
&\quad P(q_t, d_t, q_{t+1}, d_{t+1}) \\
&= \sum_{d_t}\sum_{d_{t+1}} \alpha_t(q_t, d_t)\beta_{t+1}(q_{t+1}, d_{t+1}) P(q_{t+1}|q_t) \\
&\quad P(\underline{x}_{t+1}, v_{t+1}|q_{t+1}, d_{t+1}) P(d_{t+1}|q_{t+1}, q_t, d_t),
\end{aligned} \tag{5.24}$$

where the $P(d_{t+1}|q_{t+1}, q_t, d_t)$ again ensures that only valid combinations of q_{t+1}, d_{t+1}, q_t and d_t are included in the summation.

Next, the probability distribution of the states which is needed for the energy distribution:

$$P(q_t|\bar{\mathbf{x}}, \bar{\mathbf{v}}) = \sum_{d_t} P(q_t, d_t|\bar{\mathbf{x}}, \bar{\mathbf{v}}) = \sum_{d_t} \gamma_t(q_t, d_t), \tag{5.25}$$

and finally the distributions needed for the weights and dictionary components:

$$P(z_t, q_t|\bar{\mathbf{x}}, \bar{\mathbf{v}}) = \sum_{d_t} P(z_t, q_t, d_t|\bar{\mathbf{x}}, \bar{\mathbf{v}}) \tag{5.26}$$

$$\begin{aligned} P(z_t, q_t, d_t|\bar{\mathbf{x}}, \bar{\mathbf{v}}) =& P(z_t|q_t, d_t, \bar{\mathbf{x}}, \bar{\mathbf{v}}) P(q_t, d_t|\bar{\mathbf{x}}, \bar{\mathbf{v}}) \\ =& P(z_t|q_t, d_t, \underline{x}_t)\gamma_t(q_t, d_t) \\ =& \gamma_t(q_t, d_t) \frac{P(\underline{x}_t, z_t|q_t, d_t)}{P(\underline{x}_t|q_t, d_t)} \\ =& \gamma_t(q_t, d_t) \frac{P(\underline{x}_t|z_t, q_t, d_t)P(z_t|q_t, d_t)}{\sum_{z_t} P(\underline{x}_t|z_t, q_t, d_t)P(z_t|q_t, d_t)} \\ =& \gamma_t(q_t, d_t) \frac{P(\underline{x}_t|z_t, q_t)P(z_t|q_t, d_t)}{\sum_{z_t} P(\underline{x}_t|z_t, q_t)P(z_t|q_t, d_t)} \end{aligned} \tag{5.27}$$

$$\tag{5.28}$$

5.3.2 M-Step

With all the needed distributions calculated, it is possible to find the derivatives of Equation (5.10) with respect to each of the parameters. This is referred to as the M-step in the EM-algorithm. For the case of the initial state probabilities, the derivative is:

$$\frac{\partial \mathcal{L}}{\partial \hat{P}(q_1)} = \frac{P(q_1|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\hat{P}(q_1)} - \kappa. \tag{5.29}$$

Using the requirement that $\sum_{q_1} \hat{P}(q_1) = 1$ and set the derivative to zero, gives $\kappa = \sum_{q_1} P(q_1|\bar{\mathbf{x}}, \bar{\mathbf{v}})$. Hence:

$$\begin{aligned}\hat{P}(q_1) =& \frac{P(q_1|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\sum_{q_1} P(q_1|\bar{\mathbf{x}}, \bar{\mathbf{v}})} \\ =& \frac{\gamma_1(q_1, d_1 = 0)}{\sum_{q_1} \gamma_1(q_1, d_1 = 0)}\end{aligned} \tag{5.30}$$

For the transition from state q to state q^*, the derivative is:

$$\frac{\partial \mathcal{L}}{\partial \hat{P}(q^*|q)} = \sum_t \frac{P(q_t = q, q_{t+1} = q^*|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\hat{P}(q^*|q)} - \eta_q \tag{5.31}$$

Eliminating η_q gives:

$$\hat{P}(q^*|q) = \frac{\sum_t P(q_t = q, q_{t+1} = q^*|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\sum_t \sum_{q^*} P(q_t = q, q_{t+1} = q^*|\bar{\mathbf{x}}, \bar{\mathbf{v}})} \tag{5.32}$$

The state energy distributions are Gaussian distributions, which means each of them contains two parameters: a mean (μ_q) and a variance (σ_q^2), and thus:

$$\hat{P}(v|q) = \frac{1}{\sqrt{2\pi\sigma_q^2}} \exp\left(-\frac{(v - \mu_q)^2}{2\sigma_q^2}\right) \tag{5.33}$$

The derivate of Equation (5.10) with respect to the mean (μ_q) of the state q is:

$$\frac{\partial \mathcal{L}}{\partial \mu_q} = \sum_t \frac{P(q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\hat{P}(v_t|q)} \frac{\partial}{\partial \mu_q} \hat{P}(v_t|q), \tag{5.34}$$

where

$$\begin{aligned}\frac{\partial}{\partial \mu_q}\hat{P}(v_t|q) =& \frac{1}{\sqrt{2\pi\sigma_\mu^2}}\frac{2(v_t-\mu_q)}{2\sigma_q^2}\exp\left(-\frac{(v_t-\mu_q)^2}{2\sigma_q^2}\right)\\ =& \frac{(v_t-\mu_q)}{\sigma_q^2}\hat{P}(v_t|q)\end{aligned} \tag{5.35}$$

Setting $\frac{\partial \mathcal{L}}{\partial \mu_q}$ and solving for μ_q yields:

$$\begin{aligned}\mu_q =& \frac{\sum_t v_t P(q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\sum_t P(q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}\\ =& \frac{\sum_t \sum_d v_t \gamma_t(q, d_t)}{\sum_t \sum_d \gamma_t(q, d_t)}\end{aligned} \tag{5.36}$$

Similarly, the derivate of Equation (5.10) with respect to the variance σ_q^2 is:

$$\frac{\partial \mathcal{L}}{\partial \sigma_q^2} = \sum_t \frac{P(q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\hat{P}(v_t|q)}\frac{\partial}{\partial \sigma_q^2}\hat{P}(v_t|q), \tag{5.37}$$

where

$$\frac{\partial}{\partial \sigma_q^2}\hat{P}(v_t|q) = \left(\frac{(v_t-\mu_q)^2}{2\sigma_q^{2\cdot 2}} - \frac{1}{2\sigma_q^2}\right)\hat{P}(v_t|q) \tag{5.38}$$

Setting $\frac{\partial \mathcal{L}}{\partial \sigma_q^2} = 0$ and solve for σ_q^2 results in:

$$\begin{aligned}\sigma_q^2 =& \frac{\sum_t P(q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})(v_t-\mu_q)^2}{\sum_t P(q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}\\ =& \frac{\sum_t \sum_{d_t} \gamma_t(q, d_t)(v_t-\mu_q)^2}{\sum_t \sum_{d_t} \gamma_t(q, d_t)}\end{aligned} \tag{5.39}$$

In order to find the update equation for the weights for the state q and duration d, the Equation (5.10) has to be differentiated with respect to $\hat{P}(z|q,d)$:

$$\frac{\partial \mathcal{L}}{\partial \hat{P}(z|q,d)} = \sum_t \frac{P(z_t = z, q_t = q, d_t = d|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\hat{P}(z|q,d)} - \tau_{q,d} \tag{5.40}$$

and, again, setting to zero and eliminating the Lagrange constant reveals:

$$\hat{P}(z|q,d) = \frac{\sum_t P(z_t = z, q_t = q, d_t = d|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\sum_t \sum_z P(z_t = z, q_t = q, d_t = d|\bar{\mathbf{x}}, \bar{\mathbf{v}})} \tag{5.41}$$

Finally, the derivative of Equation (5.10) with respect to $\hat{P}(f|z,q)$ is:

$$\frac{\partial \mathcal{L}}{\partial \hat{P}(f|z,q)} = \sum_t \frac{P(z_t = z, q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\hat{P}(\underline{x}_t|v_t, z, q)} \frac{\partial}{\partial \hat{P}(f|z,q)} \hat{P}(\underline{x}_t|v_t, z, q) - \rho_{z,q}, \tag{5.42}$$

where $\hat{P}(\underline{x}_t|v_t, z, q)$ is given by Equation (5.15) and the derivative is:

$$\begin{aligned} \frac{\partial}{\partial \hat{P}(f|z,q)} \hat{P}(\underline{x}_t|v_t, z, q) =& \binom{v_t}{x_t[1], x_t[2], \cdots, x_t[F]} x_t[f] \hat{P}(f|z,q)^{x_t[f]-1} \\ & \prod_{f^* \neq f} \hat{P}(f^*|z,q)^{x_t[f^*]} \\ =& x_t[f] \frac{\hat{P}(\underline{x}_t|v_t, z, q)}{\hat{P}(f|z,q)}, \end{aligned} \tag{5.43}$$

and consequently:

$$\frac{\partial \mathcal{L}}{\partial \hat{P}(f|z,q)} = \sum_t x_t[f] \frac{P(z_t = z, q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\hat{P}(f|z,q)} - \rho_{z,q}. \tag{5.44}$$

As before, the derivative is set equal to zero and the Lagrange constant is eliminated, which results in:

$$\hat{P}(f|z,q) = \frac{\sum_t x_t[f] P(z_t = z, q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})}{\sum_t v_t P(z_t = z, q_t = q|\bar{\mathbf{x}}, \bar{\mathbf{v}})} \tag{5.45}$$

The update equations are derived based on one observation sequence. However, it is easy to extend the equations to several observation sequences. This is done by changing the summations over time t in the equations to also include a summation over the sequences. Note that most of the equations have the summation over all possible outcomes of the distribution to be updated in the denominator. Hence, in practice only the numerators are calculated for each sentence, and only after all sentences have been processed, the denominators are calculated. The exception is the energy distribution, where the denominator has to be calculated separately.

5.4 Viterbi Decoding

The decoding of an unseen utterance using the NdHMM is performed using the Viterbi algorithm. However, there is an important difference from the Viterbi algorithm for the conventional HMM: since the output probability density function of the model is both dependent on the state q and the duration counter d, the best path probability needs to be remembered for all the combinations of q and d. Hence, the trellis network illustrating the Viterbi algorithm will have a separate node for each time, state and duration, in contrast to separate nodes for time and state for the conventional HMM. The best path probability may then be defined as:

$$V_t(q,d) = P(\bar{\mathbf{x}}_1^T, \bar{\mathbf{q}}_1^{t-1}, \bar{\mathbf{d}}_1^{t-1}, q_t = q, d_t = d), \tag{5.46}$$

which is the probability of the most likely state sequence at time t, which has generated the observations until time t and ends in state q with duration counter d. As for the conventional HMM, this probability is calculated in an inductive manner. First the initialisation:

$$V_1(q_1, 0) = P(q_1)P(\underline{x}_1, v_1|q_1, d_1 = 0) \tag{5.47}$$

$$B_1(q_1, 0) = (0, 0), \tag{5.48}$$

where $P(\underline{x}_1, v_1|q_1, d_1 = 0)$ is found by using Equation (5.16) (see page 83) and $B_t(q_t, d_t)$ keeps track of the previous state and duration. Further the inductive step is:

$$\begin{aligned} V_t(q_t, d_t) &= \max_{q_{t-1}, d_{t-1}} \left[V_{t-1}(q_{t-1}, d_{t-1})P(q_t, d_t|q_{t-1}, d_{t-1})\right] P(\underline{x}_t, v_t|q_t, d_t) \\ &= \max_{q_{t-1}, d_{t-1}} \left[V_{t-1}(q_{t-1}, d_{t-1})P(d_t|q_t, q_{t-1}, d_{t-1})P(q_t|q_{t-1})\right] \\ &\quad P(\underline{x}_t, v_t|q_t, d_t) \end{aligned} \tag{5.49}$$

$$B_t(q_t, d_t) = \operatorname*{argmax}_{q_{t-1}, d_{t-1}} \left[V_{t-1}(q_{t-1}, d_{t-1})P(d_t|q_t, q_{t-1}, d_{t-1})P(q_t|q_{t-1})\right], \tag{5.50}$$

where $P(d_t|q_t, q_{t-1}, d_{t-1})$ ensures that only valid combinations of q_t, d_t, q_{t+1} and d_{t+1} are considered.

After the inductive procedure is finished the path with the highest score may be found by first:

$$q_T^*, d_T^* = \operatorname*{argmax}_{q_T, d_T} \left[V_T(q_T, d_T)\right] \tag{5.51}$$

and then perform a backtracking procedure:

$$q_t^*, d_t^* = B_{t+1}(q_{t+1}^*, d_{t+1}^*). \tag{5.52}$$

to find the recognised state sequence: $(q_1^*, q_2^*, \cdots, q_T^*)$.

5.5 Experiments

In this section experiments conducted with the NdHMM will be presented. Again, the TIMIT database, which is described in Appendix A, has been chosen for the experiments. The parameter estimation (Baum-Welch) procedure and Viterbi decoding were both implemented in Python for convenience without much emphasis on time efficiency. Hence, some of the experiments require a significant number of hours to complete. Particularly the dimension of the input vectors is a factor that is costly to increase in the current implementation. However, the time consumption is likely to be reduced significantly by a more efficient implementation.

Since the model is designed for non-negative data, the choice of features has to be carefully considered. This is a topic that will be discussed next, followed by some small scale experiments that will give some insight in how the model works in Section 5.5.2. Finally in Section 5.5.3, full scale phone recognition experiments will be presented.

5.5.1 Feature Extraction

In [81] the original N-HMM was used on the spectrogram of the input signal. From the NMF perspective this is a good alternative, as the spectrogram is an image of the signal in both time and frequency containing only non-negative values. Hence, it is easy to imagine that the model would find recurring patterns in the image. Unfortunately, the dimension of the spectrogram is usually high, leading to a high number of parameters that needs to be estimated. Using a narrow window would reduce the dimension, but the frequency resolution would be compromised, which would lead to poorer performance. The spectrogram may therefore not be the best choice of features for the NdHMM when performing phone recognition.

The most used features in ASR are the MFCCs. Unfortunately, the MFCCs are *not* non-negative in their original formulation. However, it is possible to apply a transformation on the features to make them non-negative, and thereby a viable option for the NdHMM. The sigmoid function could be a good alternative for such a transformation:

$$f(x) = \frac{1}{1 - \exp(-\alpha(x - \mu_x))}, \tag{5.53}$$

where μ_x is the mean of the input and α is a constant controlling the slope of the function. The sigmoid is depicted in Figure 5.3 for a few different values of the slope factor. From the figure it is possible to envision that the transform is scaling and shifting the input signal to ensure that all the values are between zero and one. Unfortunately, very high or very low values are truncated which entails distortion. The amount of distortion is dependent on the dynamic range of the input signal and the slope factor. Thus, the slope factor has to be adjusted to fit the input data. Some experimentation showed that a slope factor of 0.05 results in low distortion, and was chosen for the experiments presented in this chapter.

Another set of possible features can be found by inspecting the extraction process of the MFCCs: the final step of the process is the discrete cosine transform (DCT). This is applied to the log-energy of the output of the Mel filterbank. Hence, by omitting the DCT, the features would be the log-energy of the filterbank output. Since energy is non-negative, the log-energy can be ensured to be non-negative by adding 1 before the log operation. In the MFCC extraction process the DCT is applied to reduce the dimensionality and for decorrelation purposes. Thus, by leaving this step out of the generation process, the dimensionality will be higher and the features will have higher correlation. However, they would not require any transformation. Also, compared to the original spectrogram, the dimensionality would be lower. The log-energy of the Mel filterbank output is therefore another possibility in the choice of features.

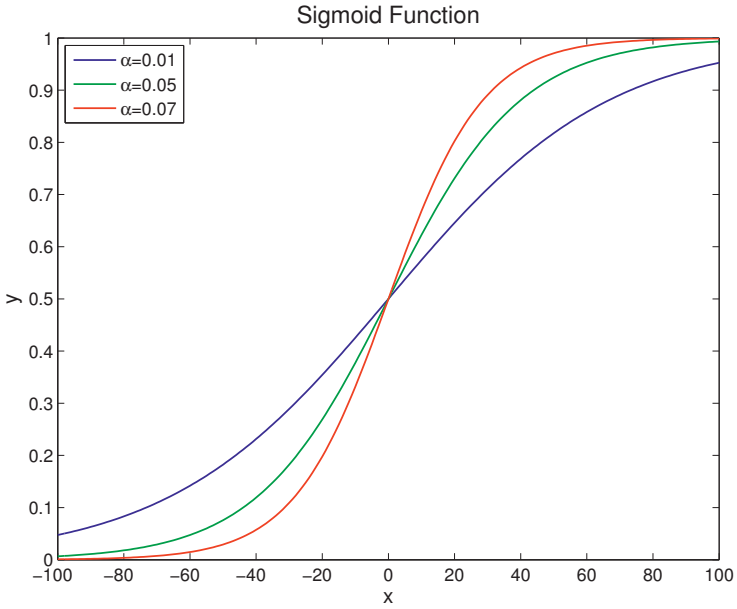


Figure 5.3: The Sigmoid Function

Since the NdHMM uses multinomial distributions to model the output vectors, the variances of the individual dimensions are not included. For a multinomial distribution the variance of each outcome is dependent on the number of draws N and the probability of that particular outcome p: $\sigma^2 = N \cdot p(1-p)$. Whereas, in the conventional HMM the variances are modelled with separate parameters. Thus if one of the dimensions have very low variance, a small deviation from the mean would have greater impact for the NdHMM compared to the HMM. If the input features are normalised with respect to the variance before they are presented to the NdHMM, the different dimensions would be emphasised more equally in the discrimination process.

The original N-HMM was designed for use on spectrograms, and were therefore interpreting the sum of the input feature vectors as energy. For the transformed MFCCs discussed above this is incorrect. After the DCT most of the energy information is placed in the 0^{th}coefficient. Hence, it would be reasonable to exclude the 0^{th}coefficient and make the sum of the rest of the non-negative vector equal to this energy coefficient. By doing so, the energy of the input signal is modelled by the energy distribution of the NdHMM, instead of as a part of the multinomial distributions.

5.5.2 Small Scale Experiments

In order to see if the model behaves as expected it has been trained on a small set of training data. This allows the parts of the NdHMM to easily be plotted along with some of the data. In the first experiment, the NdHMM was trained on isolated phone data. Both the dictionary components (solid lines) and the phone data (dotted lines) are shown in Figure 5.4 for the phones /ao/ and /er/. The log-energy of the output of a 26 channel Mel filterbank was used as features, and the original data were normalised to represent probability distributions before plotting. From the figure it is possible to observe that the dictionary components learn different variations in the data, though the frequency distributions have similar structure. For frequencies where the data have larger variation, the components are more different. For instance at the area around 1100 Hz in the lower figure, it seems the data have two variations which are covered by the two components. These dictionaries might be representing different contexts or sub-units of the phone under consideration. A comparison of the two plots in the figure shows how the NdHMM have captured the general structure of the data. The area around 1200-2500 Hz is quite different for the two cases, and both the components in both the states model the two general structures while covering the variations. In Figure 5.5 a similar plot is made for a conventional HMM, where each

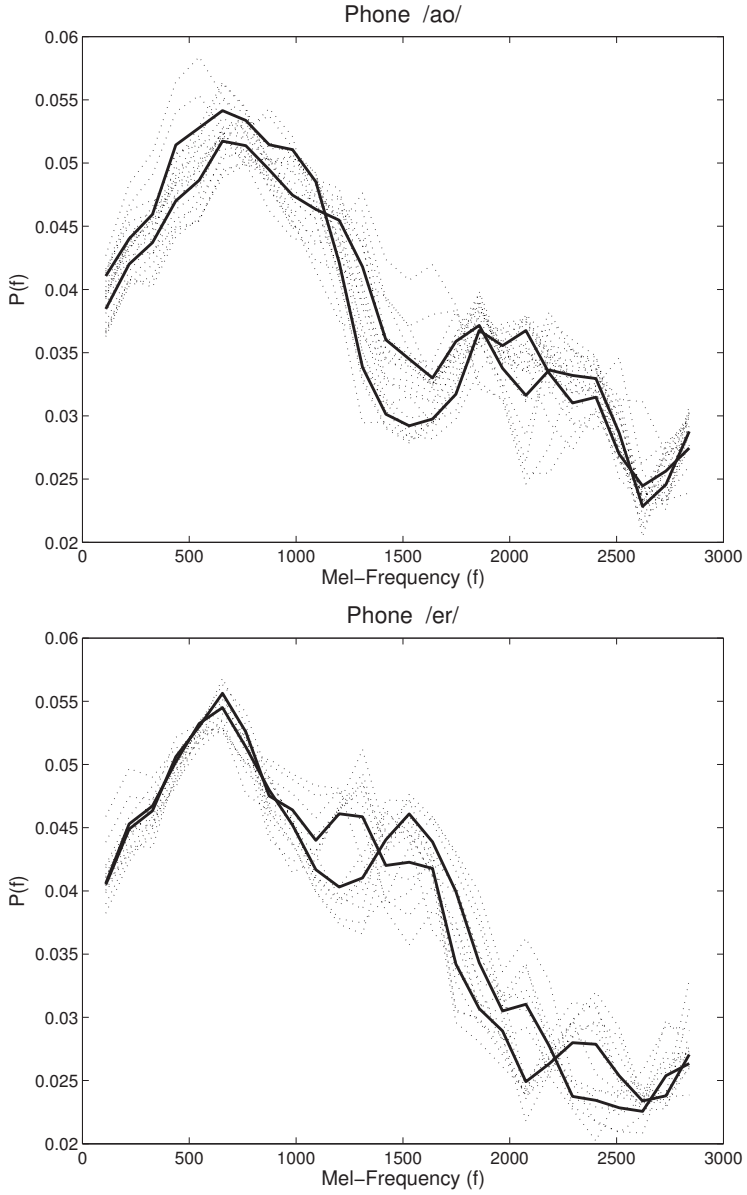


Figure 5.4: Illustrating the dictionary components (solid line) learned using NdHMM on phone data (dotted line) from TIMIT.

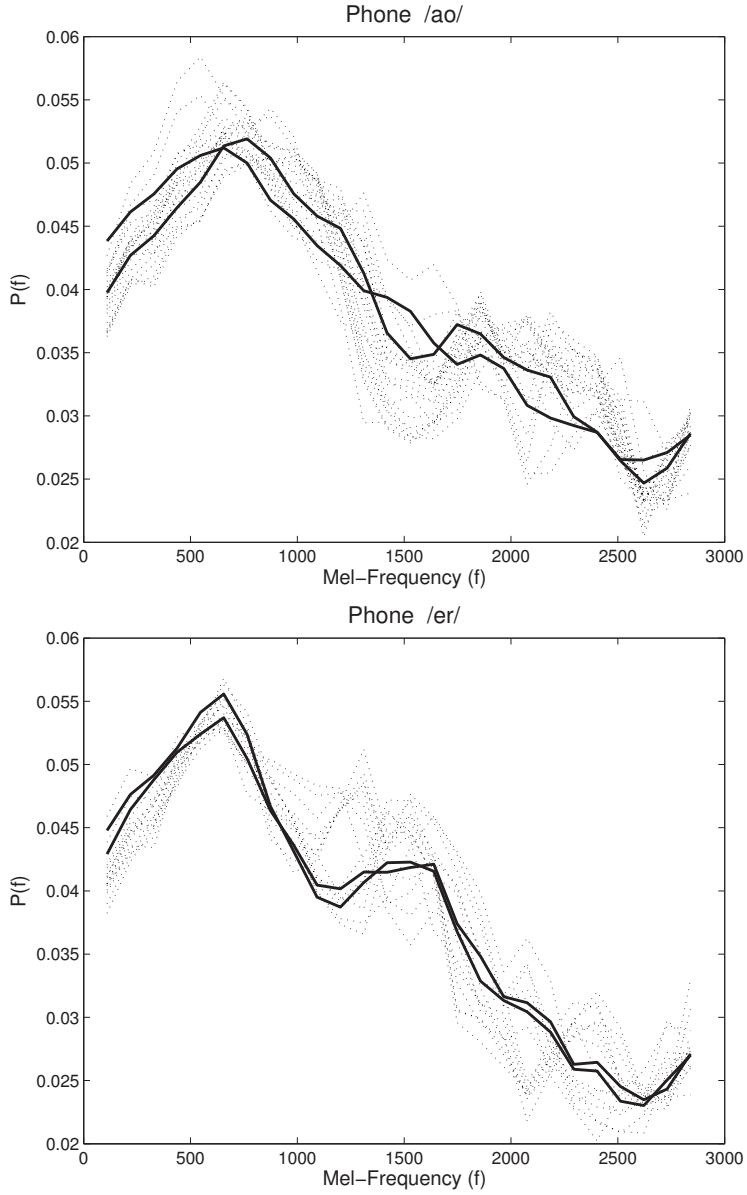


Figure 5.5: Normalised GMM means (solid line) of a single state conventional HMM trained on phone data (dotted line) from TIMIT.

phone was modelled by a single state with 2 component GMMs. The GMMs are represented by the normalised mean vectors. This is probably not a fair comparison as the HMM is not employed in a standard way. In addition the energy is completely ignored. Nevertheless it seems like the NdHMM is capturing the shape of the input in a better way than the HMM.

Another experiment was performed by only using the "sx70" and "sx231" sentences in the training set of the TIMIT corpus. In the entire training set there are 14 such sentences which consist of only 17 of the 39 phone set (see Appendix A). The NdHMM had 4 dictionary components for each state and the state durations were set individually to cover at least 90% of the durations in the data. As input to the model, the sigmoid transformed MFCCs discussed in Section 5.5.1 have been used. In Figure 5.6 and Figure 5.7 the dictionary components and the weights are shown for the two phones /er/ and /ay/. Compared to the previous experiment, the dimension is lower and the shape of the dictionary components is different, but they seem to have the same behaviour. The weights of the two phones show that they are modelling the temporal evolution of the phones as expected. In Figure 5.8 the transition matrix of the NdHMM and the parameters of the state energy distributions are plotted. The transition matrix is as expected: All the elements on the diagonal have high probabilities except for the first row which represent the initial state probabilities. Since all the sentences begins with the /h#/ which is mapped to /sil/ (state 13 in Figure 5.8), the initial state probabilities have only one non-zero component. High values on the diagonal means that self-transitions is very likely, something which is natural as most phones lasts for more than one frame. All the state energy distributions are represented by their mean (blue circle) plus/minus the standard deviation (red triangles). From the figure it is easy to imagine that the energy distributions supports the discriminative power of the model as several of the phones are seperated with more than one standard deviation.

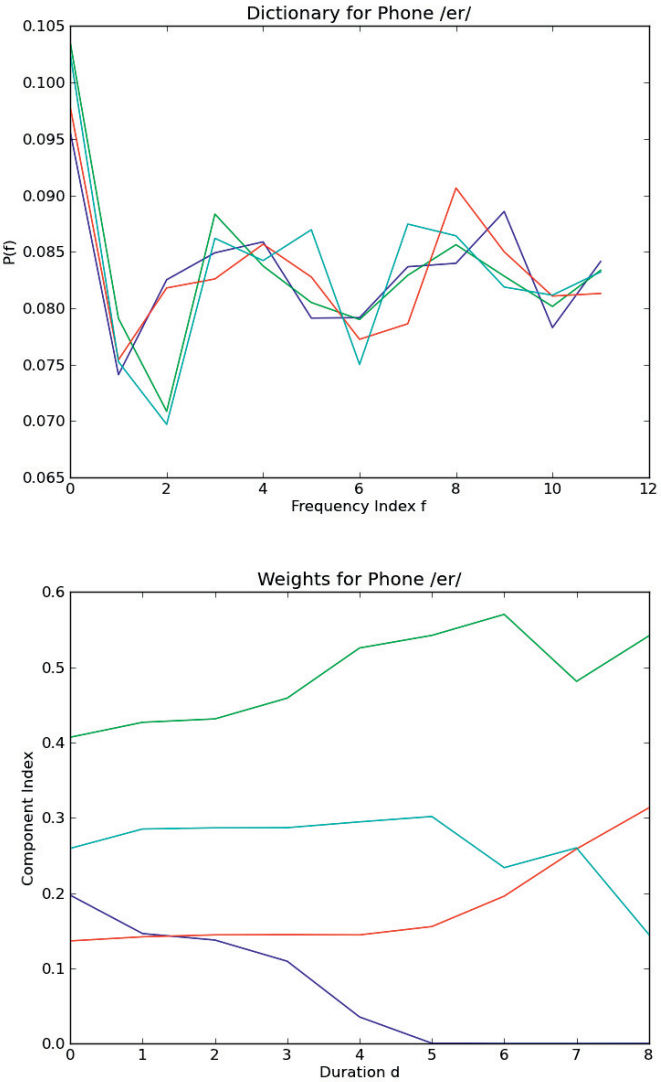


Figure 5.6: Weights and Dictionary for the phone /er/ learnt on the sx70 and sx231 files of TIMIT.

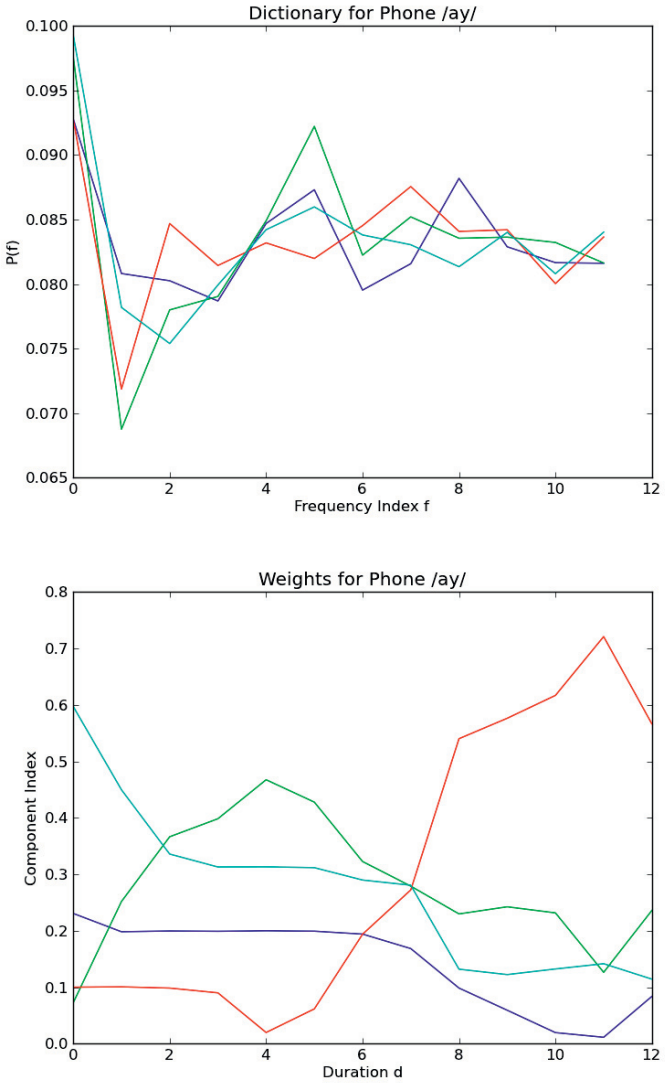


Figure 5.7: Weights and Dictionary for the phone /ay/ learnt on the sx70 and sx231 files of TIMIT

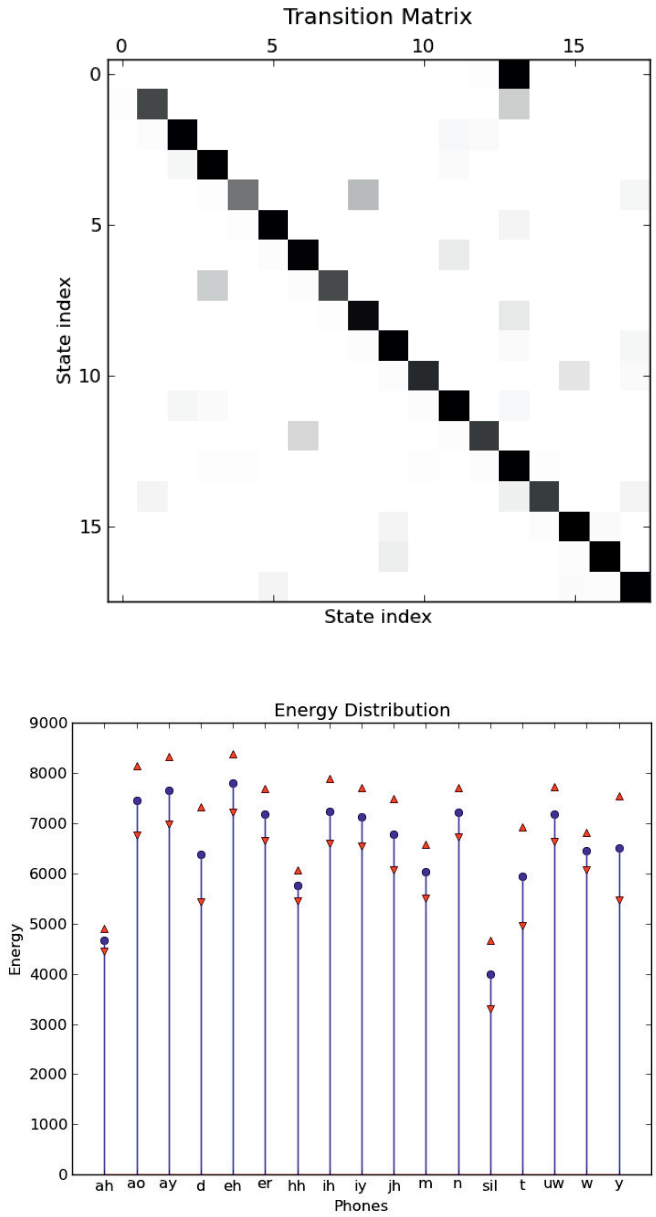


Figure 5.8: Transition matrix and state energy distribution for the NdHMM learned on the sx70 and sx231 files of TIMIT. The initial state probabilities are included in the first row of the matrix. The energy distributions are represented by their means (blue circles) plus/minus one standard deviation (red triangles).

5.5.3 TIMIT Phone Recognition

The phone recognition experiments were performed on the TIMIT database (see Appendix A). For baseline comparison a standard context independent 3-state left-to-right HMM with GMMs as emission densities was employed. In order to reduce the run time of the experiments, both training and testing were conducted with the use of the 39 phone set of TIMIT.

A set of non-negative 26 Mel-filterbank log-energy features was extracted using a 25 ms window with 10 ms shift and with 26 channels. In addition a separate set of 13 MFCCs including the C_0 was extracted. The baseline model was trained and tested on the latter 13 MFCCs, whereas the NdHMM was tried on both the Mel features and the MFCCs. When the MFCCs were combined with the NdHMM the sigmoid transform was applied to make them non-negative, and the C_0 coefficient was excluded before the vectors were scaled to have a sum equal to the C_0. These are denoted 12MFCC+S. In addition both sets of features were tried both with and without a variance normalisation (VN) (see Section 5.5.1). Hence, the NdHMM were presented with 4 different set of features: "MEL", "MEL+VN", "12MFCC+S" and "12MFCC+S+VN".

The maximum durations of the states in the NdHMM were set individually to cover 90% of the durations seen in the training data. The number of components in the GMMs for the HMM and the dictionaries for the NdHMM varied between 8, 16 and 32. An overview of the parameter usage for a single phone when the number of components in the emission densities are 16 for both models, is displayed in Table 5.1. Since the amount of weight sets in the NdHMM states are varying, the mean number, 9.44, is used in the overview. In Table 5.2 the results, in terms of accuracy, of the phone recognition experiments are shown. The relatively poor baseline performance is due to the fact that the dynamic features are not used. As mentioned the experiments have been conducted with only 39 models, not 48 as usual. This was because of the longer run-time of training 48 models

Table 5.1: Comparing the parameter usage in conventional HMM and NdHMM for a single phone. The conventional HMM uses 13 MFCCs and 16 gaussians per state, while the NdHMM employs 12 MFCCs and 16 dictionary components per state. The average number of maximum state duration for the NdHMM is 9.44. Note that this make the total parameter count to become a decimal, which might seem strange. This is however the average parameter usage for each phone.

HMM	
Means	16·13=208
Variances	16·13=208
Weights	16
Single state	432
States	3
Total	1296

NdHMM	
Dictionary	16·12=192
Weights	16·9.44=151.0
Energy	2 (μ_q and σ_q^2)
Single state	345.0
States	1
Total	345.0

Table 5.2: The accuracy of the models in the phone recognition experiments on the TIMIT database. Three experiments performed on each setup: 8, 16 and 32 components in output emission density. Two set of features were tried with the NdHMM: Sigmoid transformed MFCCs (MFCC-S) and Sigmoid transformed MFCCs with variance normalisation (MFCC-S-VN)

		Components		
System	Features	8	16	32
HMM	MFCC	46.7 %	48.0 %	48.8 %
NdHMM	MEL	-	36.9 %	-
NdHMM	MEL+VN	-	37.2 %	-
NdHMM	12MFCC+S	44.1 %	47.1 %	48.6 %
NdHMM	12MFCC+S+VN	44.9 %	47.3 %	48.9 %

in the current implementation. In order to assure that the system yields the expected performance gain when the number of models is increased to 48, the experiment with the 12MFCC+S+VN feature set and 16 dictionary components has also been run by training models for the 48 TIMIT phone set and reduce the result to 39 phones. This yielded a performance of 49.6%, which is an increase of about 2-2.5%. For the conventional HMM with 16 components in the GMMs the performance went from 48.0% to 50.2%, which is about the same increase in accuracy.

Because the Mel-filterbank output features have a dimension twice the size of the MFCCs, the run-time for these experiments were considerably longer. Hence, combined with the poor performance, these features were only tried with a single setting for the number of dictionary components. Table 5.2 shows that the proposed system performs at the same level as the conventional HMM when both systems use 32 components in the emission densities, while for 8 components the conventional HMM performs better. However, these results should be considered with the parameter usage in Table 3.2 in mind: the parameter usage of the NdHMM system with 32 components is approximately the same as the parameter usage of the conventional HMM with 8 components (688.08 vs 648).

5.5.4 Dynamic Features

The experiments that have been presented in this Chapter are all without the use of dynamic features. According to [2] the information contained in the dynamic features would give a relative reduction in error rate of about 20%. Hence, it is important for any speech recogniser to accommodate this information. To that end two approaches have been tried: adding the dynamic features to the MFCC vector before sigmoid transformation and concatenating surrounding static vectors before reducing the dimensionality.

The first approach is similar to how the dynamic features are used in the conventional HMM. Since the NdHMM demands non-negative features, which is not the case for the dynamic features, they were appended to the feature vectors before the sigmoid transform. The experiment yielding the best performance using this method was conducted with a set of 13 MFCCs (including C_0). These coefficients were extracted with 25 ms windows and 10 ms shifts. The Δ and $\Delta\Delta$ parameters were then added before the sigmoid transform was applied to each dimension. As for the experiments with only static features the best performance was achieved by normalising the feature vectors to have a sum equal to the C_0 coefficient, which was removed. The 38 dimension feature vectors were then presented to the NdHMM which yielded about 49% accuracy, which is marginally, but not significantly, better than using static information alone.

The second approach consist of concatenating a sequence of the static input vectors for each time frame. An equal amount of the preceding and the succeeding feature vectors are concatenated with the current frame. This results in a large dimensional input vector, which is reduced by principal component analysis (PCA) or linear discriminant analysis (LDA). Again the sigmoid transform needs to be applied in order for the non negativity constraint to be fulfilled. The procedure was also tested with the Mel-frequency features and a non-negative variant of the PCA algorithm [89]. The best performance was achieved by using 9 frames of the non-negative Mel-filterbank log-energies described in Section 5.5.3 yielding a total dimension of 234, which was reduced to 39 by non-negative PCA. Unfortunately, the performance was about 40% accuracy, which is significantly inferior to the performance using static information alone. If this is because of the non-negative constraint on the PCA/LDA or the combination of such features with the NdHMM is unknown.

5.6 Discussion

The experiments presented in Section 5.5.2 indicate that the proposed model is behaving as expected after the modifications have been done: the model seems to be able to learn different spectral components in the training data, while the time varying weights indeed creates time variation in the emission density of the states. Figure 5.6 and Figure 5.7 are nice examples of how the weights vary along with the duration counter. As an interesting point; notice that the weights for the phone /er/ in Figure 5.6 make one of the dictionary components unavailable after 5 frames. This component, and thereby the corresponding acoustic event, can therefore be assumed to only occur in the beginning of the realisations of the phone.

In Figure 5.8 both the transition matrix, including the initial state probabilities, and the energy distribution are displayed. The transition matrix is of the expected form with high values on the diagonal, which represents the self transitions of the states. The initial state probabilities are also as expected with only one non-zero component. In TIMIT all the sentences begins with /h#/ which is mapped to the silence model /sil/. The "energy" distributions also appear to comply with what was expected. At least for this small scale experiment the "energy" seems quite important for the discriminability of the NdHMM.

Considering the performance reported in Table 5.2 of the NdHMM, it seems that the NdHMM formulation successfully combines the advantages of NMF and HMM on the task. The NdHMM performs on the same level as the standard 3-state HMM, but with only a third of the number of parameters. When training an ASR system, the parameter count is always something that needs to be considered, as the size of the training database is never as large as desired. A high parameter count requires more training data in order for all the parameters to be trained properly. Thus, achieving the same performance by using only a third of the number

of parameters is an advantage.

Both the Mel-filterbank output and the MFCCs have been tried as features on the TIMIT phone recognition task, where the latter was made non-negative using a sigmoid transform. In the experiments, the NdHMM performs significantly better when using the sigmoid transformed MFCCs instead of the Mel-filterbank output. Thus, it might be reasonable to assume that the sigmoid transform does not hurt the performance significantly. In any case, based on the experiments presented, it is possible to conclude that the DCT in combination with the sigmoid transform increases the performance of the NdHMM. The influence of normalising features with regard to the variance is consistently positive for all the experiments.

The NdHMM has in this dissertation been used solely for phone recognition. However, as the model is able to generalise for unseen data, it may be considered for other areas of speech modelling or perhaps other pattern recognition problems.

5.6.1 Suggestions for Future Work

Although the model performs well on the experiments that are presented in this chapter, there are still some issues that need to be resolved. The most obvious is the lack of support for the dynamic features. In Section 5.5.4 two general approaches to accessing this problem were presented. Unfortunately, neither of the experiments were successful. The dynamic features contain important information for the phone recogniser. Thus, without the ability to accommodate such features, the use of the NdHMM is restricted. One reason for the deficiency of the standard approach of adding the dynamic features to the static feature vectors, may be ascribed to the fact that the NdHMM models the entire feature vector with the same dictionary, weights and energy distributions. It is probable that this makes

the NdHMM better suited for cases where all the dimensions have similar properties. When the feature vectors contains both $1^{\text{st}}$and 2^{nd}order delta coefficients in addition to the static features, there might be a mismatch. This could be avoided if the NdHMM supported parallel input streams, by using separate output distributions for each of the streams. The static feature vector, the 1^{st}order delta features and the 2nd order delta coefficients could then be modelled by individual streams. Since this change enforces major changes to the current implementation, it is left as a suggestion for future work.

Another possible weakness of the proposed model is the mismatch when modelling units with large variation in duration, like some of the phones that might vary from 10 ms to 500 ms. This mismatch is due to the dependency of the number of frames since the state was entered. Consider for instance two realisations of a unit with durations of 3 and 9 frames. Assuming that the maximum duration in the model is higher than 9, the 3 frame realisation would be modelled by the same emission densities as the 3 first frames of the 9 frame realisation. The entire first realisation is thereby modelled by the same emission densities as the first third of the second realisation. The intuitively best way of modelling these two realisations would be to align the realisations so that the first frame of the first realisation is aligned with one of the first 3 frames of the second realisation, the second frame with one of the 3 frames in the middle, and finally the last frame with one of the last 3 frames. A remedy to this problem could be to change the duration counter to be able to increment with more than one, i.e. allow skips in the duration counter. In addition the counter should probably also have the permission to not increment at all, and thus let the model use the same emission densities for more than one consecutive frame. Note that according to the equations presented in Section 5.3 this change can be implemented by changing the probability distribution $P(d_{t+1}|q_t, d_t, q_{t+1})$ to have more than one non-zero component. Although the resulting model would have an increase in complexity,

the maximum duration of the states could probably be decreased because of the re-use of emission densities. Thus, the total number of parameters would not necessarily increase significantly. Nevertheless, the implementation of this change needs to be performed while carefully evaluating its impact on the system. This is left as a suggestion for future work.

Finally, it might also be worthwhile to perform a study concerning the choice of features. The model is designed for non-negative features, therefore it might be advantageous to use features that are naturally non-negative, eliminating the need to apply a transformation which influence on the performance is unknown.

5.7 Concluding Summary

In this Chapter the N-HMM has been reviewed and several modifications have been suggested in order for it to be useful for ASR. The main modification was to make the weights of the states dependent of the number of frames since the state was entered. This modification allowed the model to generalise to unseen data, but came with a cost: the freedom of modelling each frame with individual weights was removed. Instead the weights are reused for every visit to the state. In order for the model to properly model the variation of the input signal, the generative process had to be slightly modified: instead of drawing both a dictionary component and a frequency bin, the model is only allowed to draw from one dictionary component for the entire frame. The experiments that have been performed with the modified model, the NdHMM, show that the model still is able to discover variations in the data that might represent different contexts or sub-phonetic units. Even though the model performs at the same level as the baseline with fewer parameters, the work on the model cannot be considered as completed. Several topics have been suggested for further investigation before the model may be used for ASR.

Chapter 6

Concluding Summary

In this dissertation the HMM has been investigated. The HMM has been widely used the past decades and good performance has been reported. However, there is still need for improvements in order to achieve satisfactory performance in the more difficult ASR tasks. To that end, three alternative HMM based phone recognition systems have been proposed and tested. The first system expanded the topology of the phone HMMs in order for it to explicitly model different variations of the phone realisations. In addition to the conventional 3 state left-to-right HMM, the system also allowed parallel variants with 1, 2 or 4 states. To train the model properly, despite the significant increase of parameters, a specially adopted technique was proposed. A set of Acoustic Sub-Word Units (ASWUs) was defined by first performing an automatic acoustic segmentation and then cluster the acoustic segments. The ASWUs were then given a linguistic interpretation in order to link each to one state in the HMM. Finally, the states were constructed as a linear combination of all the ASWUs. The phone recognition experiments showed performance comparable to the baseline model, albeit with a significant increase of parameter usage.

Also for the second system the idea of an expanded topology was central. However, instead of using a fixed topology for all the phone models, the topologies were customised for each phone individually. A data driven pronunciation variation modelling (PVM) algorithm was used to automatically derive the topologies. The phone realisations were modelled as a sequence of the ASWUs used in the first system, and thus the ASWUs were employed as input to the PVM algorithm. Each pronunciation variation from the PVM algorithm was used to construct one path in the HMM. The phone recognition experiments again proved the system to be inferior to the 3-state baseline system in terms of parameter usage. In addition the system yielded the best performance when only a single pronunciation variation was used for each phone, thereby impairing the hypothesis of the extended topology which both the two first systems were built on.

In the third and final system that was proposed, the focus was moved from the topology to the emission densities. A system previously employed for source separation by Mysore [81], the non-negative HMM (N-HMM), was examined. The N-HMM is a model that combines the pattern recognition capabilities of non-negative matrix factorisation (NMF) with the temporal modelling power of the HMM. In the N-HMM, the emission densities of the states are dependent on the absolute time, which restricts the model from being generalisable to unseen data, a requirement for it to be used in ASR. A remedy was proposed by instead making the emission densities dependent on the number of frames since the state was entered. The modified model was called NdHMM and tested on the TIMIT database for phone recognition, yielding comparable performance with the baseline system. In addition to the promising recognition results, it was shown that the model also used significantly less parameters compared to the conventional HMM. Several topics have been suggested for future research, like the compatibility with the dynamic parameters. Nevertheless, the experiments show an interesting system which might have a future, both in ASR and similar fields of research.

The HMM has been a natural part of most ASR systems the past decades. It is a powerful, yet simple and efficient model. Through the experiments carried out in this dissertation, the power of the conventional HMM have been demonstrated. The efforts in using a topology believed to be better suited for the phones, did not result in any improvements of the recognition accuracy. Neither did the NdHMM, although the results there were promising in terms of parameter usage and several suggestions for possible improvements were made.

Appendix A

The TIMIT Database

All systems in this dissertation were evaluated on the DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus [90]. TIMIT includes a manual transcription on the phone level, which makes it convenient to perform phone recognition experiments. The corpus consist of read speech spoken by 630 speakers from 8 major dialect regions of the United States. Each speaker read 10 sentences, totaling to 6300 read sentences. Of the 10 sentences 2 were spoken by all the speakers, and were therefore not used in this dissertation. The corpus is divided into a test set and training set, where no sentence text appear in both sets. The test set consist of 1344 sentences spoken by 168 speakers. In this dissertation the data of 50 speakser (400 sentences) in the training set were used as developement data. The list of the 50 speakers is given in Table A.1.

A.1 Phone Sets

The TIMIT phone label set consist of 61 different phones. In [91] a mapping of the 61 phones to 48 phones were proposed. Among the 48 phones, 7 groups were defined where within-group confusions were not counted, resulting in a test set of 39 phones. The command files for performing the mapping with the HTK tool HLEd ([46]) is given in Table A.2 and Table A.3. With the exception of the experiments with the NdHMM in Chapter 5 where only the set containing 39 phones was used, the experiments in this dissertation were performed by training models for 48 phones and only count confusions between the 39 phones during testing.

Table A.1: List of speakers in the development set.

mbsb0	mvrw0	mtwh1	mrlj1	mklr0
mwrp0	mtcs0	msdb0	mrmg0	mmlm0
mpar0	mbcg0	mkdb0	mpfu0	fpls0
mjra0	mrem0	fclt0	mntw0	mmea0
mrdm0	mrlk0	mkrg0	mkag0	mrmh0
mtmn0	fjrb0	mtml0	mcxm0	mmpm0
fklh0	fnkl0	mmws1	fceg0	mses0
mwre0	fbcg1	mmws0	mtlc0	msah1
mrpc1	mmdg0	mkdd0	mtpr0	mrre0
mter0	mtkd0	mtab0	mejs0	fmbg0

A.2 Confidence Intervals

The test set can be viewed as a collection of segments, where each segment is either recognised correctly or incorrectly during phone recognition. By assuming that:

Table A.2: HHEd command file for mapping from 61 to 48 symbols

```
DE q
RE m em
RE n nx
RE ng eng
RE hh hv
RE uw ux
RE ax ax-h
RE er axr
RE sil h# pau
RE vcl bcl dcl gcl
RE cl pcl tcl kcl
```

Table A.3: HHEd command file for mapping from 61 to 39 symbols

```
DE q
RE sh zh
RE m em
RE n en nx
RE ng eng
RE hh hv
RE l el
RE aa ao
RE uw ux
RE ah ax ax-h
RE ih ix
RE er axr
RE sil h# pau epi bcl dcl gcl pcl tcl kcl
```

1. the probability of recognising a segment correctly is equal for all the segments,
2. the probability of recognising a given segment correctly is independent of the other segments,

the phone recognition process can be viewed as a series of Bernoulli trials. Both of the two assumptions are incorrect, but viewing the number of phone errors as having a binomial distribution simplifies the calculation of confidence intervals for the true PER. By denoting the estimate of PER as $\hat{p}$ the confidence intervals can be found by[92]:

$$\hat{p} \pm z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \tag{A.1}$$

where α is the desired significance level, $z_{\alpha/2}$ is the corresponding tail of the standard normal distribution and n is the number of trials. This method approximates the binomial distribution with a normal distribution, and for this approximation to be considered reliable, n should to be "sufficiently large" while p should not be close to 0 or 1. In [92] $np \geq 5$ and $n(1-p) \geq 5$ is used as a requirement for the validity of the method. In the TIMIT test set there are 50754 segments, so both conditions are met as long as p is not close to 0 or 1. In Figure A.1 the size of the confidence interval is plotted as a function of PER for three values of the significance level α. Note that the approximation of the PER to be binomial distributed is questionable. However, the confidence interval sizes gives an indication of the significance level of the phone recognition results.

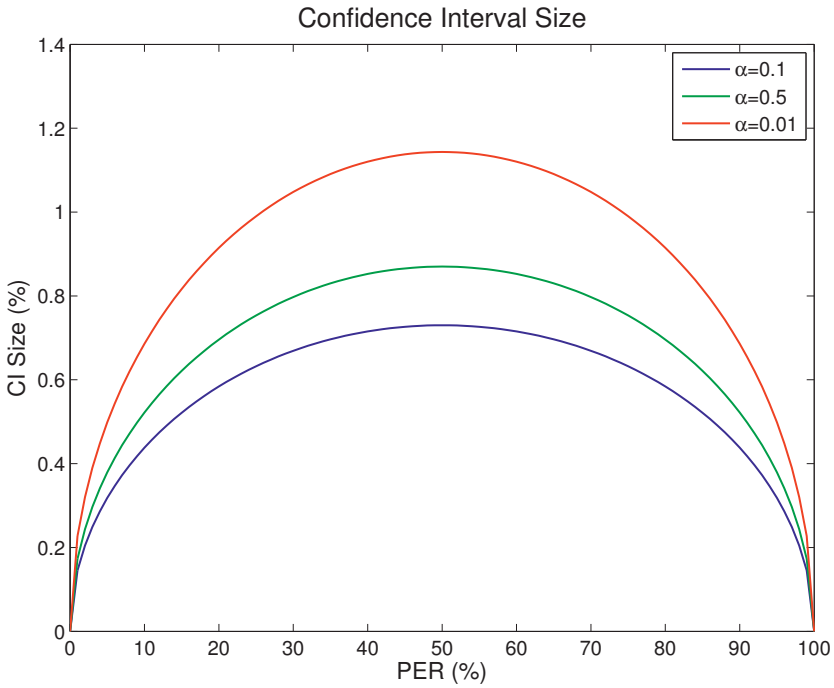


Figure A.1: Confidence interval sizes as a function of PER for three values of the significance level α.

Appendix B

Acoustic Segmentation Statistics

In Chapter 3 the acoustic segments were compared to the manual phonetic labels of TIMIT. This resulted in a linguistic interpretation of the ASWUs. However, with different choices for the over-segmentation factor, the assignment changes. In Figure B.1 to B.10 the distribution of the number of segments assigned to the realisations of each phone is shown. See Section 3.4.1 for a detailed explanation.

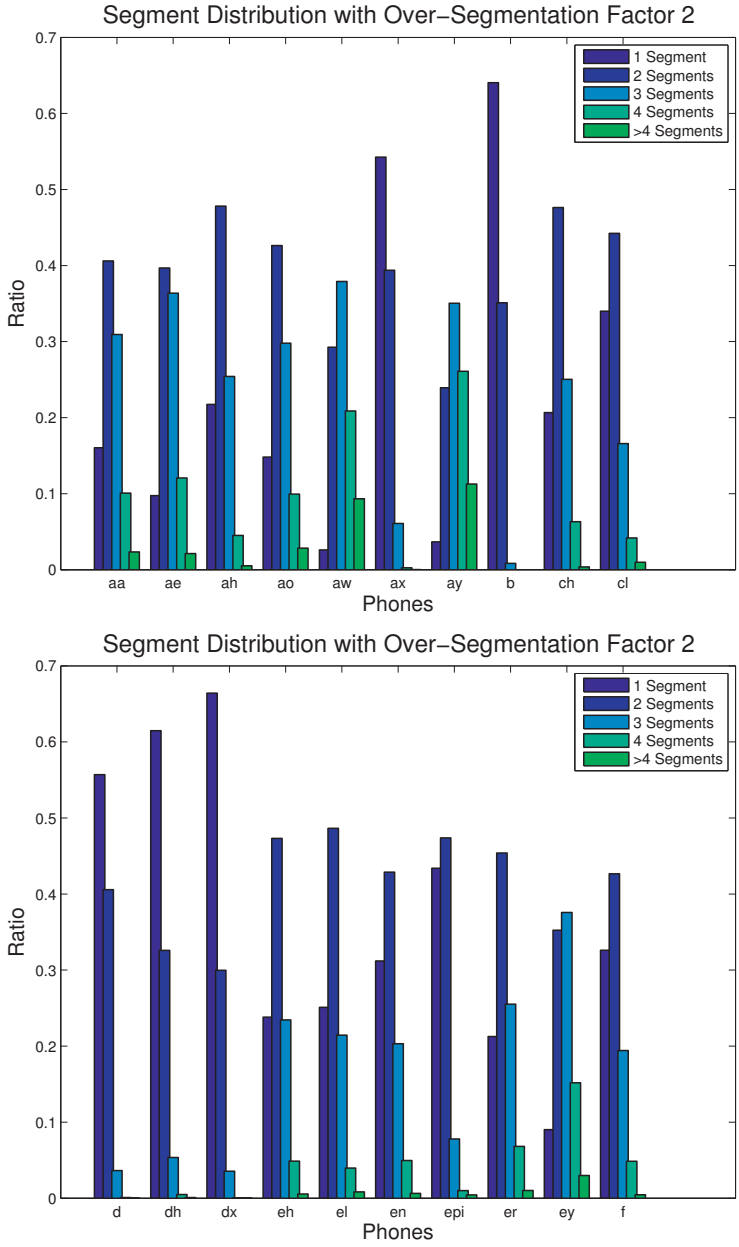


Figure B.1: Distribution of number of segments per phone when the over-segmentation factor is 2.

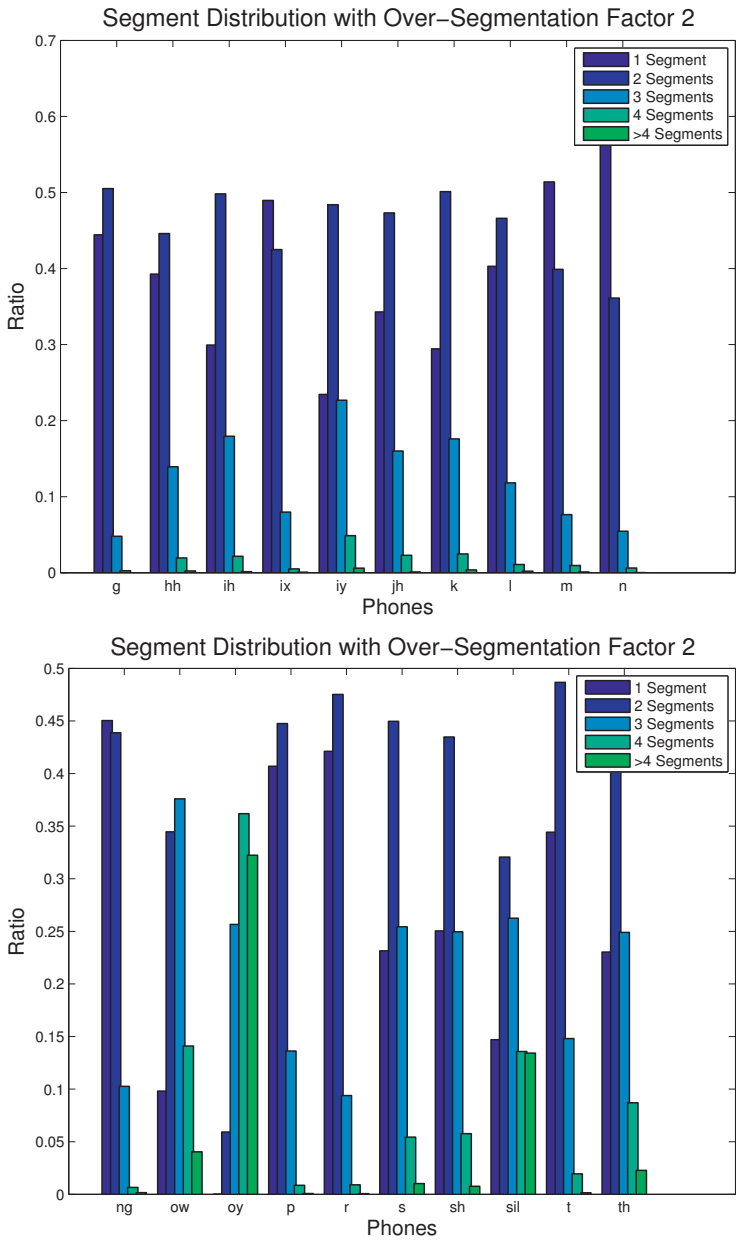


Figure B.2: Distribution of number of segments per phone when the over-segmentation factor is 2.

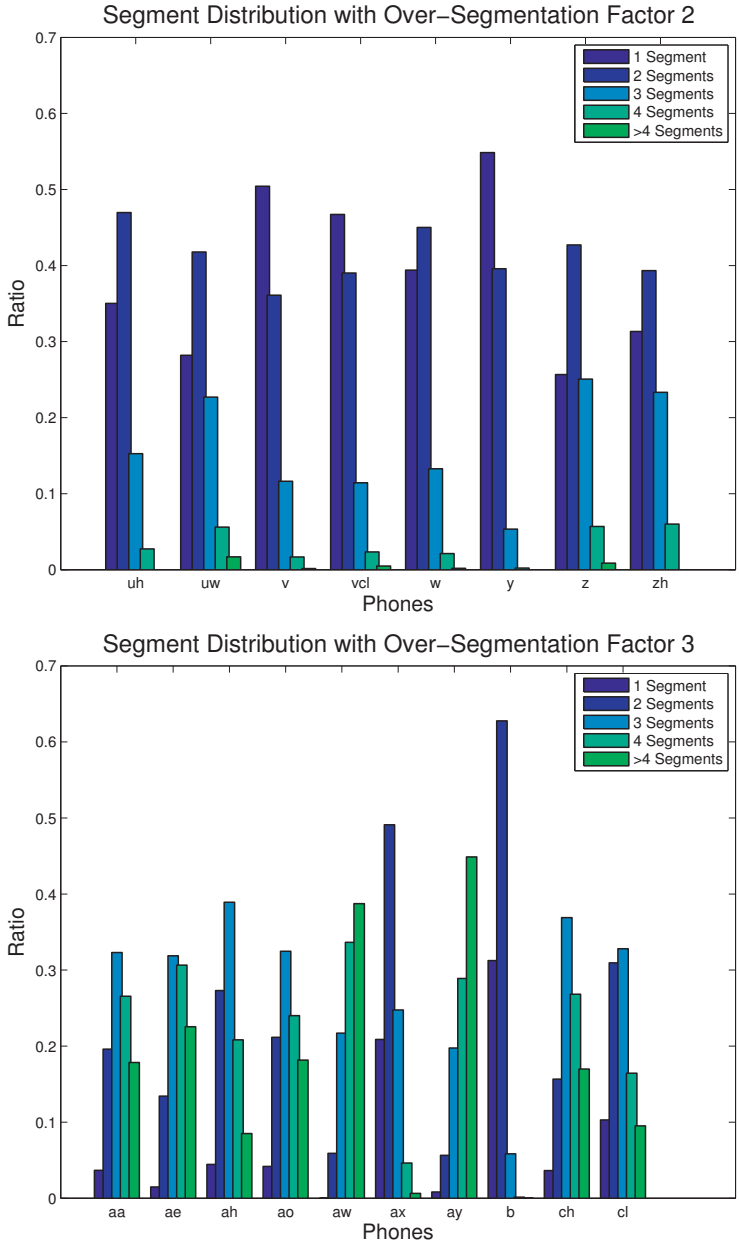


Figure B.3: Distribution of number of segments per phone when the over-segmentation factor is 2 (upper) and 3 (lower).

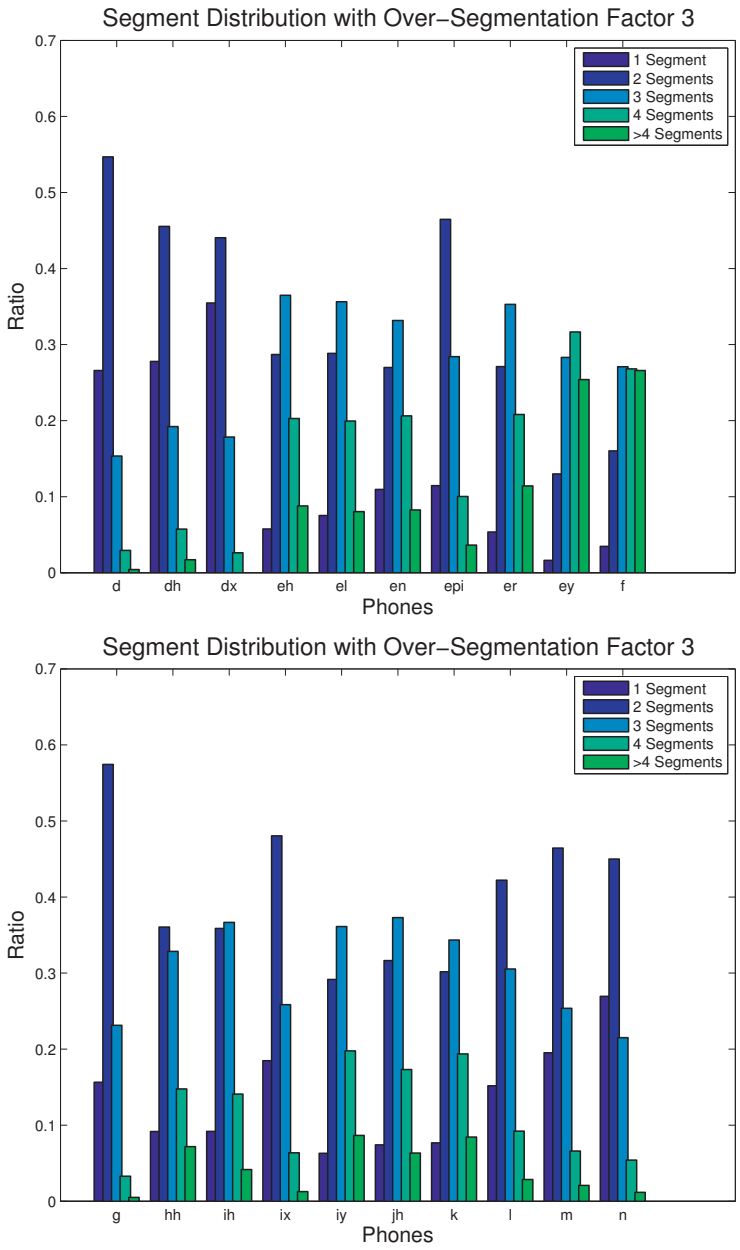


Figure B.4: Distribution of number of segments per phone when the over-segmentation factor is 3.

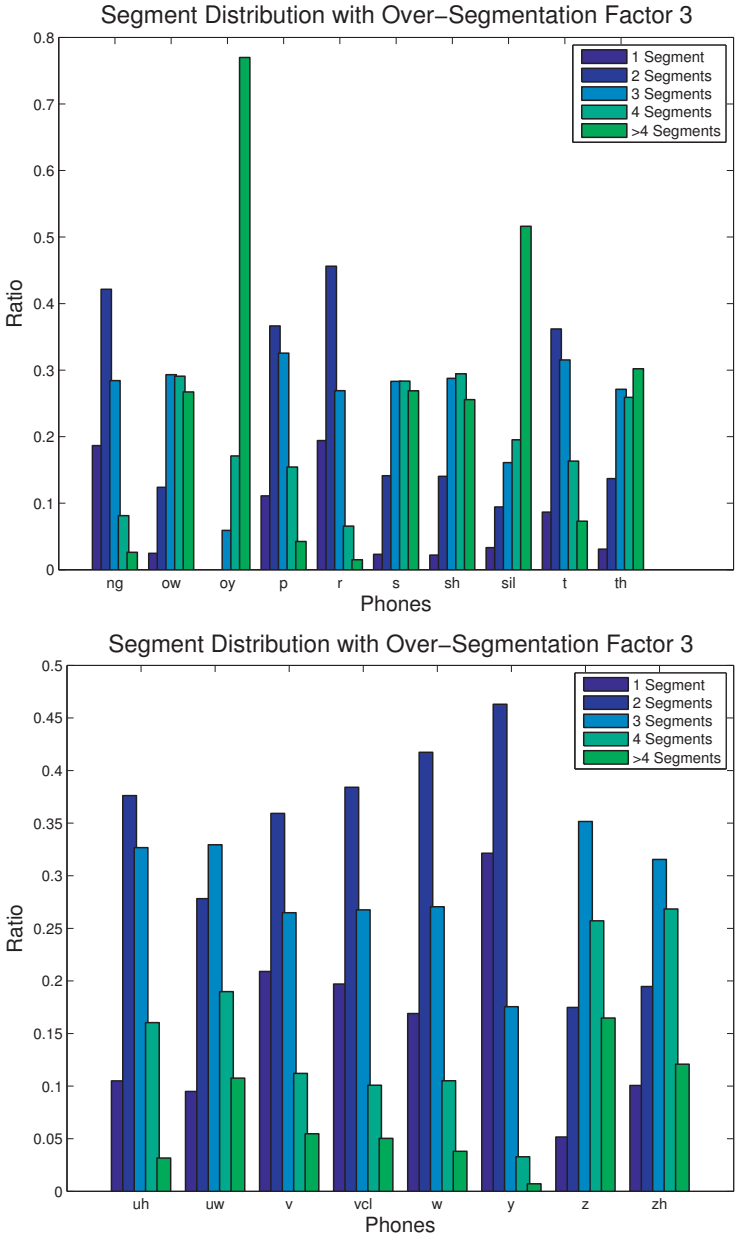


Figure B.5: Distribution of number of segments per phone when the over-segmentation factor is 3.

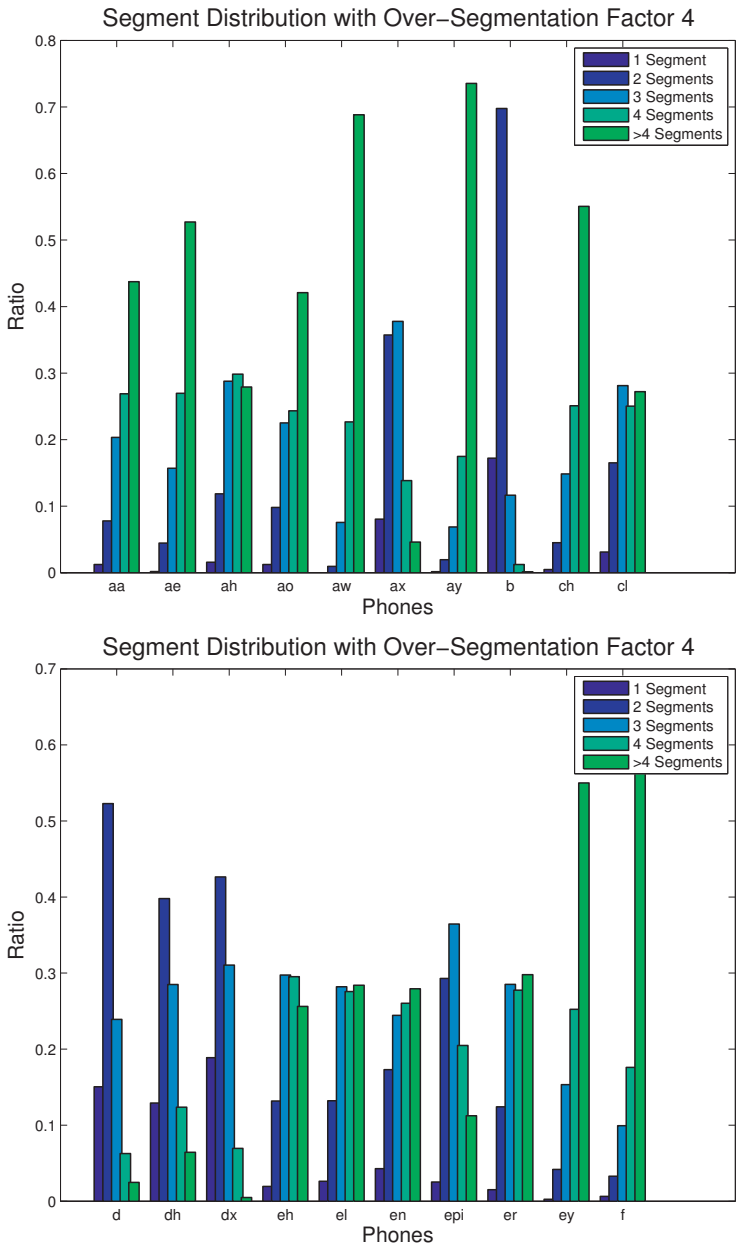


Figure B.6: Distribution of number of segments per phone when the over-segmentation factor is 4.

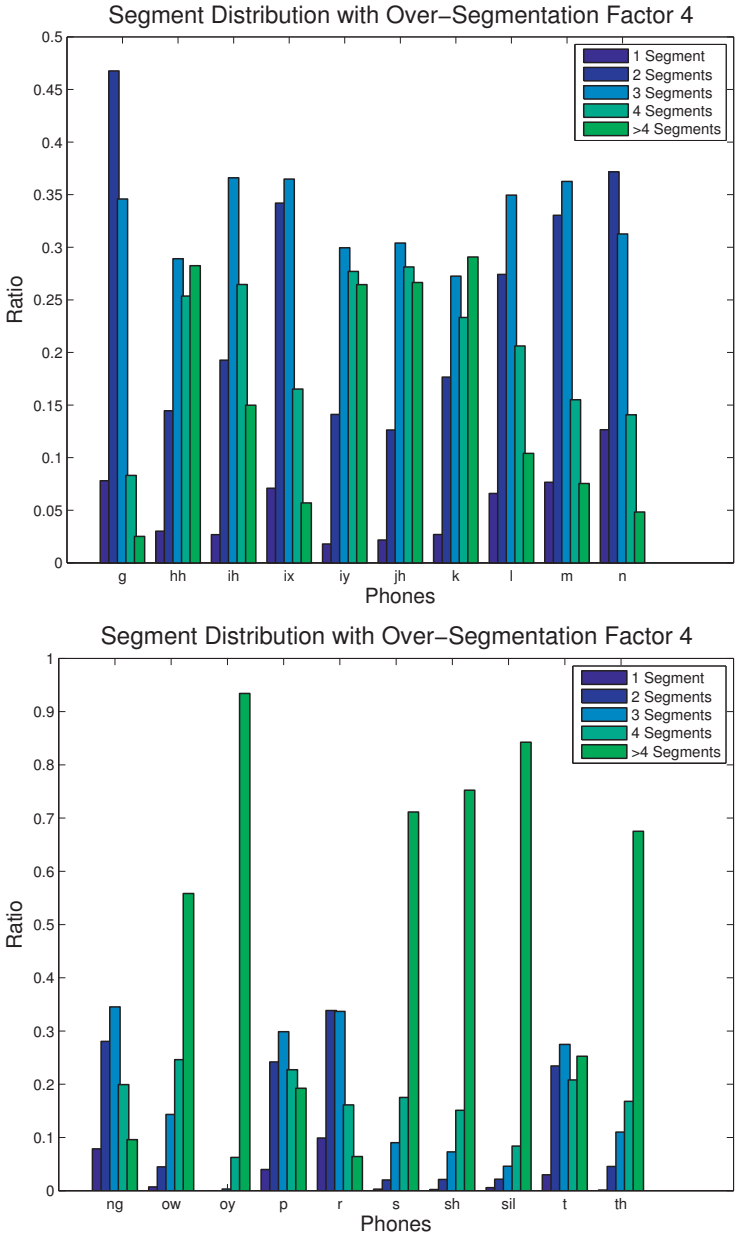


Figure B.7: Distribution of number of segments per phone when the over-segmentation factor is 4.

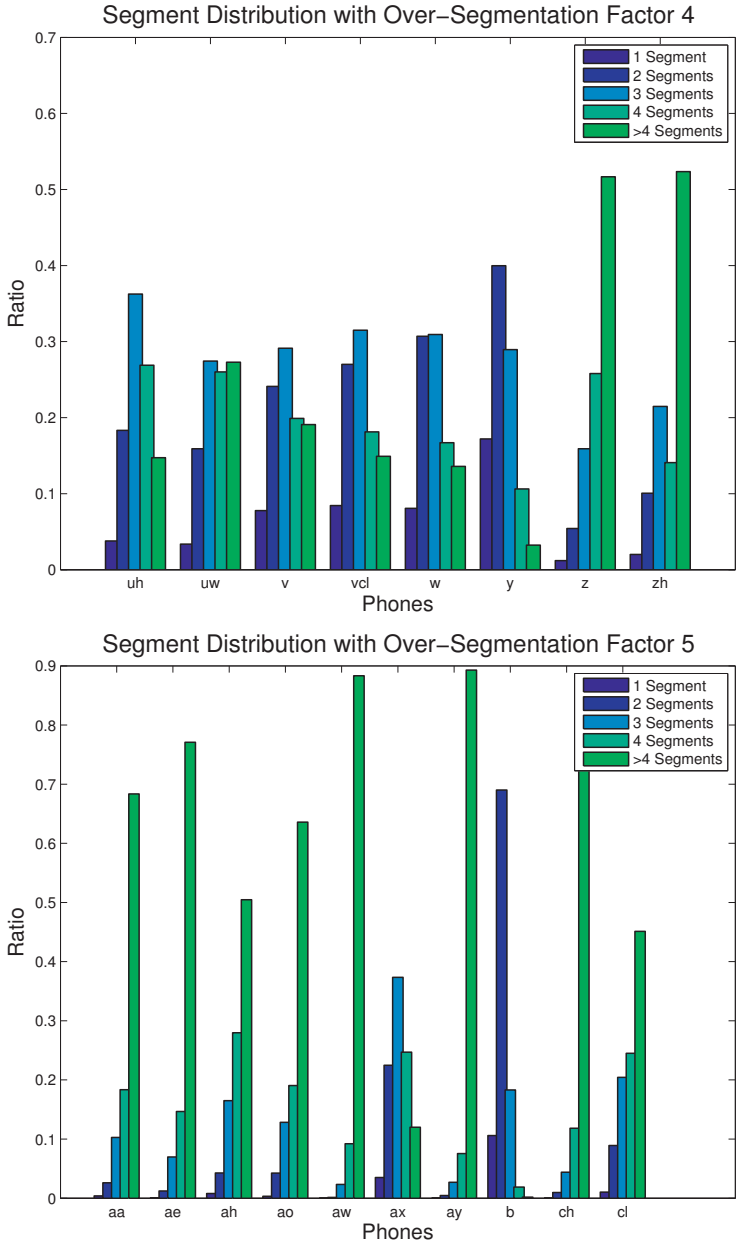


Figure B.8: Distribution of number of segments per phone when the over-segmentation factor is 4 (upper) and 5 (lower).

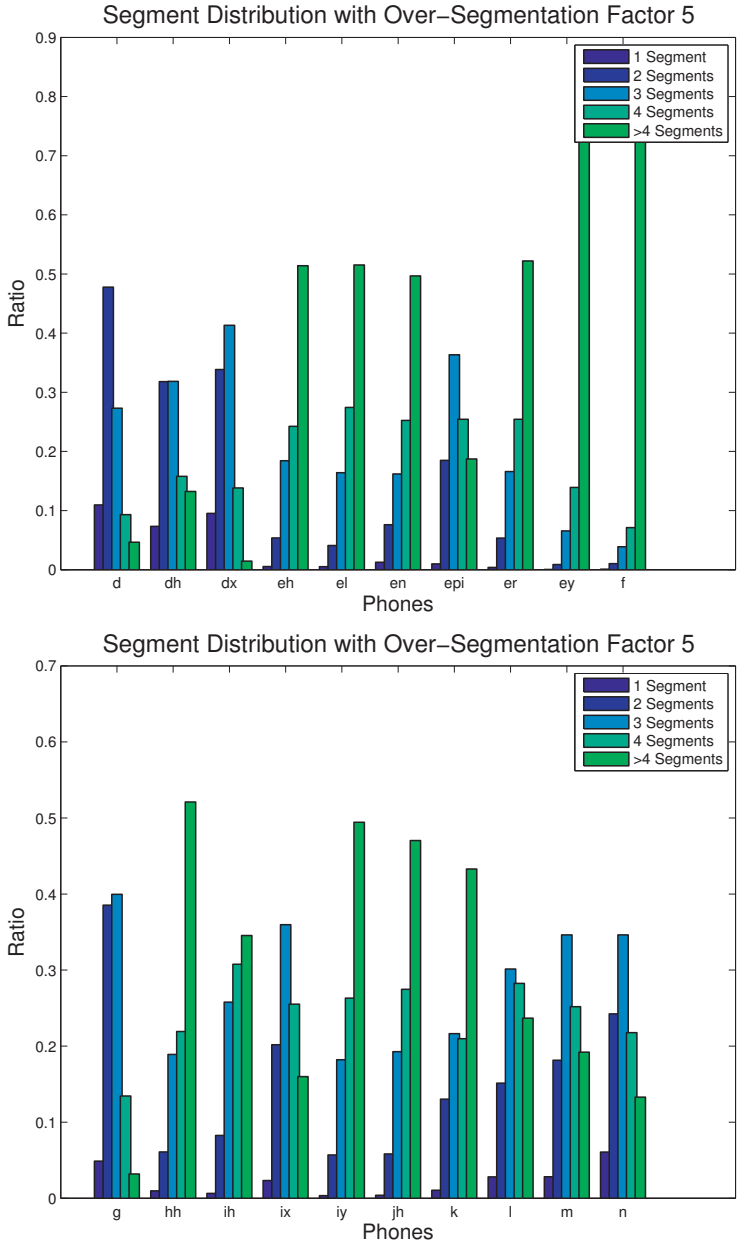


Figure B.9: Distribution of number of segments per phone when the over-segmentation factor is 5.

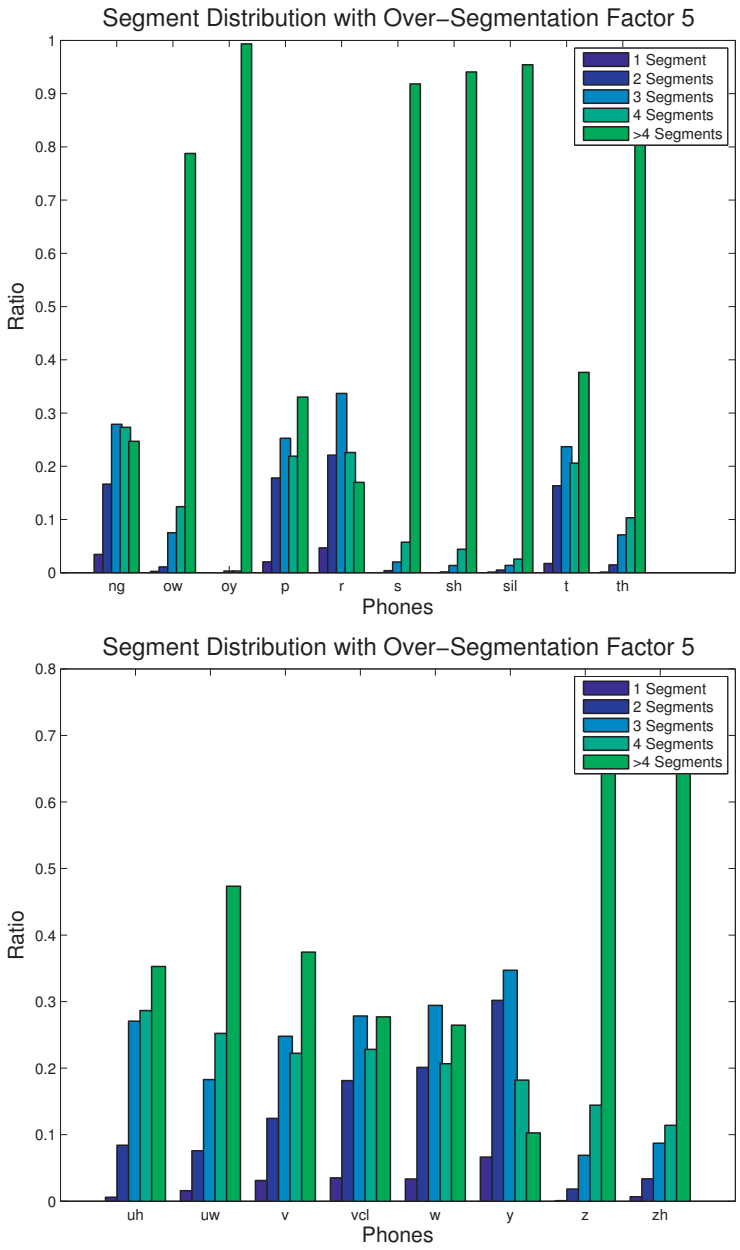


Figure B.10: Distribution of number of segments per phone when the over-segmentation factor is 5.

Bibliography

[1] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, pp. 357 – 366, aug 1980.

[2] X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing*. Prentice Hall PTR, 2001.

[3] K. Elenius and M. Blomberg, "Effects of emphasizing transitional or stationary parts of the speech signal in a discrete utterance recognition system," in *Acoustics, Speech, and Signal Processing, IEEE International Conference*, vol. 7, may 1982, pp. 535 – 538.

[4] S. Furui, "Cepstral analysis technique for automatic speaker verification," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 29, no. 2, pp. 254 – 272, apr 1981.

[5] ——, "Speaker-independent isolated word recognition using dynamic features of speech spectrum," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 34, no. 1, pp. 52 – 59, feb 1986.

[6] S. Furui, "On the role of spectral transition for speech perception," *Acoustical Society of America Journal*, vol. 80, pp. 1016–1025, Oct.

1986.

[7] S. M. Ross, *Introduction to Probability Models, 9th edition.* Academic Press, 2007.

[8] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, 1989, pp. 257–286.

[9] J. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," ICSI, Berkley CA, Tech. Rep., 1998.

[10] K. Vertanen, "An overview of discriminative training for speech recognition," University of Cambridge, UK, Tech. Rep., 2004.

[11] M. Ostendorf, V. Digalakis, and O. Kimball, "From HMM's to segment models: a unified view of stochastic modeling for speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 4, no. 5, pp. 360 –378, sep 1996.

[12] J. Bilmes, "What HMMs can do," University of Washington, Dept of EE, Tech. Rep., 2002.

[13] X. Huang and M. Jack, "Hidden Markov modelling of speech based on a semicontinuous model," *Electronics Letters*, vol. 24, no. 1, pp. 6 –7, jan 1988.

[14] X. D. Huang, H. W. Hon, and K. F. Lee, "Large-vocabulary speaker-independent continuous speech recognition with semi-continuous hidden Markov models," in *Proceedings of the workshop on Speech and Natural Language*, ser. HLT '89. Stroudsburg, PA, USA: Association for Computational Linguistics, 1989, pp. 276–279. [Online]. Available: http://dx.doi.org/10.3115/1075434.1075480

[15] M. Russell and W. Holmes, "Linear trajectory segmental HMMs," *Signal Processing Letters, IEEE*, vol. 4, no. 3, pp. 72 –74, march 1997.

[16] K. Tokuda, H. Zen, and T. Kitamura, "Trajectory modeling based on HMMs with the explicit relationship between static and dynamic features," in *Proceedings of Eurospeech*, 2003.

[17] H. Zen, K. Tokuda, and T. Kitamura, "A Viterbi algorithm for a trajectory model derived from HMM with explicit relationship between static and dynamic features," in *ICASSP*, vol. 1, may 2004, pp. I – 837–40.

[18] M. Russel and R. Moore, "Explicit modeling of state occupancy in hidden Markov models for automatic speech recognition," in *ICASSP*, 1985.

[19] S. E. Levinson, "Continuously variable duration hidden Markov models for automatic speech recognition," *Computer Speech and Language*, vol. 1, no. 1, pp. 29–45, Mar. 1986. [Online]. Available: http://dx.doi.org/10.1016/S0885-2308(86)80009-2

[20] A. Bonafonte, X. Ros, and J. B. Marifio, "An efficient algorithm to find the best state sequence in HSMM," in *EUROSPEECH'93*, 1993.

[21] J. Pylkkönen and M. Kurimo, "Duration modeling techniques for continuous speech recognition," in *ICSLP*, 2004.

[22] A. Bonafonte, J. Vidal, and A. Nogueiras, "Duration modeling with expanded HMM applied to speech recognition," in *Spoken Language Proceedings, Fourth International Conference on*, vol. 2, oct 1996, pp. 1097 –1100.

[23] A. Poritz, "Linear predictive hidden Markov models and the speech signal," in *Acoustics, Speech, and Signal Processing, IEEE International Conference*, vol. 7, may 1982, pp. 1291 – 1294.

[24] L. Rabiner and B.-H. Juang, *Fundamentals of speech recognition*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[25] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer Academic Publishers, 1993.

[26] H. Bourlard and N. Morgan, "Hybrid HMM/ANN systems for speech recognition: Overview and new research directions," *In adaptive processing of sequences and data structures, ser. lecture notes in artificial intelligence*, vol. 1387, pp. 389–417, 1998.

[27] E. Trentin and M. Gori, "A survey of hybrid ANN/HMM models for automatic speech recognition," *Neurocomputing*, vol. 37, pp. 91 – 126, 2001. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231200003088

[28] A.-R. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *INTERSPEECH*, 2010, pp. 2846–2849.

[29] A.-R. Mohamed, T. Sainath, G. Dahl, B. Ramabhadran, G. Hinton, and M. Picheny, "Deep belief networks using discriminative features for phone recognition," in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*, may 2011, pp. 5060 –5063.

[30] A.-R. Mohamed, G. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 14 –22, jan. 2012.

[31] L. Bahl, P. Brown, P. de Souza, and M. Picheny, "Acoustic Markov models used in the Tangora speech recognition system," in *Acoustics, Speech, and Signal Processing, International Conference on*, apr 1988, pp. 497 –500 vol.1.

[32] L. Bahl, P. Brown, P. de Souza, R. Mercer, and M. Picheny, "A method for the construction of acoustic Markov models for words," *Speech and Audio Processing, IEEE Transactions on*, vol. 1, no. 4, pp. 443 –452, oct 1993.

[33] C.-H. Lee, F. Soong, and B.-H. Juang, "A segment model based approach to speech recognition," in *Acoustics, Speech, and Signal Processing, International Conference on*, apr 1988, pp. 501 –541 vol.1.

[34] C.-H. Lee, B.-H. Juang, F. Soong, and L. Rabiner, "Word recognition using whole word and subword models," in *Acoustics, Speech, and Signal Processing, International Conference on*, may 1989, pp. 683 –686 vol.1.

[35] T. Svendsen, K. K. Paliwal, E. Harborg, and P. O. Husøy, "An improved sub-word based speech recognizer," *Acoustics, Speech, and Signal Processing, International Conference on*, pp. 108 – 111 vol.1, 1989.

[36] J. Glass and V. Zue, "Detection and recognition of nasal consonants in American English," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, vol. 11, apr 1986, pp. 2767 – 2770.

[37] J. R. Glass and V. Zue, "Signal representation for acoustic segmentation," in *Proc. of the First Australian Conference on Speech Science and Technology*, 1986, pp. 124–129.

[38] J. Glass and V. Zue, "Multi-level acoustic segmentation of continuous speech," in *Acoustics, Speech, and Signal Processing, International Conference on*, apr 1988, pp. 429 –432 vol.1.

[39] T. Svendsen and F. K. Soong, "On the automatic segmentation of speech signals," *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, 1987.

[40] K. Paliwal, "Lexicon-building methods for an acoustic sub-word based speech recognizer," in *Acoustics, Speech, and Signal Processing, International Conference on*, apr 1990, pp. 729 –732 vol.2.

[41] M. Bacchiani, M. Ostendorf, Y. Sagisaka, and K. Paliwal, "Unsupervised learning of non-uniform segmental units for acoustic modeling in speech recognition," in *IEEE ASR Workshop*, 1995.

[42] ——, "Design of a speech recognition system based on acoustically derived segmental units," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, vol. 1, may 1996, pp. 443 –446 vol. 1.

[43] T. Holter and T. Svendsen, "Combined optimisation of baseforms and model parameters in speech recognition based on acoustic subword units," in *Automatic Speech Recognition and Understanding ASRU, Proceedings on*, dec 1997, pp. 199 –206.

[44] H. Gish and K. Ng, "A segmental speech model with applications to word spotting," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, vol. 2, april 1993, pp. 447 –450 vol.2.

[45] T. Fukada, M. Bacchiani, K. Paliwal, and Y. Sagisaka, "Speech recognition based on acoustically derived segment units," in *Spoken Language, Proceedings, Fourth International Conference on*, vol. 2, oct 1996, pp. 1077 –1080 vol.2.

[46] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK Book, version 3.4.* Cambridge, UK: Cambridge University Engineering Department, 2006.

[47] S. B. Jouvet, "Parameter tying for flexible speech recognition," 1996.

[48] M. Eskenazy, "Trends in speaking styles research," in *Proceedings of Eurospeech*, 1993, pp. 501–509.

[49] W. Labov, *Sociolinguistic Patterns*. University of Pennsylvania Press, 1972.

[50] J.-C. Junqua, S. Fincke, and K. Field, "The lombard effect: a reflex to better communicate with others in noise," in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, vol. 4, mar 1999, pp. 2083 –2086 vol.4.

[51] T. S. Polzin and A. Waibel, "Pronunciation variations in emotional speech," in *In: Proc. of the ESCA Workshop 'Modeling Pronunciation Variation for Automatic Speech Recognition'*, 1998.

[52] H. Strik and C. Cucchiarini, "Modeling pronunciation variation for ASR: A survey of the literature," *Speech Communication*, vol. 29, no. 2-4, pp. 225 – 246, 1999.

[53] H. Strik, "Pronunciation adaptation at the lexical level," in *Proceedings ISCA ITRW Workshop Adaptation Methods for Speech Recognition, Sophia Antipolis, France on CD-ROM*, 2001, pp. 123–130.

[54] M. Wester, "Pronunciation modeling for ASR - knowledge-based and data-derived methods," *Computer Speech and Language*, vol. 17, no. 1, pp. 69 – 85, 2003.

[55] I. Amdal and E. Fosler-Lussier, "Pronunciation variation modeling in automatic speech recognition," in *Telektronikk*, vol. 2, 2003, pp. 70–92.

[56] T. Svendsen, "Pronunciation modeling for speech technology," in *Signal Processing and Communications*, dec. 2004, pp. 11 – 16.

[57] M. Wester, J. M. Kessens, and H. Strik, "Pronunciation variation in ASR: Which variation to model," in *ICSLP, Beijing*, 2000, pp. 488–491.

[58] S. Schaden, "Rule-based lexical modelling of foreign-accented pronunciation variants," in *Proceedings of EACL*, 2003.

[59] K. Bartkova and D. Jouvet, "On using units trained on foreign data for improved multiple accent speech recognition," *Speech Communication*, vol. 49, no. 10–11, pp. 836 – 846, 2007.

[60] J. Humphries and P. Woodland, "Using accent-specific pronunciation modelling for improved large vocabulary continuous speech recognition," in *Proceedings of Eurospeech*, 1997.

[61] M. Bacchiani and M. Ostendorf, "Joint lexicon, acoustic unit inventory and model design," *Speech Communication*, vol. 29, no. 2-4, pp. 99 – 114, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167639399000333

[62] N. Cremelie and J.-P. Martens, "In search of better pronunciation models for speech recognition," *Speech Communication*, vol. 29, no. 2–4, pp. 115 – 136, 1999.

[63] I. Amdal, F. Korkmazskiy, and A. Surendran, "Data-driven pronunciation modelling for non-native speakers using association strength between phones," in *Proc. ISCA ITRW ASR*, 2000, pp. 85–90.

[64] ——, "Joint pronunciation modelling of non-native speakers using data-driven methods," in *ICSLP*, 2000, pp. 622–625.

[65] Q. Yang, J. pierre Martens, P.-J. Ghesquiere, and D. V. Compernolle, "Pronunciation variation modeling for ASR: Large improvements are possible but small small ones are likely to achieve," in *Proc. PMLA*, 2002, pp. 123–128.

[66] S. Goronzy and R. Kompe, "Generating non-native pronunciation variants for lexicon adaptation," *Speech Communication*, vol. 42, pp. 109–123, 2004.

[67] A. Raux, "Automated lexical adaptation and speaker clustering based on pronunciation habits for non-native speech recognition," in *ICSLP*, 2004.

[68] C. V. Bael, L. Boves, H. van den Heuvel, and H. Strik, "Automatic phonetic transcription of large speech corpora," *Computer Speech & Language*, vol. 21, no. 4, pp. 652 – 668, 2007.

[69] M. Adda-Decker and L. Lamel, "Pronunciation variants across system configuration, language and speaking style," *Speech Communication*, vol. 29, 1999.

[70] T. Fukada, T. Yoshimura, and Y. Sagisaka, "Automatic generation of multiple pronunciations based on neural networks and language statistics," *Speech Communication*, vol. 27, pp. 63–73, 1999.

[71] G. Lehtinen and S. Safra, "Generation and selection of pronunciation variants for a flexible word recognizer," in *Modeling Pronunciation Variation for Automatic Speech Recognition*, 1998, pp. 67–72.

[72] T. Sloboda and A. Waibel, "Dictionary learning for spontaneous speech recognition," in *ICSLP 1996*, 1996, pp. 2328–2331.

[73] E. Fosler-lussier and N. Morgan, "Effects of speaking rate and word frequency on conversational pronunciations," in *ESCA Tutorial and Research Workshop on Modeling Pronunciation Variation for Automatic Speech Recognition*, 1999, pp. 35–40.

[74] M. Riley, W. Byrne, M. Finke, S. Khudanpur, A. Ljolje, J. McDonough, H. Nock, M. Saraclar, C. Wooters, and G. Zavaliagkos, "Stochastic pronunciation modelling from hand-labelled phonetic corpora," *Speech Communication*, vol. 29, no. 2–4, pp. 209 – 224, 1999.

[75] Q. Yang and J.-P. Martens, "On the importance of exception and cross-word rules for the data-driven creation of lexica for ASR," in *Proceedings IEEE ProRisk (Veldhoven)*, 2000.

[76] J. M. Kessens, C. Cucchiarini, and H. Strik, "A data-driven method for modeling pronunciation variation," *Speech Communication*, vol. 40, no. 4, pp. 517 – 534, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167639302001504

[77] T. Holter, "Maximum likelihood modelling of pronunciation in automatic speech recognition," Ph.D. dissertation, The Norwegian University of Science and Technology, 1997.

[78] T. Holter and T. Svendsen", "Maximum likelihood modelling of pronunciation variation," *Speech Communication*, vol. 29, no. 2-4, pp. 177 – 191, 1999.

[79] I. Amdal, T. Holter, and T. Svendsen, "Maximum likelihood pronunciation modelling of Norwegian natural numbers for automatic speech recognition," in *Norwegian Signal Processing Symposium (NORSIG)*, 1999.

[80] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999. [Online]. Available: http://dx.doi.org/10.1038/44565

[81] G. J. Mysore, "A non-negative framework for joint modeling of spectral structure and temporal dynamics in sound mixtures," Ph.D. dissertation, Standford University, 2010.

[82] G. Mysore, P. Smaragdis, and B. Raj, "Non-negative hidden Markov modeling of audio with application to source separation," in *Latent Variable Analysis and Signal Separation*, ser. Lecture Notes in Computer Science, V. Vigneron, V. Zarzoso, E. Moreau, R. Gribonval, and E. Vincent, Eds. Springer Berlin / Heidelberg, 2010, vol. 6365, pp. 140–148.

[83] G. Mysore and P. Smaragdis, "A non-negative approach to semi-supervised separation of speech from noise with the use of temporal

dynamics," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, may 2011, pp. 17 –20.

[84] V. Stouten, K. Demuynck, and H. Van hamme, "Discovering phone patterns in spoken utterances by non-negative matrix factorization," *Signal Processing Letters, IEEE*, vol. 15, pp. 131 –134, 2008.

[85] P. D. O'Grady and B. A. Pearlmutter, "Discovering speech phones using convolutive non-negative matrix factorisation with a sparseness constraint," *Neurocomput.*, vol. 72, no. 1-3, pp. 88–101, Dec. 2008.

[86] P. Smaragdis, "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs," in *Independent Component Analysis and Blind Signal Separation*, ser. Lecture Notes in Computer Science, C. Puntonet and A. Prieto, Eds. Springer Berlin / Heidelberg, 2004, vol. 3195, pp. 494–499.

[87] B. Raj and P. Smaragdis, "Latent variable decomposition of spectrograms for single channel speaker separation," in *Applications of Signal Processing to Audio and Acoustics, IEEE Workshop on*, oct. 2005, pp. 17 – 20.

[88] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Machine Learning*, vol. 42, pp. 177–196, 2001.

[89] R. Zass and A. Shashua, "Nonnegative sparse pca," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007, pp. 1561–1568.

[90] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1," *NASA STI/Recon Technical Report N*, pp. 27 403–+, Feb. 1993.

[91] K.-F. Lee and H.-W. Hon, "Speaker-independent phone recognition using hidden Markov models," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 37, no. 11, pp. 1641 –1648, nov 1989.

[92] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye, *Probability and Statistics*, 8th ed. Pearson International, 2007.