Dissertation ETH Nr. 19507

Speech Recognition Techniques for Languages with Limited or No Resources

A dissertation submitted to the
ETH ZURICH

for the degree of
DOCTOR OF SCIENCES

presented by
MICHAEL GERBER
Dipl. El.-Ing. ETH
born December 14, 1975
citizen of Langnau i. E. (BE), Switzerland

accepted on the recommendation of
Prof. Dr. Lothar Thiele, examiner
Prof. Dr. Jean-Pierre Martens, co-examiner
Dr. Beat Pfister, co-examiner

2011

Acknowledgements

I would like to thank everybody who has in one way or another contributed to this thesis.

First and foremost I would like to thank Prof. Dr. Lothar Thiele and Dr. Beat Pfister for supervising my research. In particular I would like to thank Beat for guiding me in a research field which is treated for several decades but still leaves the performance of the human brain unmatched. I would also like to thank Prof. Dr. Jean-Pierre Martens for the valuable suggestions for this thesis and for taking part in the examination committee.

Furthermore I would like to thank the other members of the speech group whom I could always consult – be it with scientific, programming, linguistic or whatever else questions. In particular I would like to thank René for encouraging the use of HMMs and Tobias for his valuable contributions to my papers.

I am very grateful for the support I got from my family and friends. Chimgee, this thesis would not have been possible with your support. Amina, thank you for smiling at daddy even if he was very often preoccupied with his work. I would also like to express my gratitude to my parents for everything they have done for me. In general I would like to thank everybody who supported my wife and me in the last year, which was very intense with our new task as parents and the final stage of our PhD theses about *glucosinolate-rich plants* and *resource-poor languages*.

Contents

List of Abbreviations

CMS	cepstral mean subtraction
DTW	dynamic time warping
EER	equal error rate
GMM	Gaussian mixture model
HMM	hidden Markov model
IPA	international phonetic alphabet
IWR	isolated word reconition / recognizer
KNN	k nearest neighbors
LBG	Linde Buzo Gray
LVCSR	large vocabulary continuous speech recognition / recognizer
MLP	multilayer perceptron
PM	pattern matching
UBM	universal background model
VMLP	verification multilayer perceptron

Notation

$\mathbf{X}^{(k)}$	observation sequence k
$\mathbf{x}_t^{(k)}$	observation at time t in $\mathbf{X}^{(k)}$
$\mathcal{X}$	set of observation sequences
$\mathbf{X}_t^{(k)}$	sequence of first t observations of $\mathbf{X}^{(k)}$
φ	elementary HMM describing a phoneme or abstract acoutic element
Φ	set of elementary HMMs
λ	composite HMM, composed of several elementary HMMs
A_n	abstract acoustic element with index n
$\mathcal{A}$	set of abstract acoustic elements
N	number of phonemes or abstract acoutic elements in a set
n	index of phoneme or abstract acoustic element
M	number of mixtures of a GMM
m	index of a mixture in a GMM
K	number of feature sequences
k	index of a feature sequence
W	number of words
w	index of a word
U	number of sub-word units in a word
u	index of a sub-word unit
$\boldsymbol{\mu}_{nm}$	mean of mixture m in state n
$\tilde{\boldsymbol{\mu}}_{nm}$	estimate of mean of mixture m in state n
$\boldsymbol{\Sigma}_{nm}$	variance of mixture m in state n

$\tilde{\boldsymbol{\Sigma}}_{nm}$	estimate of variance of mixture m in state n
c_{nm}	weight of mixture m in state n
$\tilde{c}_{nm}$	estimation of a mixture weight
$Q^{(k)}$	state sequence for obsservation sequence k
$q_t^{(k)}$	state visited at time t in $Q^{(k)}$
$\mathcal{Q}$	set of state sequences
$\hat{Q}^{(k)}$	optimal state sequence of observation sequence k
$\hat{q}_t^{(k)}$	state visited at time t in $\hat{Q}^{(k)}$
$\hat{\mathcal{Q}}$	set of optimal state sequences
$\hat{Q}_t^{(k)}$	sequence of first t states of $\hat{Q}^{(k)}$
$\tilde{Q}^{(k)}$	selected state sequence for observation sequence k
$\tilde{q}_t^{(k)}$	state visited at time t in $\tilde{Q}^{(k)}$
$\tilde{\mathcal{Q}}$	set of selected state sequences
$Z^{(k)}$	sequence of visited sub-word units for observation sequence k
$z_u^{(k)}$	sub-word unit at index u of $Z^{(k)}$
$\hat{Z}^{(k)}$	optimal sequence of sub-word units for observation sequence k
$\hat{z}_u^{(k)}$	sub-word unit at index u in $\hat{Z}^{(k)}$
$\hat{G}^{(k)}$	sequence of best mixtures within states for obs. seq. k
$\hat{g}_t^{(k)}$	best mixture within a state at time t for obs. seq. k
$\mathbf{z}_n$	codebook vector with index n
$\delta_t(n)$	auxiliary variable for Viterbi in state n and time t
$\gamma_t^{(k)}(n)$	auxiliary variable for HMM training (observation sequence k, time t and state n)
$\zeta_t^{(k)}(n,m)$	auxiliary variable for HMM training (observation sequence k, time t, state n and mixture m)
$b_n(\mathbf{x})$	observation probability of observation $\mathbf{x}$ in state n
S_n	state n of an HMM

Abstract

There are several thousand languages in the world and each language has a multitude of dialects. State-of-the-art speech recognition technologies are however only available for a few of them because of the lack of acoustic and textual resources which are necessary to build these recognizers.

In this thesis we aim at the development of speech recognition technologies for languages with limited or no resources. For many applications such as the control of machines or home appliances by voice it is not necessary to have a continuous speech recognizer with a large vocabulary. It is then possible to resort to techniques which need only very little language-specific resources.

In order to build isolated word recognizers for any language we relied on speech recognition techniques with an utterance-based vocabulary. In these techniques each word of the vocabulary is defined by one or several sample utterances. This way of defining the vocabulary is language-independent and has the further advantage that it can be done by everybody since no expert knowledge is required.

To improve the recognition rate of speech recognition with an utterance-based vocabulary we worked with two techniques: the first one based on dynamic time warping in combination with specially trained artificial neural networks and the second one based on hidden Markov models with data-driven sub-word units.

With the availability of moderate resources from the target language we were able to develop a recognizer technique which yielded compa-

rable results to a state-of-the-art recognizer which requires in contrast
to our technique a pronunciation dictionary to build the word models.
When no resources of the target language were available and resources
from other languages than the target language had to be used instead,
the performance of state-of-the art recognition was not achievable with
the utterance-based recognizer techniques developed in this thesis. Yet,
in this case the developed approaches allowed to halve the error rate
of isolated word recognition with an utterance-based vocabulary com-
pared to a standard approach based on dynamic time warping using
the Euclidean distance measure.

We also applied the developed techniques to other applications such
as acoustic data mining. In this way it was possible to tackle these prob-
lems for speech signals of any lanuage since the developed techniques
do not require resources of the target lanuage.

Kurzfassung

Weltweit existieren einige Tausend Sprachen, und in jeder Sprache werden viele verschiedene Dialekte gesprochen. Spracherkenner, welche dem Stand der Technik entsprechen, stehen allerdings nur in den wenigsten Sprachen zur Verfügung, da zu ihrer Implementierung umfangreiche akustische und linguistische Ressourcen notwendig sind.

In dieser Arbeit haben wir Techniken entwickelt und getestet, welche die Spracherkennung in Sprachen mit wenigen oder keinen Ressourcen verbessern. Für viele Anwendungen, wie zum Beispiel die Steuerung von Maschinen oder Haushaltsgeräten, ist es nicht nötig, einen kontinuierlichen Spracherkenner mit einem grossen Vokabular zur Verfügung zu stellen. Mit diesen geänderten Anforderungen werden Techniken, welche keine sprachspezifischen Ressourcen benötigen, möglich.

Um die Erkennung von isolierten Wörtern in beliebigen Sprachen zu ermöglichen, haben wir Techniken, die ein Vokabular verwenden, das auf Musteräusserungen basiert, verbessert. Bei diesen Techniken wird jedes zu erkennende Wort durch eine oder mehrere Musteräusserungen definiert. Neben der Sprachunabhängigkeit haben diese Techniken auch den Vorteil, dass ein Vokabular von jedermann definiert werden kann, da kein Expertenwissen nötig ist.

Zur Verbesserung musterbasierter Spracherkennung haben wir grob mit zwei Techniken gearbeitet: die erste basiert auf dynamischer Zeitanpassung in Kombination mit speziell trainierten künstlichen neuronalen Netzen, und die zweite basiert auf *Hidden Markov Modellen* mit speziellen Daten-motivierten Modellen.

Wenn einige wenige Ressourcen der Zielsprache zur Verfügung standen, konnten wir mit den entwickelten Techniken Erkennungsraten erreichen, welche jenen eines dem Stand der Technik entsprechenden, Aussprachewörterbuch-basierten Erkenners in nichts nachstehen, auch wenn dieser mehr Ressourcen wie zum Beispiel ein Aussprachewörterbuch benötigt. Falls gar keine Ressourcen in der Zielsprache zur Verfügung standen und auf Ressourcen einer anderen Sprache für das Training der Modelle zurückgegriffen werden musste, konnten die Erkennungsraten von Aussprachewörterbuch-basierten Erkennern nicht erreicht werden. Die Fehlerraten welche wir mit unseren Erkennern erreichten, waren allerdings trotzdem nur halb so gross wie jene von konventionellen Mustervergleich-Erkennern.

Wir haben die entwickelten Techniken auch für andere Anwendungen, wie zum Beispiel die Suche von lautlich ähnlichen Abschnitten, wie Wörtern, in zwei Sprachsignalen angewendet. Diese Anwendungen werden dank den neuen Techniken in beliebigen Sprachen möglich.

Chapter 1

Introduction

1.1 Problem Statement

Recognition of isolated words is a fundamental application of speech recognition. It is for example necessary to control machines or home appliances by voice.

Most research in automatic speech recognition is nowadays focused on large vocabulary continuous speech recognition (LVCSR) and isolated word recognition (IWR) is considered as a special case of LVCSR and tackled with the same methods. LVCSR have high resource requirements to the language which they are used in. They need for example a pronunciation dictionary and large annotated speech corpora.

Around 4000 languages exist worldwide ([SW01]), but only in some tens of them a pronunciation dictionary is available ([SW01]). Besides that most people do usually not speak the canonical form of a language but use a multitude of dialects, which usually lack dialectal dictionaries. This makes the standard LVCSR techniques unusable for most languages and dialects.

Another problem is the number of languages which a recognizer needs to cover. If an internationally operating company wants to incorporate speech recognition into its products, it needs to offer a huge

portfolio of languages and it is difficult to use language-dedicated recognizers for all languages even if they might be available.

In this thesis we aim at building IWR with a satisfactory performance in any language and dialect.

1.2 Isolated Word Recognition

Isolated word recognition can be categorized by the way the words which need to be recognized are represented in the vocabulary. In the word-based approach the representations of the words are independent - i.e. they do not share components or parameters. In the sub-word-based approach every word is represented by a sequence of sub-word units. The set of sub-word units which the sequences are composed of is shared among the words.

1.2.1 Word-based Recognition

The words in word-based recognizers are often represented by templates (i.e. by feature sequences of example utterances of the words). This approach is referred to as template-based recognition. A major challenge in template-based recognition is the variability among signals of the same word, even if they are recorded from the same speaker over the same channel. Some of these variations can be reduced by considering the frame-wise representation of appropriately chosen features. Temporal variations can be compensated by the flexibility inherent to dynamic time warping, which is normally used to compare signals.

An alternative approach to word-based recognition is to represent each word with an individual model, i.e. a hidden Markov model (HMM) or an artificial neural network. A disadvantage of this approach is that many utterances of each word are necessary to train the word models.

1.2.2   Sub-Word-based Recognition

A way to alleviate the need for a lot of training material for each word to
be recognized is to represent each word as a sequence of sub-word units.
Constructing the vocabulary for the recognizer is then divided into two
tasks: The training of an appropriate set of sub-word unit models on
the one hand and the appropriate concatenation of the sub-word units
for each word to be recognized on the other hand.

Sub-word-based recognizers can be further characterized by the way
the sub-word units are concatenated. In the first category, which is here
termed dictionary-based recognition, the sub-word units are concate-
nated according to a dictionary. In the second category, the sub-word
units are concatenated according to sample utterances of the words to
be recognized.

Dictionary-based Recognition

In this recognizer category the items to be recognized are usually mod-
eled with hidden Markov models (HMMs), which are concatenated from
phone models according to a pronunciation dictionary. This allows the
construction of HMMs for various recognizer topologies such as IWR or
LVCSR. There are several prerequisites of phoneme-based recognizers:

1. Annotated data to train appropriate statistical models of the
   phonemes has to be available.

2. A translation of each occurring word into one or several sequences
   of phonemes has to be available. This is usually achieved with a
   pronunciation dictionary.

3. An appropriate language model which defines the sequences of
   words which the recognizer is able to understand and the proba-
   bilities of word sequences is necessary.

These prerequisites make the phoneme-based approach language de-
pendent since the resources mentioned above need to be available for
every language in which a recognizer should be deployed.

For isolated word recognition only the first two prerequisites are necessary since the language model is trivial. The first prerequisite can be partly circumvented by using cross-lingual resources and therefore only the second prerequisite remains. The available cross-lingual techniques are summarized in Section 1.3. There were attempts to circumvent the need for a pronunciation dictionary by taking context-dependent graphemes as sub-word units in the speech recognizer ([KN02]). These approaches were successful for some languages, including languages with non-Latin script such as Thai ([CHS06]) but yielded poorer results for languages which have a more complex mapping from graphemes to pronunciation such as English ([KSS03]).

Utterance-based Concatenation of Sub-Word Units

An alternative method to determine the sequences of sub-word units which is also applicable if no pronunciation dictionary is available is to determine the sub-word unit sequence according to utterances of the words. In this case there is also more freedom in terms of the selection of an appropriate set of sub-word units, since the sub-word units do not need to be linguistic units such as phonemes or graphemes.

1.3 Available Cross- and Multi-Lingual Techniques

In this section we give an overview of problems in cross- and multilingual speech recognition and the available methods to solve them.

1.3.1 Multilingual Vocabulary

There are several applications for which a speech recognizer for a single language is not enough. Tourist information systems should for example be controllable in several languages. In this case it can be expected that a user uses only one language and therefore the cross-lingual aspect is only that the system has to be operable in several languages. The situation gets somewhat more complicated if the user can switch the language. In bilingual communities it is for example quite common that the speakers switch the language even within one sentence. A sys-

tem which is able to recognize two languages at the same time was for example presented in [Wen97].

A number of systems have been presented which combine the phoneme inventories of several languages into one fused phoneme inventory in order to avoid the need for language-dependent phoneme models for every language which has to be understood by a system. Usually these systems are based on standardized international phoneme inventories such as Sampa ([Wel89]), IPA or Worldbet ([Hie93]). A straightforward approach is to merge the phonemes with the same IPA symbol. In [Kun04] even more phonemes could be merged in this way because the distinction between long and short vowels was abolished. A data-driven fusion of acoustically similar phonemes was presented in [BGM97]. In [DAB98] the fusion of phonemes was guided by a similar log likelihood in recognition experiments. Sharing of Gaussian mixtures of the models in the different languages was allowed in [Wen97]. These approaches usually resulted in a moderate performance loss compared to the use of dedicated phoneme models for each language to be recognized.

An interesting cross-lingual application is the recognition of proper names such as city names since a speaker of one language may pronounce a proper name of another language in different ways: either he can pronounce the name in his own language or in the foreign language. When he uses the foreign language he may have a stronger or a weaker accent. A possibility to handle this case which was for example chosen in [Bea03] is to add additional pronunciation variants such that both the native and the foreign pronunciation is recognized. In [SM07a] phonemes of a proper name which might be pronounced in a non-native way were represented with a phonologically inspired back-off model. In [SNN01] a German speech recognizer which is also able to recognize English movie titles was presented. To that end, the phonemes of English and German were merged.

1.3.2 Languages with Limited Acoustic Training Data

For some languages a pronunciation dictionary is available but there is not enough acoustic training data which is appropriately annotated for the training of phone models. In this case it may be possible to take

phone models from one or several other languages, here called source languages, to derive models of the target language.

Also these approaches are often based on international phoneme inventories. If phone models which are necessary for a target language have an equivalent phone model in one or several source languages, the models of the target language can be substituted by the models of source languages. The crucial part is usually how phone models of the target language which are not available in any of the source languages are handled. In [Ueb01] these phone models are substituted with acoustically close phone models. For the selection of the appropriate model a small amount of data has to be available for the phonemes of the target language. A way which is a bit more sophisticated was taken in [Byr00]. Here the phone models of the target language are combined from one or several models of the source languages with dynamic model combination introduced in [Bey98].

If some training data is available for the target language the models derived from other languages as explained in the previous paragraph can be taken as seed models which are adapted with the limited training data of the target language. Such an approach was for example implemented in [SW01].

There are alternative approaches which suggest a reduction of the phoneme model mismatch between languages by an appropriate selection of features. To use articulatory features was for example done in [Sin08]. Alternatively, phonological features can be used as presented in [SM07b].

1.3.3 Non-Native Speakers of a Language

Speech recognizers often perform much worse for non-native speakers than for native speakers. In [LHG03] it was for example found that speech recognition for non-native English speakers of different mother tongues was almost twice as accurate if training data was used from speakers of the same mother tongue than if training data from speakers of another mother tongue was used. In [Hua01] it was observed that the native accent of speakers introduced the second most important source of inter-speaker variability right after the gender difference. That non-native accents are even more difficult for speech recognizers than native

accents has been argued in [VC01]. In [FSM03] it was investigated how the first language of bilinguals influences the pronunciation of English phonemes. A way to alleviate this problem was presented in [BJ04]: The phonemes of a French speech recognizer were enhanced for non-native speakers by adapting the French models with data of phoneme-equivalents of the speakers native language.

1.4 Investigated Approaches to Isolated Word Recognition

State-of-the-art recognition of isolated words is usually performed with a dictionary-based approach. The resource requirements for dictionary-based recognizers are however unsatisfiable for most languages – even if using cross-lingual techniques as described in Section 1.3.

Therefore we have investigated approaches which can be used for any language. Uttering the words of the vocabulary is an easy and user-friendly way of defining a small vocabulary. The basic concept of an utterance-based vocabulary does not make any assumption about the language which the recognizer is used in. Also no assumption is made about the user or user population of the recognizer. This is a further advantage since the acoustic models of dictionary-based recognizers are often shaped for a special speaker population. Most recognizers are for example optimized for the use by adults and have a very poor performance for child speakers (see for example [EB04]).

The usual approach to recognizers with utterance-based vocabularies is template-matching with dynamic time warping and an Euclidean distance measure as described in 1.2.1. These recognizers usually yield a considerably lower accuracy than state-of-the-art recognizers do, especially if the template utterances were spoken by a different speaker than the user of the recognizer. In this thesis we investigated two approaches to enhance recognition with utterance-based vocabularies:

- Use a more appropriate distance measure in template-based recognizers based on dynamic time warping.

- Use utterance-based concatenation of sub-word HMMs as outlined in Section 1.2.2

1.5   Evaluation of the Recognizers

The experiments shown in this thesis are designed to evaluate how different recognizers with utterance-based vocabularies compare among each other and how they compare with a dictionary-based recognizer.

A key question is the type of resources which are necessary from the target language. The basic principle of utterance-based recognition needs only utterances of the words to be recognized as resources. For particular techniques other resources may however be necessary. In our case this is for example training data for the distance measure in the dynamic time warping approach or for the sub-word unit models in the HMM approach.

In the *intra-language* case, i.e. if the training data is taken from the target language, it is interesting to evaluate what quality the training material needs to have (e.g. if orthographic annotations are necessary). In the *cross-language* case, i.e. if the training data is not taken from the target language, no data except for sample utterances of the words to be recognized is required from the target language. Then it is however important to evaluate what impact the language-mismatch has on the recognition performance.

In order to evaluate isolated word recognition we performed ten-word recognition tasks. In all tasks many ten-word vocabularies were tested with several test utterances. The tasks are described in detail in Appendix E.2. We performed tests with German and with French tasks. The tasks were performed with speakers of the speaker sets $\mathcal{S}_{G,poly,3}$ and $\mathcal{S}_{F,poly,3}$ as described in Appendix E.1.

To train sub-word units and multilayer perceptrons we used data from speakers of the speaker sets $\mathcal{S}_{G,poly,1}$ and $\mathcal{S}_{F,poly,1}$, which are disjoint from the test speaker sets.

To test the performance in the intra-language case, the tests were performed with models (multilayer perceptrons or sub-word unit models) trained on data of the target language. To test the performance in the cross-language case, the German tests were performed with models trained on French data and vice versa.

A big challenge for recognizers with an utterance-based vocabu-

lary is a speaker-mismatch between the speakers of the vocabulary-utterances and the test speaker. In order to evaluate the approaches with different degrees of speaker-mismatch we tested three scenarios:

- *Speaker-dependent:* The reference and test utterances are from the same speaker. This is the case for applications in which each user of a system will define his own vocabulary by uttering each word one or several times. This is the easiest case since there is only intra-speaker variability.

- *Cross-speaker:* The reference utterances are spoken from one speaker and the test utterances are spoken by another speaker. This is the case for applications in which we have only reference utterances from one speaker but need to build a recognizer which is used for any speaker (i.e. in a speaker-independent fashion). In this case the inter-speaker variability is likely to be a big source of error.

- *Speaker-independent:* The references utterances are from a bigger population of speakers and the test utterances are from speakers of another population. This is the case for applications wherein we have utterances from many speakers available. In this case a speaker-independent word model can be built from several utterances spoken by different speakers.

1.6   Scientific Contributions

1. We developed a new approach to use appropriately trained multilayer perceptrons instead of other distance measures such as the Euclidean distance in speech-recognition methods which are based on dynamic time warping.

2. To obtain appropriate sub-word unit models for recognition with an utterance-based vocabulary we developed a scheme to train abstract acoustic elements.

3. We extended the Viterbi algorithm in a way that makes is possible to find a sequence of sub-word units which optimally describes several utterances. Additionally we devised an approximation of

the exact algorithm which performs well but is computationally much less complex.

4. We achieved a big improvement of the recognition rate of recognizers with an utterance-based vocabulary compared to a baseline approach. With appropriate training data we were able to build a recognizer with an utterance-based vocabulary which had a performance similar to a dictionary-based recognizer.

5. We developed algorithms to use the developed techniques in other applications of speech processing – for example for acoustic data mining. These algorithms were successfully used for a pattern-matching approach to speaker verification.

1.7   Structure of the Thesis

The thesis is structured in six chapters:

Chapter 2 describes isolated word recognition with DTW and shows that appropriately trained verification multilayer perceptrons are a good alternative to other distance measures.

Chapter 3 describes HMM-based isolated word recognition with sub-word units which are concatenated according to sample utterances. This includes the description of abstract acoustic elements used as sub-word units and an extension of the Viterbi algorithm for several observation sequences.

Chapter 4 compares different recognizer techniques with utterance-based vocabulary among each other. These techniques are also compared with a dictionary-based recognizer.

Chapter 5 describes further applications of speech processing such as data mining, utterance verification and speaker verification which may profit from the techniques developed in this thesis.

Chapter 6 gives some concluding remarks of this thesis including an outlook.

Chapter 2

Improving DTW-based Word Recognition

The first isolated word recognizers were template-based. In [VZ70] or [Ita75] the words which are represented in the vocabulary as feature sequences are compared to the test utterances with dynamic time warping (DTW) to compensate for temporal variations. The recognized word is the one with the smallest distance to the test utterance.

Improvements of DTW-based approaches were mainly achieved by the use of feature transformations and alternative distance measures which are reviewed in Section 2.1. A speedup for the DTW approach by first using a coarse temporal resolution was suggested in [SC04]. DTW has recently regained some interest even for large vocabulary continuous speech recognition (LVCSR). Different schemes to use DTW for LVCSR are reviewed in [GL98] and a complete LVCSR based on DTW was presented in [DW07].

This chapter introduces the verification multilayer perceptron (VMLP), a specially structured MLP and shows how it can be used as an alternative distance measure in Section 2.2. Section 2.3 contains experiments which show the performance of the VMLP in comparison to other distance measures and feature transformations.

2.1 Overview of Discriminative Approaches for DTW

A speech signal does not only contain information about the spoken text but also about the speakers voice, the speakers mood, the characteristics of the recording channel, the background noise and so forth. A good overview of factors which lead to the variability of speech signals is for example [Ben07]. In order to improve speech recognition, various discriminative methods have been proposed which (pre-)process speech in a way that the information of the spoken text has a bigger effect than other information which is disturbing for speech recognition. Using VMLPs, which are presented in Section 2.2, is such a method. In this section we give a short overview of work which is done to improve speech recognition by using discriminative methods. In Section 2.1.1 we will have a look at alternative distance measures applicable for DTW. Discriminative feature transformations will be summarized in Section 2.1.2.

2.1.1 Alternative Distance Measures for DTW-based Recognizers

DTW-based recognizers need either a distance measure which determines the difference between two frames or – seen from a different point of view – a similarity measure which gives the probability that two frames are from the same phoneme. Very often the Euclidean or the Mahalanobis distance are used for this purpose. We have suggested to use verification multilayer perceptrons (VMLPs) as a similarity measure. A VMLP computes the posterior probability that two frames are from the same class. The VMLPs are described in Section 2.2. Following the introduction of VMLPs another research group has suggested in [Pic09] that the scalar product of phoneme posterior vectors can be used as an alternative to VMLP. This approach is discussed in Section 2.2.2.

There are also other approaches which are for example based on local distance measures. In [DW04] the two frames for which the distance has to be computed are first classified with a phoneme recognizer

and for each state in each phoneme a local distance measure is defined which makes use of the variance within each state. In [Mat04] a similar approach is pursued but the local distance measure is optimized with a gradient descent learning procedure presented in [PV00] to perform well in a k nearest neighbors classifier.

2.1.2   Feature Transformations

Another approach to using a better distance measure is to use feature transformations which transform an input feature space into a new one in which classes are easier separable. Some of these transformations are linear and can therefore be expressed with a transformation matrix. Other approaches perform a nonlinear transformation and are mostly based on MLPs.

Linear Transformation

The linear feature transformations differ among each other in the way the transformation matrices are trained. In the standard form of linear discriminant analysis as described for example in [DHS01] the objective is to maximize the ratio of between-class data scatter to within-class data scatter. The training has a closed solution which is based on finding Eigenvectors, is however based only on the scatter matrices.

With the heteroscedastic discriminant analysis an alternative was therefore presented in [KA98]. Here the transformation is based on iterative maximum likelihood training. A further refinement presented in [Sao00] performs an additional maximum-likelihood linear transform on top of the heteroscedastic linear discriminant analysis to ensure the linear independence of the resulting features.

In [Dem99] an approach which is based on the maximization of mutual information is presented. This approach is thought as a replacement of the discrete cosine transform which is used in the last step of the calculation of the Mel frequency cepstral coefficients.

MLP-based Feature Transformations

Several methods to train MLPs which perform a feature transformation have been suggested. Nonlinear discriminant analysis which was introduced in [Fon97] is based on a bottleneck layer. An MLP is trained which has at the input the untransformed features and at the output a phoneme vector which is 0 for all phonemes except for the correct phoneme for which it is 1. This MLP has a smaller last hidden layer which is termed bottleneck layer since all information has to be squeezed through this bottleneck layer. Later, when the MLP is used, only the part from the input layer to this bottleneck layer is used and the activations of the bottleneck layer are the transformed features.

A very popular feature transformation was presented with the Tandem features introduced in [HES00]. Here an MLP is trained in a similar way as in the bottleneck approach described above. However, the outputs of the trained MLP are taken directly as the transformed features. These transformed features are also termed phoneme posteriors since every feature corresponds to the posterior probability of a given phoneme. These transformed features were originally intended to be used with HMMs but in [AVB06] it was shown that they also yielded good results in DTW-based recognizers. It was shown in [ZCMS04] that phoneme posteriors are less speaker-dependent than the untransformed features. In [ZCMS04] it was also shown how phoneme posteriors can be merged. If different phoneme-posterior transformations are trained for the same target (i.e. the same phonemes) but for different input features the phoneme posteriors of the different transformations can be combined as a weighted sum. More elaborate merging techniques, e.g. with an additional MLP for merging were presented in [HM05].

2.2 Multilayer Perceptron for Class Verification

Multilayer perceptrons (MLPs) are successfully used in speech processing such as for example to calculate phoneme posteriors as described in Section 2.1.2. In this case they are used to *identify* a phoneme from a given feature vector. Expressed in more general terms, the MLPs are

used for the *identification* of input vectors with a class from within a closed set of classes.

There are applications however, where the identification of input vectors is not necessary but it has to be *verified* whether two feature vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ are from the same class or not. Therefore we developed the concept of VMLPs which we expect to calculate the posterior probability

$$P(class(\mathbf{x}_1) = class(\mathbf{x}_2)|\mathbf{x}_1, \mathbf{x}_2) \qquad (2.1)$$

Such an application in speech processing is for example the verification whether two given speech frames are from the same phoneme or not. The objective is therefore to verify phonemes. Another application is in a pattern matching-based approach to speaker verification presented in Section 5.2. Here the objective is to verify speakers.

In experiments which are described in Appendix A we experimentally showed that the verification results of appropriately trained VMLPs are close to optimal. Furthermore we showed that VMLPs can even be used to discriminate between classes which were not present in the training set, but have the same verification objective (e.g. verifying phonemes or speakers). This is an especially useful property since it allows that for example training data for speaker VMLPs does not need to be collected from the target speaker population but can be collected from another population of virtually unlimited size.

2.2.1 Verification MLP Structure

Since the VMLP has to decide whether two given input vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ are from the same class, the VMLP has to process vector pairs rather than single vectors. The target output of the VMLP is o_s if the two vectors of the pair are from the same class and o_d if they are from different classes. The vectors are decided to belong to the same class if the output is closer to o_s and to different classes otherwise.

Thus the structure of the VMLP is as shown in Figure 2.1. Alternatively the VMLP could be implemented with two outputs – one for the probability that the input vectors are from the same class and the other for the probability that the input vectors are from different

classes. This topology would allow the use of a softmax output layer which is often used for classification problems (e.g. for phoneme classification in [BM94]) since it guarantees that the outputs sum to one. We have experimentally seen however that a topology with one output yielded similar results than the simpler network topology which we used.
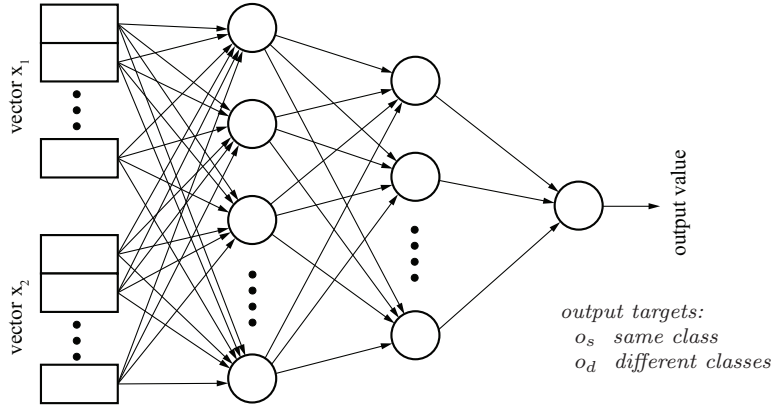
Figure 2.1: *Structure of the VMLPs.*

The VMLP shown in Figure 2.1 has two hidden layers. It is known that this topology is able to describe any input to output mapping provided that enough units are available (see for example [Lip87]). It has to be evaluated whether one hidden layer suffices for a given problem and how many neurons are necessary. The evaluation of these parameters for our task in speech recognition is done in Section 2.3.4.

2.2.2   Posterior Scalar Product

Following the introduction of VMLPs an alternative was presented in [Pic09] and [APB10]. The method is based on the observation that a VMLP which is trained with a standard error criterion such as mean squared error will optimize the same criterion as the scalar product of the vectors of N phoneme posteriors $P(phoneme_n|\mathbf{x}), n = 1 \ldots N$ (cf.

Section 2.1.2) of two feature vectors $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$\sum_{n=1}^{N} P(phoneme_n|\mathbf{x}_1) * P(phoneme_n|\mathbf{x}_2) \qquad (2.2)$$

This assumption was both theoretically and experimentally confirmed for a closed set of classes.

In contrast to the tests performed in [APB10], we aim at verifying whether two feature vectors are from the same class for an open set of classes. For this open set of classes only the classification objective (e.g. classifying speakers or classifying phonemes) is expected to be known.

A further difference is the requirements for the training data. Whereas the posterior scalar product approach needs a phoneme segmentation to train the phoneme posterior MLPs, only orthographic annotations are necessary for the training material of VMLPs.

The posterior scalar product would be a very interesting alternative from the point of view of computational complexity. The computation of the phoneme posteriors is confined to the individual feature sequences and between the signals only the scalar product has to be computed.

Comparative results of the posterior scalar product and VMLPs will be given in Section 2.3.5.

2.3 Experiments

2.3.1 Description of the used DTW Recognizer

We used an asymmetric DTW implementation which used Itakura constraints ([Ita75]) to warp the test utterance on the reference utterances. This is a reversed approach to mapping the reference utterances on the test signal which is often used. Our approach required the accumulated distance to be normalized with the number of frames in the reference utterance (i.e. using the average distance). We achieved better results with this approach, probably because the features in the frames of the reference template are more reliable especially if several utterances are

used to generate a template. At the start and at the end of both reference and test signal we allowed the DTW algorithm to skip up to 50 ms (5 frames) in order to compensate for slightly incorrect endpoints. The final distance between a reference signal and a test signal was not directly the accumulated distance on the last frame of the warping curve. Instead we used the weighted average Euclidean distance of all frame pairs along the warping curve as the final distance. As weighting factor we used the mean RMS value of both frames in a pair. We have empirically optimized the parameters of this DTW recognizer.

2.3.2   DTW Recognizer with Perceptron-based Distance Measure

For the DTW recognizer it is possible to use a feature transformation as described in Appendix 2.1.2, an alternative distance measure such as a distance measure based on a verification multilayer perceptron (VMLP) presented in Section 2.2 or both. When a VMLP was used as a distance measure, its output was linearly mapped in a way that the positive MLP output which indicated the same class of both input vectors was mapped to 0 and the negative MLP output which indicated different classes for both input vectors was mapped to 1.

2.3.3   Verification MLP Training

The VMLPs were trained by means of the backpropagation algorithm with the mean squared error as error criterion. The weights were randomly initialized and a momentum term was used during the training. For a hyperbolic tangent output neuron a good choice for the output targets is $o_s = 0.75$ and $o_d = -0.75$ such that the weights are not driven towards infinity (see for example [Hay99]). With these settings we experienced that at the beginning of the training the difference between desired and effective output decreased quite slowly but that the training never got stuck in a local minimum.

Training Data for Phoneme Verification MLPs

In order to train a VMLP which will verify whether two input feature vectors are from the same phoneme we needed on the one hand positive feature vector pairs for which we know that the feature vectors are from the same phoneme. On the other hand we needed negative feature vector pairs with feature vectors from different phonemes. A method to obtain such training vector pairs was to apply DTW on vector sequences extracted from utterances of the same word. The feature vector pairs located on the warping curve obtained from the DTW algorithm could then be taken as the positive feature vector pairs. The negative feature vector pairs were randomly taken from points outside the warping curve which had an Euclidean distance above a certain threshold.

We used data from speakers of the sets $\mathcal{S}_{G,poly,1}$ or $\mathcal{S}_{F,poly,1}$ to train the phoneme VMLPs and data from speakers of the sets $\mathcal{S}_{G,poly,2}$ or $\mathcal{S}_{F,poly,2}$ as validation data. The speaker sets are described in Appendix E.1.

2.3.4 Determination of Appropriate Structure and Size of the VMLP

As pointed out in Section 2.2.1, it is necessary to optimize the structure and the size of the VMLP. The network should be as small as possible since training problems such as overfitting may arise (see for example [Hay99]) if the training data is limited. The network size has also a substantial impact on the speed of the recognizer.

We have investigated both, networks with one hidden layer and networks with two hidden layers. Both network types were tested with different sizes. For the networks with two hidden layers we chose the size of the first hidden layer to be three times as big as the size of the second hidden layer since the first hidden layer should not be too small compared to the number of input neurons ([Lip87]). The tested network sizes are shown in Table 2.1.

We evaluated the VMLPs by measuring the performance of DTW-based recognizers which use a VMLP as a distance measure in a speaker-dependent scenario for German (task 1) and for French (task 4)

approx. number of parameters	2 hidden layers		1 hidden layer
	size of hidden layer 1	size of hidden layer 2	size of hidden layer
1900	30	10	35
4400	60	20	82
7500	90	30	140
11300	120	40	209
15600	150	50	290
20500	180	60	381

Table 2.1: *Tested network structures and sizes.*

and in a cross-speaker scenario (tasks 2 and 5 for German and French, respectively) as described in Appendix E.2. At the input are two feature vectors $Feat_{noCms}$ as described in Appendix C.1.

The results showed that the two topologies with one and two hidden layers yielded similar results. The single-layer was slightly better, especially with the bigger networks. Only for very small networks the two-layer topology was a bit better.

The networks could also be quite small without the performance degrading too much. This allows to implement faster recognizers by minimizing the network size. A network with around 8000 tunable weights was enough for this application.

2.3.5   Evaluation of Discriminative Methods

We then evaluated the performance of the VMLP and compared it to alternative distance measures such as the posterior scalar product described in Section 2.2.2 and to feature transformations such as phoneme posteriors or linear discriminant analysis (LDA) as described in Section 2.1.2.

The results of the German and French tasks of the speaker-dependent scenario (tasks 1 and 4) and of the cross-speaker scenario (tasks 2 and 5) as described in Appendix E.2 are listed in Table 2.2. We used features $Feat_{noCms}$ as described in Appendix C.1 and a VMLP
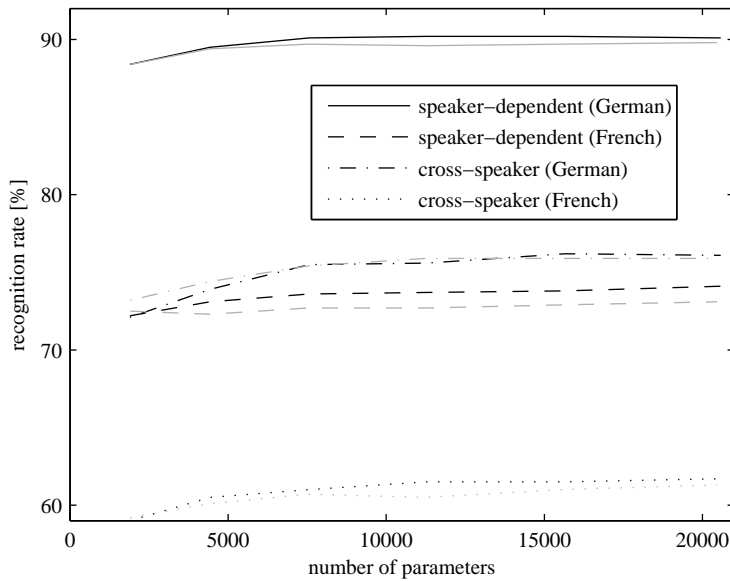
Figure 2.2: *The influence of network structure and network size on the recognition rate of speaker-dependent and cross-speaker template-based IWR. The recognition rates of the architectures with two hidden layers are in grey, the ones for architectures with one hidden layer in black.*

with one hidden layer of 209 neurons.

In the speaker-dependent scenario the recognitions rates achieved with feature transformations alone were higher than the recognition rates achieved with the alternative distance measures. The VMLP was however always better than the Euclidean distance with untransformed features. This ranking was different for the cross-speaker scenario: here the alternative distance measures were better than the feature transformations. In all scenarios the best results were achieved if the VMLP was used in combination with the phoneme posteriors. This suggests that the VMLP is able to compensate a different sort of variability in the data than the phoneme posteriors.

feature transf.	distance measure	speaker-dependent		cross-speaker	
		German	French	German	French
none	Eucl. dist.	86.6	67.5	62.1	49.7
none	VMLP	90.1	73.6	75.5	61.0
none	post. scal. prod.	84.3	74.3	70.8	65.0
LDA	Eucl. dist.	92.9	77.6	69.7	55.3
LDA	VMLP	92.2	77.3	78.0	63.4
poster.	Eucl. dist.	92.5	77.2	73.7	60.3
poster.	VMLP	93.1	79.1	80.6	67.5

Table 2.2: *Evaluation of different discriminative feature transformations and distances measures in a DTW recognizer. The recognition rates in % are given for the speaker-dependent and cross-speaker IWR scenarios both for German and French.*

Cross-Language Performance

It is also important to evaluate how the feature transformations and alternative distance measures behave in a different language since they might not generalize for languages other than the language on which the VMLPs, phoneme posteriors or the linear discriminant analyses were trained. In Table 2.3 we give the results of the best performing methods in the intra-language case for cross-language experiments. A small drop of the recognition rate could be observed but the results were still much better than the ones of a DTW recognizer which used untransformed features and an Euclidean distance measure.

2.4 Concluding Remarks

We have seen that using VMLPs as a distance measure in DTW-based word recognition yields much better results than the Euclidean distance, especially in combination with phoneme posteriors as features.

feature transf.	distance measure	speaker-dependent		cross-speaker	
		German	French	German	French
none	Eucl. dist.	86.6	67.5	62.1	49.7
none	VMLP	90.5	72.5	75.6	59.7
poster.	Eucl. dist.	92.5	75.9	72.1	58.0
poster.	VMLP	92.9	76.9	79.2	63.9

Table 2.3: *Evaluation of different discriminative feature transformations and distances measures in a DTW recognizer for the cross-language case. The test languages are noted in the table and the transformations and distance measures were always trained on the other language. The recognition rates in % are given for the speaker-dependent and cross-speaker IWR scenarios both for German and French.*

To train a VMLP for the target language it is necessary to have orthographic annotations of the training data. A good property of the VMLP is that the recognition performance does not suffer very much in the cross-language case. Therefore a VMLP trained on another language can be used for languages with scarce resources.

Chapter 3

Utterance-based Word Recognition with Hidden Markov Models

A suitable HMM structure for isolated word recognition (IWR) was first suggested by [Vin71]. Each word in the vocabulary is modeled with a word HMM λ_w, which is for example a linear HMM. These word HMMs are connected in parallel to form the HMM λ which is used for recognition. The word HMM λ_w through which the optimal path as determined by a Viterbi decoder leads indicates the recognized word.

The word models λ_w can be constructed in several ways:

- The parameters of the word HMMs can be estimated individually as suggested by [Bak76] from a number of utterances of each word. This approach would in principle be applicable for our task of recognition with an utterance-based vocabulary. We have not used it since every word has to be uttered quite often in order to get reliable estimates of the parameters.

- The word HMMs λ_w can be built by concatenating sub-word

unit HMMs. The recognition network then looks as shown in Figure 3.1. The training task is divided into the estimation of the parameters of a set Φ of N sub-word HMMs $\varphi_n, n = 1, \ldots, N$ and the determination of the sub-word unit sequences $Z^{(w)} = z_1^{(w)} z_2^{(w)} \ldots z_{U^{(w)}}^{(w)}$ with $z_u^{(w)} \in \Phi$, which compose each word w.

In dictionary-based recognizers the sub-word unit HMMs correspond to linguistic units such as phonemes which can be concatenated according to a pronunciation dictionary as first suggested in [Jel76]. In our case of recognition with an utterance-based dictionary the sequences of sub-word units $Z^{(w)}$ are determined from sample utterance of the words. This method was first suggested in [BBdSP88] and [BB93].
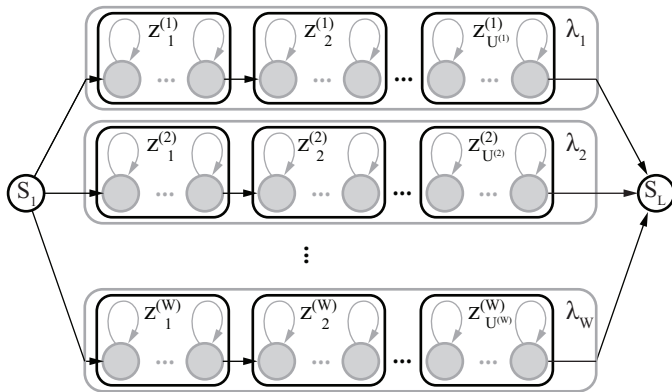
Figure 3.1: *A HMM λ which is used for word recognition. Each word HMM λ_w is concatenated from a sequence of sub-word units $Z^{(w)} = z_1^{(w)} z_2^{(w)} \ldots z_{U^{(w)}}^{(w)}$ with $z_u^{(w)} \in \Phi$.*

In this chapter we investigate the factors which are crucial for a well-performing isolated word recognizer with an utterance-based vocabulary. The first factor – the way the word models are formed from sub-word unit models – is described in Section 3.1. The second factor – the sub-word units from which the word models are formed – is described in Sections 3.3 to 3.5. Experiments to evaluate the developed techniques are presented in Section 3.6.

In this thesis we use an HMM-definition which is defined to have L states. It starts in the non-emitting state S_1 and then repeatedly transits with probability a_{ij} from a state S_i to a state S_j while emitting with probability $b_j(\mathbf{x})$ the observation $\mathbf{x}$ and finally transits with probability a_{iL} from a state S_i to the final non-emitting state S_L. Therefore, the HMM generates a sequence of T observations while visiting T times an emitting state.

3.1 Determination of the Word Models

The determination of $Z^{(w)}$ from utterances of a word boils down to the problem of finding the sequence of sub-word units which optimally describes one or several utterances of that word. In order to determine such a sequence we use a sub-word unit loop, which is a composite HMM built by connecting all elementary HMMs $\varphi_n, n = 1, \ldots, N$ in parallel with a feed-back loop. An equivalent to this fully connected HMM is shown in Figure 3.2. This equivalent HMM uses two additional non-emitting states to prevent a quadratic increase of the number of transitions with the number of sub-word units.

Especially if the elementary HMMs can be traversed while producing only one observation (e.g. if the elementary HMMs contain only one emitting state) it is quite likely that the optimal path through the HMM would entail a very long sequence of sub-word units $\hat{Z}^{(w)}$. It could then happen that the word HMMs obtained from this $\hat{Z}^{(w)}$ cannot describe shorter utterances of the same word. Therefore we favor shorter sequences $Z^{(w)}$ by penalizing transitions to a different sub-word unit with a penalty H.

3.1.1 Building a Word Model from a Single Utterance

If we have only one utterance per vocabulary word the optimal sequence of sub-word units $Z^{(w)}$ given the composite HMM as described above can be determined with the normal Viterbi algorithm.
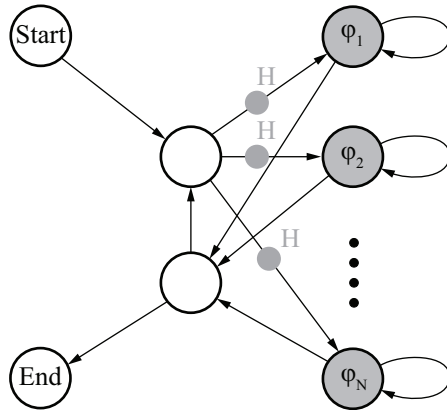
Figure 3.2: *A sub-word loop with penalties H.*

3.1.2   Building a Word Model from Several Utterances

If we have several utterances available per word in the vocabulary we are confronted with the problem of finding the sequence of sub-word units which optimally describes several utterances. In this thesis we have developed an algorithm to solve this problem which guarantees to find the optimal sequence in a maximum-likelihood sense. The algorithm solves this K-dimensional decoding problem by finding an optimal path through a (K+1)-dimensional trellis (one dimension for the states and K dimensions for the frames of the K example utterances of the word under investigation) with an extended Viterbi algorithm. This algorithm is presented in Section 3.2.2.

Since the computational complexity of the exact extended Viterbi algorithm is exponential with the number of utterances K we also developed an approximation of the extended Viterbi algorithm which starts by finding the optimal sequence of sub-word units of two utterances and then iteratively changes the resulting sequence by adding one utterance after the other. For every utterance which is added an additional two-dimensional Viterbi algorithm has to be performed. This algorithm is presented in Section 3.2.4. The complexity of this approximative

algorithm is only linear with respect to K.

3.2 Extended Viterbi Algorithm for Several Observation Sequences

For the training of the abstract acoustic elements (cf. Section 3.5.3) and to compose a word model given several utterances of a word for a word recognizer (cf. Section 3.1) we were confronted with the problem of finding a sequence of sub-word units which optimally describes several observation sequences.

For a single utterance, the problem of finding the optimal (in terms of maximum likelihood) sequence of sub-word models can easily be solved by means of the Viterbi algorithm: The sub-word HMMs are connected in parallel to form a sub-word loop and the optimal state sequence $\hat{Q}$ through the resulting HMM λ is evaluated. Then we can derive the optimal sequence of sub-word units $\hat{Z}$ from $\hat{Q}$, which we denote as: $\hat{Z} = \mathrm{SWU}(\hat{Q})$.

In contrast to this simple case, determining the sequence of sub-word models, which maximizes the joint likelihood of several utterances, leads to a non-trivial optimization problem. This problem can be stated more formally as follows: Given a set of M sub-word HMMs $\varphi_1, \ldots, \varphi_M$ and K utterances of a word, designated as $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(K)}$, find the optimal sequence of sub-word units $\hat{Z}$, i.e. the sequence of sub-word units with the highest probability to produce the utterances $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(K)}$.

Since the utterances $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(K)}$ generally are not of equal length, it is not possible to find a common state sequence for HMMs as defined earlier in this chapter (page 45).

However, our aim is not to find the optimal common state sequence for $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(K)}$, but the optimal common sequence of sub-word units $\hat{Z}$. We can formulate this optimization task more specifically as follows: we look for the K state sequences $Q^{(1)}, \ldots, Q^{(K)}$ that maximize the product of the joint probabilities $P(\mathbf{X}^{(k)}, Q^{(k)} | \lambda)$ under the condition that all state sequences correspond to the same sequence of sub-word

units. Note that λ still designates the sub-word loop mentioned above.

Thus, the problem of finding the optimal sequence of sub-word units $\hat{Z}$ can be written as:

$$\hat{Z} = \operatorname*{argmax}_{Z} \prod_{k=1}^{K} \max_{Q \in \mathcal{Q}_Z^{(k)}} P(\mathbf{X}^{(k)}, Q|\lambda) \tag{3.1}$$

where $\quad \mathcal{Q}_Z^{(k)} = \{Q \mid \mathrm{SWU}(Q) = Z \text{ and } |Q| = T_k + 2\}.$

$\mathcal{Q}_Z^{(k)}$ is the set of all state sequences that are consistent with Z and include T_k emitting states (together with the start and end states the sequences are of length $T_k + 2$).

The remainder of this section is structured as follows: First an overview of related solutions from the literature is given in Section 3.2.1. In Section 3.2.2 we describe an exact solution to the problem which is basically an extension of the Viterbi algorithm. An illustrative example of the case for two observation sequences is given in Section 3.2.3. In Section 3.2.4 we present an approximation of the Viterbi algorithm which has a much lower computational complexity. Qualitative examples of the extended Viterbi are given in Appendix B.

3.2.1   Related Work

There are several approaches to solve the optimization problem for several utterances. Most of these approaches are based on heuristics and can be roughly divided into two categories. A first category of approaches is based on generating sequences of sub-word units for each utterance and choosing the one which best matches the complete set of utterances. The simplest variant of this category is to find the best sequence of sub-word units for every utterance in isolation and choose the one which best describes the whole set of utterances. The problem with this approach is that the globally optimal sequence of sub-word units is often not among the candidates. A solution to this problem was suggested in [MJ99]. This solution employs the method described in [SH90] to generate the n best sequences of sub-word units for every utterance and chooses the best candidate from this extended set of

utterances. This increases the probability to find the optimal sequence of sub-word units but cannot guarantee to find it.

The other category of approaches is based on A* tree search. In [BB93] a method was described which chooses the best node to be evaluated next in the tree search using a heuristically determined estimate of the likelihood for the remainder of the optimal path through that node. This approach finds the optimal solution only if this likelihood is overestimated. Then, however, the tree-search algorithm is likely to be intractable. An improvement to this algorithm was presented in [SSP95]. Here the normal forward pass of Viterbi search is executed individually for each signal and the likelihoods are stored for all utterance-state-frame triples. The tree search is then performed in a backward pass, while a better estimate of the continuation likelihood of the backward path can be computed based on the stored likelihoods from the forward pass. Since this estimate is based on the forward scores of the individual paths it is still an over-estimate as it is argued in [WG99]. Finding the optimal path is therefore still based on heuristics. An approach which uses breadth-first tree-search was presented in [BN01]. This approach does not guarantee optimality either since it requires strong pruning.

3.2.2 Extension of the Viterbi Algorithm

The standard Viterbi algorithm is used to solve the decoding problem, i.e. to determine for an observation sequence $\mathbf{X} = \mathbf{x}_1 \mathbf{x}_2 \ldots \mathbf{x}_T$ and a given HMM λ with states $S_1, S_2, \ldots, S_N$ (S_1 and S_N being the non-emitting start and end states) the optimal sequence of states $\hat{Q} = S_1 \hat{q}_1 \hat{q}_2 \ldots \hat{q}_T S_N$. With the principle of dynamic programming, the joint probability of the partial observation sequence $\mathbf{X}_t = \mathbf{x}_1 \mathbf{x}_2 \ldots \mathbf{x}_t$ and the optimal partial state sequence $\hat{Q}_t = S_1 \hat{q}_1 \hat{q}_2 \ldots \hat{q}_t$ that ends in state S_j at time t

$$\delta_t(j) = \max_{\text{all } Q_t \text{ with } q_t = S_j} P(\mathbf{X}_t, Q_t | \lambda) \tag{3.2}$$

can be computed recursively with

$$\delta_t(j) = \max_{1 < i < N} \delta_{t-1}(i) \, a_{ij} \, b_j(\mathbf{x}_t). \tag{3.3}$$

$\delta_T(N)$ is the probability $P(\mathbf{X}, \hat{Q}|\lambda)$. In order to find $\hat{Q}$ the optimal precursor state has to be stored for each time and state tuple:

$$\Psi_t(j) = \operatorname*{argmax}_{1 < i < N} \delta_{t-1}(i)\, a_{ij}. \qquad (3.4)$$

The optimal state sequence $\hat{Q}$ can then be determined recursively, starting at the end. $\hat{Q}$ can be visualized as a path through a two-dimensional trellis spanned by the observations and the states.

Now we extend the Viterbi algorithm such that it finds the optimal sequence of sub-word units $\hat{Z}$ for K observation sequences, as defined by equation (3.1). The corresponding trellis has $K+1$ dimensions and the path through this trellis has to meet the two following conditions:

1. The path has to proceed in every step by zero or one in every observation sequence (i.e. in all of the K temporal dimensions in the trellis) but has to proceed in at least one observation sequence.

2. A path Q through the trellis is valid if this path and its projections $Q^{(k)}$ meet the condition: $\mathrm{SWU}(Q) = \mathrm{SWU}(Q^{(k)}) = Z$. This condition is obviously met if each transition between states that do not belong to the same sub-word model, advances a time step in all observation sequences.

The mathematical formulation of our extension to the Viterbi algorithm is as follows: The joint probability of the partial observation sequences $\mathbf{X}_{t_1}^{(1)}, \mathbf{X}_{t_2}^{(2)}, \ldots, \mathbf{X}_{t_K}^{(K)}$ and the optimal partial state sequences $\hat{Q}_{t_1}^{(1)}, \hat{Q}_{t_2}^{(2)}, \ldots, \hat{Q}_{t_K}^{(k)}$ with $q_{t_1}^{(1)} = q_{t_2}^{(2)} = \cdots = q_{t_K}^{(K)} = S_j$ is defined in equation (3.5). Of course, the optimal partial state sequences have to meet the condition that the corresponding sequences of sub-word units are identical.

Probability $\delta_{t_1,\ldots,t_K}(j)$ can be computed recursively for all points in the trellis with equation (3.6). Note that all partial paths that violate the above conditions are excluded. For backtracking the best path we have to save the optimal precursor state i like in the one-dimensional Viterbi algorithm. Additionally we need the time vector $(c_1, c_2, \ldots, c_K)$ which points from the current time point to the precursor time point. All these values are represented as a (K+1)-dimensional

vector $\Psi_{t_1,t_2,\ldots,t_K}(j) = (c_1, c_2, \ldots, c_K, i)$. This vector is determined with equation (3.7).

Let's give some additional explanations to equations (3.6) and (3.7): With $\sum c_k > 0$ the first condition is met. In order to satisfy the second condition, transitions from one sub-word unit to a another one are allowed only if $\sum c_k = K$. Otherwise the sub-word unit must not change, i.e. $SWU(S_i)$ has to be equal to $SWU(S_j)$. By using c_k as an exponent in $b_j(\mathbf{x}_{t_k}^{(k)})^{c_k}$, the probability of an observation $\mathbf{x}_{t_k}^{(k)}$ is multiplied to the total path probability if and only if the path proceeds by one observation in dimension k. With $a_{ij}^{\sum c_k}$ also the transition probability a_{ij} is multiplied to the total path probability for each observation sequence in which the path proceeds by one.

The recursion of the algorithm is initialized with

$$\delta_{1,1,\ldots,1}(j) = a_{1j}^K \prod_{k=1}^K b_j(\mathbf{x}_1^{(k)}) \tag{3.8}$$

and terminated with

$$\delta_{T_1,\ldots,T_K}(N) = \max_{1<i<N}[\delta_{T_1,\ldots,T_K}(i)a_{iN}^K] \tag{3.9}$$

$$\Psi_{T_1,\ldots,T_K}(N) = \operatorname*{argmax}_{1<i<N}[\delta_{T_1,\ldots,T_K}(i)a_{iN}^K] \tag{3.10}$$

Note that the $c_1, \ldots, c_K$ in $\Psi_{T_1,\ldots,T_K}(N)$ are not defined by equation (3.10). They are implicitly defined as zero, since the end of all observation sequences has been reached. The optimal path $\hat{Q}$ through the trellis and thus the optimal sequence of sub-word units $\hat{Z}$ can now be found by backtracking.

The resulting sequence of sub-word HMMs is guaranteed to be the optimal one, because all allowed paths through the corresponding multidimensional trellis are considered.

3.2.3 Illustrative Example

This is a simple example which illustrates the extended Viterbi algorithm for a case with two observation sequences. Given are two observation sequences $\mathbf{X}^{(1)} = 0\,1\,0\,1\,1$ and $\mathbf{X}^{(2)} = 0\,0\,0\,1$. The HMM has

$$\delta_{t_1,\dots,t_K}(j) \;=\; \max_{\substack{\text{all } Q_{t_1}^{(1)},\dots,Q_{t_K}^{(K)} \\ \text{with} \begin{cases} \mathrm{SWU}(Q_{t_1}^{(1)})=\dots=\mathrm{SWU}(Q_{t_K}^{(K)}) \\ q_{t_1}^{(1)},\dots,q_{t_K}^{(K)}=S_j \end{cases}}} P(\mathbf{X}_{t_1}^{(1)},\dots,\mathbf{X}_{t_K}^{(K)},Q_{t_1}^{(1)},\dots,Q_{t_K}^{(K)}|\lambda) \tag{3.5}$$

$$\delta_{t_1,\dots,t_K}(j) \;=\; \max_{\substack{(c_1,\dots,c_K,i) \\ 1<i<N \\ \text{with} \begin{cases} c_k\in\{0,1\} \\ \sum c_k>0 \end{cases}}} \begin{cases} \delta_{t_1-c_1,\dots,t_K-c_K}(i)\, a_{ij}^{\sum c_k}\, b_j(\mathbf{x}_{t_1}^{(1)})^{c_1}\dots b_j(\mathbf{x}_{t_K}^{(K)})^{c_K} & \text{if } (\sum c_k = K) \text{ or } \mathrm{g}(i,j) \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

$$\Psi_{t_1,\dots,t_K}(j) \;=\; \operatorname*{argmax}_{\substack{(c_1,\dots,c_K,i) \\ 1<i<N \\ \text{with} \begin{cases} c_k\in\{0,1\} \\ \sum c_k>0 \end{cases}}} \begin{cases} \delta_{t_1-c_1,\dots,t_K-c_K}(i)\, a_{ij}^{\sum c_k}\, b_j(\mathbf{x}_{t_1}^{(1)})^{c_1}\dots b_j(\mathbf{x}_{t_K}^{(K)})^{c_K} & \text{if } (\sum c_k = K) \text{ or } \mathrm{g}(i,j) \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

$$\text{where } \mathrm{g}(i,j) \;=\; \begin{cases} true & \text{if} \quad \mathrm{SWU}(S_i)=\mathrm{SWU}(S_j) \\ false & \text{otherwise} \end{cases}$$

four states: the non-emitting start and end states S_1 and S_4 and two emitting states S_2 and S_3, which have discrete observation probabilities $b_2(0) = 0.9$, $b_2(1) = 0.1$, $b_3(0) = 0.2$ and $b_3(1) = 0.8$. The non-zero transition probabilities are $a_{1,2} = a_{1,3} = \frac{1}{2}$ and $a_{2,2} = a_{2,3} = a_{2,4} = a_{3,2} = a_{3,3} = a_{3,4} = \frac{1}{3}$.

The three-dimensional trellis is illustrated in Figure 3.3. For all time points the two emitting states are shown as cubes. Printed on every cube are the partial log probabilities δ, the precursor state i and the vector $\mathbf{c}$ pointing to the selected precursor time point. The states on the optimal path $\hat{Q}$ are printed in grey.
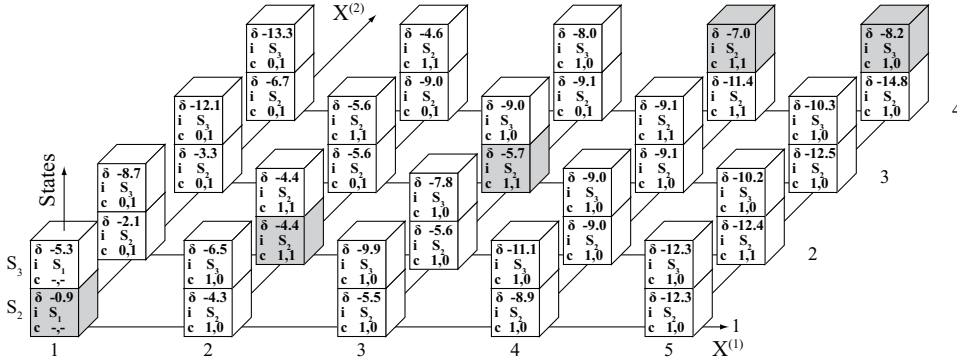
Figure 3.3: *Trellis of a 2-dimensional Viterbi example*

From this optimal path the optimal sequence of sub-word units $\hat{Z}$ is found to be $S_1 S_2 S_3 S_4$. For observation sequence $\mathbf{X}^{(1)}$ the resulting state sequence is $\hat{Q}^{(1)} = S_1 S_2 S_2 S_2 S_3 S_3 S_4$ and for $\mathbf{X}^{(2)}$ it is $\hat{Q}^{(2)} = S_1 S_2 S_2 S_2 S_3 S_4$.

It can be seen that the found path conforms to the used constraints; a state change only takes place on the transition from times $(3,3)$ to $(4,4)$, i.e. if the path proceeds on both time axes.

An example where the effect of the second constraint can be observed is $(S_3,4,2)$. Without the constraints the preceding state would

have been $(S_2,3,2)$ rather than $(S_3,3,2)$ because of the higher log like-
lihood of S_2. A state transition on the temporal transition from $(3,2)$
to $(4,2)$ is however not allowed since the path proceeds only in one
temporal dimension.

3.2.4 Approximation of the Extended Viterbi Algorithm

In the approximation to the K-dimensional Viterbi algorithm we do
not compute the optimal sequence of sub-word units at once (i.e. with
one forward and one backward pass). Rather we first perform a two-
dimensional Viterbi with any two of the K observation sequences $\mathbf{X}^{(1)}$
and $\mathbf{X}^{(2)}$. From the optimal sequence of this two-dimensional Viterbi
we build a *virtual* observation sequence $\mathbf{X}^{(v,2)}$ which represents the
two observation sequences which were processed. With this virtual
observation sequence $\mathbf{X}^{(v,2)}$ and another observation sequence $\mathbf{X}^{(3)}$ we
perform a further two-dimensional Viterbi which yields again a new
virtual observation sequence $\mathbf{X}^{(v,3)}$, which now represents the three
already processed observation sequences. We proceed in this manner
until all of the K observation sequences are included in the virtual
observation sequence. The desired sequence of sub-word units is then
the optimal sequence of sub-word units for this last virtual observation
sequence $\mathbf{X}^{(v,K)}$. Thus we perform rather $K-1$ Viterbi algorithms for
two observation sequences than one Viterbi algorithm for K observation
sequences. We now need to describe how observation sequences are
aligned and how two aligned observation sequences are merged.

Alignment of Two Observation Sequences

From the two-dimensional Viterbi algorithm we get the optimal se-
quence of sub-word units and the corresponding alignment of the pre-
vious virtual observation sequence $\mathbf{X}^{(v,k-1)}$ and the newly added ob-
servation sequence $\mathbf{X}^{(k)}$. From the alignment of $\mathbf{X}^{(v,k-1)}$ and $\mathbf{X}^{(k)}$ we
determine the new virtual observation sequence $\mathbf{X}^{(v,k)}$.

From the two-dimensional Viterbi we get the alignment as a se-
quence of sub-word units with the assigned observations from both ob-

servation sequences. This situation is depicted in Figure 3.4 b) where observations of the two sequences assigned to the same sub-word unit are printed in the same color. Within all observed sub-word units we need to align the sequence of observations which were assigned from $\mathbf{X}^{(v,k-1)}$ with those observations assigned from $\mathbf{X}^{(k)}$. Since the observations within a sub-word unit are supposed to be similar we argue that a linear alignment within a sub-word unit is enough.

The question which arises is how exactly this linear alignment should be performed and in particular how long the part of the new virtual sequence $\mathbf{X}^{(v,k)}$ which corresponds to an observed sub-word unit should be. In experiments which are not shown in this thesis we have seen that a good length of the full new virtual observation sequence $\mathbf{X}^{(v,k)}$ is chosen such that it is as long as the average length of the observation sequences contributing to the the virtual observation sequence. Therefore we calculate the length of each part of $\mathbf{X}^{(v,k)}$ corresponding to an observed sub-word unit with

$$T_{v,k} = \text{round}\left(\frac{T_{v,k-1} * (k-1) + T_k}{k}\right) \tag{3.11}$$

where $T_{v,k-1}$ is the remaining time of $\mathbf{X}^{(v,k-1)}$ within a given state, T_k the remaining time of $\mathbf{X}^{(k)}$ in that state and $T_{v,k}$ the length of the corresponding part on the new virtual observation sequence.

This implies that some parts of the observations sequences $\mathbf{X}^{(v,k-1)}$ and $\mathbf{X}^{(k)}$ have to shrink and others have to dilate. We implemented the dilation by inserting some of the observations more than once. In Figure 3.4 c) this dilation is shown as two identical observations separated with a dashed line. Shrinking is implemented by using the average of two or more observations at one time index of the virtual observation sequence. In the illustration this is depicted with a horizontal bar separating two observations at one time index of the new virtual observation sequence.

Merging of Aligned Observation Sequences

We will now explain how two aligned observation sequences are merged. Merging of observation sequences would get complicated especially if one observation sequence is a virtual observation sequence and is therefore composed of several individual observation sequences.
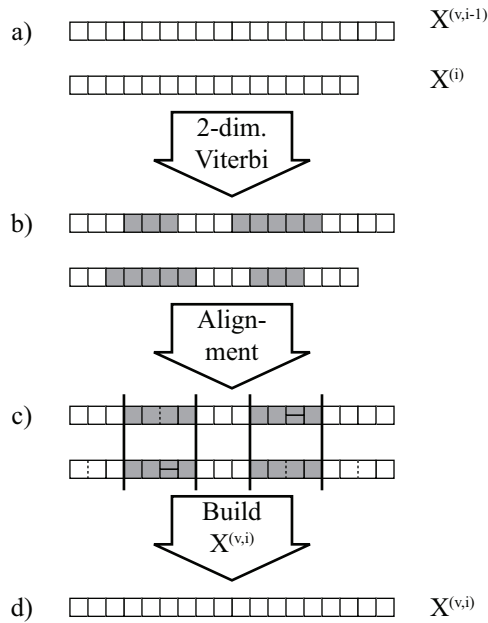
Figure 3.4: *Illustration of the merging process in the approximative K-dimensional Viterbi algorithm.*

In the core of the Viterbi algorithm we do however not need the observation sequences themselves. We rather need for every state S_n a sequence L_n which gives for every observation the log likelihood that it was produced by state S_n. We can therefore perform the merging operations analogously on all log likelihood sequences L_n, $n = 1, \ldots, N$. Merging of two aligned observation sequences thus leads to an addition of the log likelihoods in the sequences L_n.

From this process it can also be seen that the average log likelihood of subsequent virtual observation sequences gets smaller with the number of added observation sequences. This has the desired effect that the virtual observation sequence has the bigger influence than an individual observation sequence in the two-dimensional Viterbi algorithm.

This approximation of the Viterbi algorithm is computationally

much less costly. The exact K-dimensional Viterbi algorithm has a complexity of $\mathcal{O}(T^K)$ for K signals which have a mean length of T, whereas the complexity of the approximative algorithm is only $\mathcal{O}(K * T^2)$.

The way observations sequences are dilated or shrunk leads to a slightly different objective as the one of the exact algorithm. Whereas in the exact algorithm the weight of an observation sequence is proportional to the number of observations in the sequence, each observation sequence has the same weight in the approximative algorithm. This could be changed by using a different strategy of observation weighting. We have seen however, that this leads to slightly worse results for our tasks.

3.3   Appropriate Sub-Word Units

In order to build a speech recognizer with an utterance-based vocabulary which is based on HMMs we need suitable sub-word units, i.e. sub-word units which can be concatenated to model speech utterances in any language. The sub-word units should therefore fulfill the following requirements:

1. Each sub-word unit should only model the acoustics of a quasi-stationary segment of speech – an acoustic sub-space – but no temporal patterns because temporal patterns can be modeled by the concatenation of sub-word units.

2. The acoustic sub-space covered by a sub-word unit should be small enough and it should be positioned in a way that all acoustic vectors located in this sub-space are perceived as phonetically very similar.

3. The acoustic sub-spaces should be large enough to cover the variability of the same phoneme being produced by different speakers and transmitted over different channels.

4. All sub-word units together should cover the whole part of the acoustic space which occurs in the languages for which the sub-word units are used.

3.3.1 Phonemes

In dictionary-based recognizers phonemes are used as sub-word units. There the choice is given because the pronunciation dictionaries specify the pronunciation in phonemes. Phonemes, which are nowadays usually modeled with a linear HMM with three emitting states, cover segments which are not necessarily quasi-stationary. Therefore they do not meet the first requirement.

Phonemes have further disadvantages as sub-word units for utterance-derived word models. The optimal number of states may vary between the phonemes since some phonemes such as plosives are likely to have more different phases than others. Another issue is that for example begin or end phases of different phonemes might be modeled by the same statistical model because they have similar acoustic properties. This is not possible with phoneme models unless techniques such as enhanced tree clustering ([YS03]), which is an extension of tree-based state tying ([You94]), is used. Furthermore the optimal mapping of states to phoneme models may not always be the same for each speaker, an issue which is tackled with pronunciation variants or a more flexible phoneme to model mapping as for example described in [HW99].

Further considerations go in the direction of language independence. It was observed in [WTK98] that monophones perform better than triphones if they are used in a cross-language scenario. In [SW00] it was shown that monophones cover foreign languages on average better than triphones. In [Byr00] experiments with model-mapping to foreign languages showed that a mapping at the state-level performed better than a mapping at the phoneme level. These results suggest that the fourth requirement is easier to satisfy for a language-independent system if the sub-word unit models have only one state and no context.

3.3.2 Abstract Sub-Word Units

The use of abstract, data-driven sub-word units was motivated by the fact that phoneme models did not meet the requirements. Some approaches to abstract acoustic elements have been described in the lit-

erature.

A simple approach is to use the codebook classes, which are for example used in discrete-density HMMs, as acoustic units. We have not performed experiments with this approach because poor results have been reported for recognizers based on discrete-density HMMs in [Rab89] and [Har01]. An alternative presented in [HO99] is based on the information-theoretic approach of multigrams introduced in [DB95]. This approach yielded good results in a language identification task. In [EP06] abstract acoustic elements are based on quasi-stationary segments of the speech signal.

We have chosen abstract acoustic elements as described in [BB93], there called fenones, as a basis for this work. There they have however only been used for discrete-density HMMs. A fenone-variant called senone for continuous-density HMMs was presented in [HH92]. Some further suggestions for the training of abstract acoustic elements are given in [Jel98].

In this thesis we have further developed these ideas and defined abstract acoustic elements that satisfy all requirements given above.

3.4   Abstract Acoustic Elements

We now have to find a way to create a suitable set of abstract acoustic elements Φ. This means that an appropriate structure for the elementary HMMs φ_n which model the abstract acoustic elements has to be defined and a method to train their parameters has to be found. The structure of the elementary HMMs will be defined in Section 3.4.1. Before describing the training of abstract acoustic elements we will look at the training of phonemes in Section 3.4.2 because there is much more knowledge available to train phonemes than to train abstract acoustic elements. Based on this we will then formulate a training scheme for abstract acoustic elements in Section 3.4.3.

3.4.1 Structure of Abstract Acoustic Elements

According to the first requirement an abstract acoustic element should not define temporal properties since it covers a quasi-stationary segment of a speech signal. Therefore the elementary HMM φ_n which describes an abstract acoustic element A_n is composed of only one emitting state and has fixed transition probabilities ($a_{22} = a_{23} = 0.5$, $a_{12} = 1$), resulting in the structure shown in Figure 3.5. Each $\varphi_n, n = 1 \ldots, N$ is thus defined only through the probability density function $b_n(\mathbf{x})$, which is used to calculate the observation likelihood of an observation $\mathbf{x}$, given the abstract acoustic element A_n.

From now on we will refer to the abstract acoustic element A_n with the elementary HMM φ_n which is used to model it and to a set of abstract acoustic elements $\mathcal{A}$ with the set of elementary HMMs Φ.
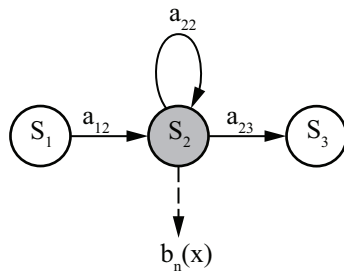
Figure 3.5: *Structure of an HMM φ_n, which is used to model an abstract acoustic element. The only emitting state S_2 is printed in grey. The production likelihood $b_n(\mathbf{x})$ of an observation $\mathbf{x}$ is defined with a GMM.*

$b_n(\mathbf{x})$ is defined with a Gaussian mixture model (GMM) which has the form

$$b_n(\mathbf{x}) = \sum_{m=1}^{M} c_{nm} \, \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_{nm}, \boldsymbol{\Sigma}_{nm}), \qquad (3.12)$$

where c_{nm}, $\boldsymbol{\mu}_{nm}$, and $\boldsymbol{\Sigma}_{nm}$ are the weights, the mean vectors and the covariance matrices of the M mixture components, respectively. Since

the features used in the experiments (see Appendix C.1) are supposed not to be very much correlated, we use diagonal covariance matrices $\mathbf{\Sigma}_{nm}$.

3.4.2 Training of Phonemes as a Starting Point

Before explaining the training of abstract acoustic elements we outline the training mechanisms which are normally used to build phoneme models. The set of elementary HMMs $\Phi = \{\varphi_1, \ldots, \varphi_N\}$ models the phonemic alphabet. The elementary HMMs are nowadays usually linear HMMs with three emitting states.

Usually the sequence of phonemes which was spoken in an utterance (i.e. a training utterance) is known through orthographic annotations and a pronunciation dictionary. Therefore a composite HMM λ_k can be built for every training utterance $\mathbf{X}^{(k)}$.

The optimization criterion which is normally used to determine the parameters of elementary HMMs $\varphi_n \in \Phi$, is the maximization of the likelihood for K observation sequences $\mathcal{X} = \{\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(K)}\}$:

$$P(\mathcal{X}|\Phi) = \prod_{k=1}^{K} P(\mathbf{X}^{(k)}|\lambda_k) \tag{3.13}$$

There is no analytical solution to solve this optimization problem. With the Baum-Welch algorithm ([Bau70]) there is however a method to iteratively determine the optimal parameters.

An alternative method to find the set of parameters is the *segmental k-means* algorithm (see [Rab86],[JR90]), which is also known as Viterbi training. Here the objective is to maximize the joint probability of the set of observation sequences $\mathcal{X}$ and the set of optimal state sequences $\hat{\mathcal{Q}} = \{\hat{Q}^{(1)}, \ldots, \hat{Q}^{(K)}\}$:

$$P(\mathcal{X}, \hat{\mathcal{Q}}|\Phi) = \prod_{k=1}^{K} P(\mathbf{X}^{(k)}, \hat{Q}^{(k)}|\lambda_k) \tag{3.14}$$

This training method is based on two steps. In the first step the optimal state sequences $\hat{Q}^{(k)}$ are determined for each observation sequence

$\mathbf{X}^{(k)}$. For continuous-density HMMs like in our case also the optimal sequence of used mixtures $\hat{G}^{(k)}$ (the sequence which indicates for every chosen state on $\mathbf{X}^{(k)}$ the best mixture) has to be determined. With this first step an unambiguous assignment of observations to mixtures in the HMM states is performed. In the second step the statistics can be computed for all mixtures individually and the parameters can be updated. Also the new estimate of the transition probabilities can be computed from the alignment determined in the first step.

In [JR90] it was shown that the production likelihood of the training data for a set of phoneme models Φ in a given iteration is at least as hight as the likelihood in the previous iteration. Furthermore in [ME91] it was both theoretically and experimentally shown that Viterbi training leads to similar results as Baum-Welch training for the typical settings in speech recognition.

Difference to the Training of Abstract Acoustic Elements

In the phoneme training the classes into which the acoustic space has to be partitioned are known prior to model training. The training procedure thus has to align the observation sequences $\mathbf{X}^{(k)}$ appropriately to the corresponding composite HMMs λ_k and estimate the parameters.

In the case of abstract acoustic elements the training is more complicated since the classes into which the acoustic space is divided are not known in advance but have to be estimated together with the model parameters.

3.4.3   Training Procedure of Abstract Acoustic Elements

We use a training procedure which determines the parameters c_{nm}, $\boldsymbol{\mu}_{nm}$, and $\boldsymbol{\Sigma}_{nm}$ for all elements n and mixtures m by iterating between the assignment of all feature vectors to the states of the abstract acoustic elements vectors and the parameter reestimation. The assignment is defined by means of state sequences $\tilde{\mathcal{Q}} = \{\tilde{Q}^{(1)}, \ldots, \tilde{Q}^{(K)}\}$ for all utterances $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(K)}$. How the state sequences $\tilde{\mathcal{Q}}$ are determined will be described later. The parameters are reestimated by maximizing

the joint probability of the observation sequences and the given state sequences $P(\mathcal{X}, \tilde{\mathcal{Q}}|\Phi)$. This criterion is quite similar to the criterion in equation (3.14) used in Viterbi training, but $\tilde{\mathcal{Q}}$ is used instead of $\hat{\mathcal{Q}}$.

The training of abstract acoustic elements follows a training flow as shown in Figure 3.6. It consists of three loops. In the innermost loop the parameters are reestimated while $\tilde{\mathcal{Q}}$ is not changed. In the next outer loop $\tilde{\mathcal{Q}}$ is redetermined. In the outermost loop the number of mixtures is increased. This loop starts directly with a reestimation of the parameters since a determination of $\tilde{\mathcal{Q}}$ does not make sense directly after splitting the mixtures.

The properties of abstract acoustic elements are therefore determined through the parameters of this training flow. The following parameters need to be defined:

initial Φ	the set of initial elementary HMMs
$\tilde{\mathcal{Q}}$ determination	the algorithm with which $\tilde{\mathcal{Q}}$ is determined
R_1	the number of reestimations which is performed without changing $\tilde{\mathcal{Q}}$
R_2	the number of redeterminations of $\tilde{\mathcal{Q}}$ for a given number of mixtures
M	the final number of mixtures in each abstract acoustic element

How the initial models are constructed will be described in Section 3.4.4. The reestimation of the parameters will be described in Section 3.4.5. The main difference between different abstract acoustic elements types results from the way $\tilde{\mathcal{Q}}$ is determined. Different abstract acoustic element types and their method to choose $\tilde{\mathcal{Q}}$ are presented in Section 3.5.

3.4.4 Initial Models

The initial models are determined with a clustering of the acoustic space. The whole acoustic space is partitioned in as many clusters as abstract acoustic elements are desired.
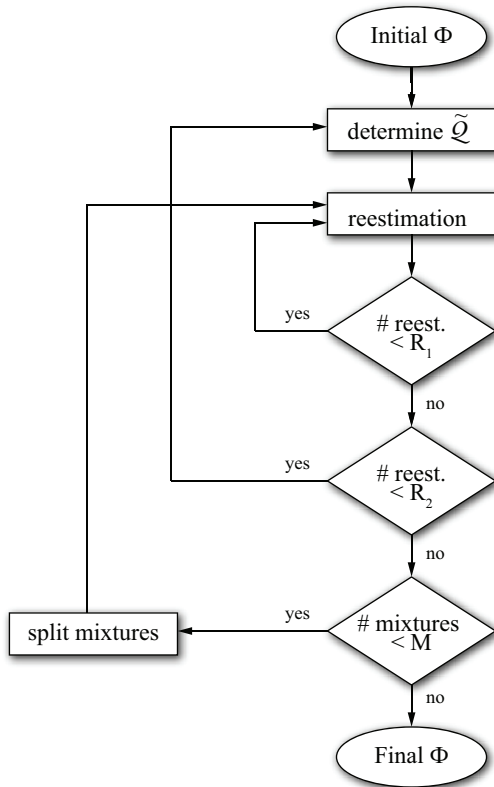
Figure 3.6: *Flow chart of the training process of abstract acoustic elements.*

Clustering algorithms are used in the construction of vector quantizers. There a codebook vector is used to represent a cluster of vectors. The vector quantizer assigns an input vector to the cluster whose codebook vector is closest. In the domain of speech, vector quantization is used for speech coding (see for example [MRG85]) or for HMMs with discrete-valued observation sequences ([RLS83]). Given a set of training vectors, the clustering algorithms aim at choosing the codebook vectors such that the distortion within each cluster (i.e. the distortion

of all training vectors which are assigned to a codebook vector) is minimized. A very popular algorithm is the *Linde-Buzo-Gray* or short *LBG* algorithm, which was introduced in [LBG80].

Using the LBG algorithm with the squared-error distortion measure d we partition the acoustic space into the number of desired abstract acoustic elements N. A GMM with a single mixture is then taken as initial model for each abstract acoustic element φ_n. Mean and variance of the mixture are determined from the observation vectors which were assigned to codebook vector $\mathbf{z}_n$.

3.4.5 Parameter Reestimation

All methods to train abstract acoustic elements, which will be described in Section 3.5, use the same way to update the model parameters according to the collected statistics. The algorithms described in Section 3.5 assign each frame $\mathbf{x}_t^{(k)}$ to a state. This state is given with $\tilde{q}_t^{(k)}$. The algorithms do not determine the mixture within the states. Always the most likely mixture is taken in every state like in the Viterbi training of phonemes. This mixture is therefore termed $\hat{g}_t^{(k)}$.

For the parameter reestimation the following auxiliary variables are defined for all observation sequences k, times t, states S_n and mixtures m:

$$\gamma_t^{(k)}(n) = \begin{cases} 1 & \text{if } \tilde{q}_t^{(k)} = S_n, \\ 0 & \text{otherwise} , \end{cases} \tag{3.15}$$

$$\zeta_t^{(k)}(n,m) = \begin{cases} 1 & \text{if } \tilde{q}_t^{(k)} = S_n \text{ and } \hat{g}_t^{(k)} = m, \\ 0 & \text{otherwise} . \end{cases} \tag{3.16}$$

The new parameters can then be reestimated in every iteration with

the following formulas:

$$\tilde{c}_{nm} = \frac{\sum_{k=1}^{K}\sum_{t=1}^{T_k}\zeta_t^{(k)}(n,m)}{\sum_{k=1}^{K}\sum_{t=1}^{T_k}\gamma_t^{(k)}(n)} \tag{3.17}$$

$$\tilde{\boldsymbol{\mu}}_{nm} = \frac{\sum_{k=1}^{K}\sum_{t=1}^{T_k}\zeta_t^{(k)}(n,m)\,\mathbf{x}_t^{(k)}}{\sum_{k=1}^{K}\sum_{t=1}^{T_k}\zeta_t^{(k)}(n,m)} \tag{3.18}$$

$$\tilde{\boldsymbol{\Sigma}}_{nm} = \frac{\sum_{k=1}^{K}\sum_{t=1}^{T_k}\zeta_t^{(k)}(n,m)(\mathbf{x}_t^{(k)} - \tilde{\boldsymbol{\mu}}_{nm})(\mathbf{x}_t^{(k)} - \tilde{\boldsymbol{\mu}}_{nm})^t}{\sum_{k=1}^{K}\sum_{t=1}^{T_k}\zeta_t^{(k)}(n,m)} \tag{3.19}$$

3.5   Abstract Acoustic Element Types

In this Section we present three different types of abstract acoustic elements. The training protocol of all types corresponds to the training flow shown in Section 3.4.3, but differs in the way $\tilde{\mathcal{Q}}$ is determined.

The first two types of abstract acoustic elements do not require any annotation of the training data. The last type presented in Section 3.5.3 requires orthographic annotations.

3.5.1   Purely Acoustic Clustering Based on the LBG Algorithm

The first approach to the construction of a set of abstract acoustic elements is purely acoustical and therefore completely unsupervised. The idea of this approach is to keep the acoustic clustering which was

done during the model initialization explained in Section 3.4.4 during the whole training process. Each cluster is modeled with a GMM.

Using the terminology introduced in Section 3.4.3 the elements of the sequences $\tilde{Q}^{(k)}$ are determined as follows:

$$\hat{n} = \operatorname*{argmin}_{1 \leq n \leq N} d(\mathbf{x}_t^{(k)}, \mathbf{z}_n) \tag{3.20}$$

$$\tilde{q}_t^{(k)} = \varphi_{\hat{n}} \tag{3.21}$$

The parameter R_2 is in this case 1 and therefore $\tilde{Q}$ will never change during the process. This also means that the GMM for each cluster and therefore for each abstract acoustic elements is modeled independently of the other clusters.

3.5.2 Acoustic Clustering Optimized for GMMs

The approach described in Section 3.5.1 has the drawback that the shapes of the clusters are still restricted to the shape which was given by the Voroni tessellation determined by the codebook vectors even though GMMs are capable to model more complex distributions. The approach presented in this section allows the clustering of the acoustic space to change with the increased modeling flexibility of the GMM if the number of mixtures is increased.

The state sequences $\tilde{Q}$ which determine the assignment of observations to abstract acoustic elements are in this case for every observation sequence $\mathbf{X}^{(k)}$ determined like in Viterbi training as the optimal state sequence $\hat{Q}^{(k)}$ which can be determined with the Viterbi algorithm:

$$\tilde{Q}^{(k)} = \hat{Q}^{(k)} = \operatorname*{argmax}_{Q^{(k)}} P(Q^{(k)}, \mathbf{X}^{(k)}|\lambda) \tag{3.22}$$

The composite HMM λ is here an abstract element loop composed of the elementary HMMs from Φ which describe the abstract acoustic elements $\varphi_n, n = 1, \ldots, N$. Note that in contrast to the usual training of phonemes summarized in Section 3.4.2 this composite HMM λ is the same for all training utterances.

The state sequences $\tilde{Q}$ are periodically updated. Therefore the clusters can gradually adapt to the modeling capability of GMMs which

grows with the number of mixtures used to model each abstract acoustic element.

There is no analytically proven guarantee that this training procedure, nor any of the ones described in Section 3.5.3 will converge to a good clustering of the acoustic space. That the clustering is reasonable has to be shown experimentally.

Favor Longer Sequences of the Same Element

Speech is often considered as a quasi-stationary process since the vocal tract of a speaker does not make arbitrarily fast movements. Therefore the properties of the signal are usually considered as stationary within a speech frame. We also expect consecutive frames to be similar. With some modification to the approach presented in this section we aim at making the abstract acoustic elements less sensitive to small fluctuations of the features. In other words the decoder which uses the abstract acoustic elements should remain as long as possible in one abstract acoustic element.

In order to achieve this we use again an abstract acoustic element loop λ but impose a fixed additional penalty H on all transitions leading to a different abstract element. Such an HMM was shown in Figure 3.2. With this additional penalty in the HMM the state sequences $\tilde{\mathcal{Q}}$ are determined according to equation (3.22) as well.

3.5.3   Use of Orthographic Annotations

All previously described approaches did not use any annotation of the training data. It is interesting to evaluate whether knowledge of the content of the training material may be used to create better abstract acoustic elements.

In [Jel98] a method is proposed, which assumes that it is known for every training utterance, which word it contains. All training utterances containing the same word are then supposed to be described by the same sequence of abstract acoustic elements. We developed an approach similar to the one presented in [Jel98], with modifications

necessary mainly since the HMMs used in [Jel98] allow non-emitting transitions.

The K training utterances are therefore distributed into groups according to the word which they contain. Therefore we have as many groups W as we have words in the training material and each group contains the utterances which contain the corresponding word. Here we describe every group with the set of $V_w, w = 1 \ldots, W$ of utterance indices which contain the word w. Now the sequence of abstract acoustic elements $\hat{Z}^{(w)}$ which produces the observation sequences $\mathbf{X}^{(k)}, k \in V_w$ with highest probability has to be computed for every word w. In Section 3.2 this optimization problem and its solution are described. Now a linear HMM λ_w can be built by concatenating elementary HMMs $\varphi_n \in \Phi$ in the same order as they occur in $\hat{Z}^{(w)}$ for every word. The state sequence $\tilde{Q}^{(k)}$ can then be determined for every training utterance:

$$\tilde{Q}^{(k)} = \underset{Q^{(k)}}{\operatorname{argmax}} P(Q^{(k)}, \mathbf{X}^{(k)} | \lambda_w) \qquad (3.23)$$

This equation is almost identical to equation (3.22), but the λ_w is specific for the word the training utterance k belongs to.

3.6   Experiments

3.6.1   Used Training Parameters and Training Data

The parameters introduced in Section 3.4.3 to control the training process were as follows: R_1 was 8 and R_2 was 4 for all abstract acoustic elements with the exception of the abstract acoustic elements described in 3.5.1. Here R_1 was 32 and R_2 was 1. We have empirically chosen these parameters in a way that the average log likelihood per frame on the training data did not increase further when more iterations were used.

All sub-word units used in this thesis were trained on data of speakers of the speaker set $\mathcal{S}_{G,poly,1}$ for German and $\mathcal{S}_{F,poly,1}$ for French. These speaker sets are disjoint from the speaker sets $\mathcal{S}_{G,poly,3}$ and $\mathcal{S}_{F,poly,3}$ which were used in the test tasks.

3.6.2   Comparison of Different Sub-Word Units

In Section 3.5 we have presented different methods to train abstract acoustic elements. In this Section we evaluate the performance of the abstract acoustic elements generated by the described methods. In all experiments we kept the used number of elements N at 128 and the number of mixtures per element M at 16. These numbers yield about the same number of trainable parameters as for the training of phonemes. The influence of these parameters will be described in Section 3.6.5. We used the features $Feat_{noCms}$ as described in Appendix C.1.

In Table 3.1 we show the results for the German and the French ten-word IWR task in a speaker-dependent scenario (tasks 1 and 4). In these experiment we used only one utterance to construct each word HMM λ_w. $\mathcal{A}_{LBG}$ are the elements which perform the clustering only with the LBG algorithm as described in Section 3.5.1. The elements $\mathcal{A}_{Free}$ are elements trained with the unsupervised clustering described in Section 3.5.2. Here we used an additional log-likelihood of -2 as a penalty on transitions to other elements. Orthographic annotations were necessary to train the elements $\mathcal{A}_{WordConstrain}$ as described in Section 3.5.3. Additionally we also evaluated phonemes as described in Appendix D.

The ranking of the systems was the same for both languages even though the recognition rates were considerably lower for the French test tasks. The reason is most probably the shorter average duration of the French words. The abstract acoustic elements $\mathcal{A}_{LBG}$ had by far the worst performance. This clearly showed that a clustering based only on the LBG algorithm is not sufficient. The abstract acoustic elements $\mathcal{A}_{Free}$ already yielded much better results but were still worse than phonemes. They were however still interesting since they require no annotation of the training data. The abstract acoustic elements $\mathcal{A}_{WordConstrain}$ showed the best performance, also better than phonemes.

sub-word unit type	German	French
$\mathcal{A}_{LBG}$	65.3 %	55.7 %
$\mathcal{A}_{Free}$	85.7 %	71.6 %
$\mathcal{A}_{WordConstrain}$	93.3 %	86.6 %
phonemes	90.9 %	84.8 %

Table 3.1: *Recognition rates for different sub-word unit types in a speaker-dependent scenario. All tests are performed with sub-word models of the target language.*

3.6.3 Language-Independence of Abstract Acoustic Elements

In contrast to training of phoneme models there is no pronunciation dictionary necessary to train abstract acoustic elements for a given language or dialect. If abstract acoustic elements have to be trained for the target language it is however still necessary to have acoustic training data of this language and for abstract acoustic elements of type $\mathcal{A}_{WordConstrain}$ additionally orthographic annotations are necessary.

It would therefore be desirable if abstract acoustic elements could be used for a cross-language scenario, i.e. that abstract acoustic elements which were trained on one or several languages could be used in a recognizer of a language not used for the training.

Therefore we evaluated which sub-word unit type is most appropriate to be used in a cross-language scenario. We tested phonemes, abstract acoustic elements of type $\mathcal{A}_{WordConstrain}$ and abstract acoustic elements of type $\mathcal{A}_{Free}$ in the German and French tasks of a speaker-dependent scenario (tasks 1 and 4). The results are given in Table 3.2. *Intra-language* means that tests were performed with sub-word models of the same language. *Cross-language* means that French sub-word models were used in the German test task and vice versa. The relative increase of the error rate when switching from language-dependent models to cross-language models is shown in Table 3.3.

The first fact to notice is that the decrease in performance if models from the other language are used was much higher for the French test task than for the German test task. For the German test task the

sub-word unit type	German		French	
	intra-language	cross-language	intra-language	cross-language
$\mathcal{A}_{Free}$	85.7 %	85.0 %	71.6 %	65.7 %
$\mathcal{A}_{WordConstrain}$	93.3 %	91.5 %	86.6 %	76.3 %
phonemes	90.9 %	86.0 %	84.8 %	71.5 %

Table 3.2: *Recognition rates for different sub-word unit types in a speaker-dependent scenario. Results for intra-language and cross-language tests are given.*

average relative increase of the error rate was only 4.9 % whereas it was 20.1 % for the French task. This indicates that the German language was much better covered by the French models than the other way round. At the first glance this is astonishing since the German phoneme-based recognizer had 47 phoneme models whereas the French recognizer had only 40. If we look closer at the German phoneme models we see that there are a lot of diphthongs and affricates which may be substituted by two phoneme models (e.g. 'a͜u' or 't͜s'). Furthermore there are a lot of vowel-phoneme models which occur in a long and a short version (e.g. 'u' and 'uː'). On the other side French has a lot of nasals which do not occur in German.

sub-word unit type	German	French	average
$\mathcal{A}_{Free}$	4.9 %	20.1 %	12.5 %
$\mathcal{A}_{WordConstrain}$	26.9 %	76.9 %	51.9 %
phonemes	53.9%	87.5 %	70.7 %
average	28.7 %	61.5 %	45.0 %

Table 3.3: *Relative increase of the error rates for different sub-word unit types if using cross-language instead of language-dependent sub-word units.*

Now we look at the different sub-word unit types. As expected the performance of the phoneme models degrades most if phoneme models of another language are used instead of phoneme models of the target language. The abstract acoustic elements $\mathcal{A}_{WordConstrain}$ show a bigger performance loss in cross-language tests compared to intra-

language tests than the elements $\mathcal{A}_{Free}$. This is probably due to a language dependence which is introduced into the models by forcing all utterances of a word to the same sequence of abstract acoustic elements Z during the training.

Looking again at the absolute recognition rates, the elements $\mathcal{A}_{WordConstrain}$ were still much better than the elements $\mathcal{A}_{Free}$. Furthermore, the superiority over phonemes was bigger in cross-language tests than in intra-language tests.

3.6.4   Influence of Transition Penalties

In Section 3.5.2 we have described how transition penalties can be used to force the Viterbi decoder to remain longer in an abstract acoustic element. This can be used during the training of the abstract acoustic elements but also during the concatenation of sub-word units to a word HMM from utterances as described in 3.1.

In this section we report results of the influence of the penalties H_{train} used during the abstract acoustic element training and H_{test} used during the construction of the word models. We tested the influence in the two families of abstract acoustic elements $\mathcal{A}_{Free}$ and $\mathcal{A}_{WordConstrain}$, as defined in Section 3.6.2. The penalties are implemented in the HMMs by subtracting the value H_{train} or H_{test} for each transition between abstract acoustic elements.

All parameters except for the penalties were constant: The number of abstract acoustic elements was 128 and the number of mixtures per element was 16. The used features $Feat_{noCms}$ are described in Appendix C.1.

The resulting recognition rates are shown in Figure 3.7 for the German speaker-dependent task (task 1) and in Figure 3.8 for the French speaker-dependent task (task 4). In order to distinguish the influence of H_{train} from H_{test} we ran the experiments twice. In the first runs we changed both H_{train} and H_{test}. In the second runs we did not apply a penalty during training but only during testing. These are the plots denoted as "$H_{train} = 0$". For German we also printed the average log likelihood per frame during the training and the average segment

length, i.e. the average number of frames during which the decoder remained in the same abstract acoustic element.

We could observe that it was crucial to have appropriately chosen penalty values H_{train} and H_{test} for $\mathcal{A}_{Free}$. If varying only H_{test} and keeping $H_{train} = 0$ the results were constantly worse than if penalties were used both during training and testing. This demonstrated that the quality of the abstract acoustic elements with respect to our task was better if a penalty H_{train} was used. In the case of $\mathcal{A}_{WordConstrain}$ the values of H_{train} and H_{test} could be chosen much smaller. This showed that using several utterances to determine a sequence of abstract acoustic elements Z leads to better results and that a high penalty on transitions to a different sub-word unit was not necessary.

The average log likelihood per frame during training did not decrease very much with an increased segment length. It was however much lower if all utterances of the same word were forced on the same sequence of abstract acoustic elements.

3.6.5 Suitable Number of Abstract Acoustic Elements and Mixtures

So far we have not observed the influence of the number of abstract acoustic elements N and the number of mixtures per element M on the recognition rate. Therefore we trained and tested abstract acoustic elements of the $\mathcal{A}_{WordConstrain}$ family to find the number of elements which is suitable to model speech.

The results of a speaker-dependent scenario are given in Figure 3.9 for German (task 1) and in Figure 3.10 for French (task 4). Each line corresponds to a set of abstract acoustic elements with a given size. The recognition rate is then given as a function of the number of mixtures in each element.

The results showed that it is especially important to have enough abstract acoustic elements. Using more than 128 elements did however not improve the recognition rate a lot but increased the recognition time. This number of abstract acoustic elements is much more than the number of phonemes which are usually used in monophone-
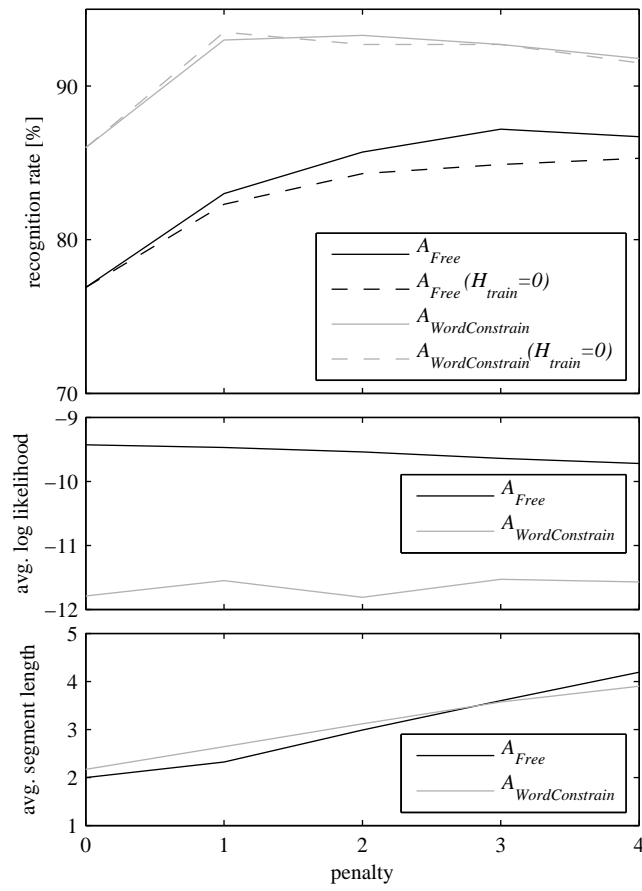
Figure 3.7: *Influence of transition penalties on the recognizer for a German speaker-dependent scenario. The effect on the recognition rate, the average log likelihood during training and the average segment length in frames during training is shown as a function of the penalty. The abstract acoustic element types $\mathcal{A}_{WordConstrain}$ and $\mathcal{A}_{Free}$ are tested twice. In the tests denoted "$H_{train}=0$" only H_{test} is varied while H_{train} remains 0. In the other tests H_{test} and H_{train} were varied.*
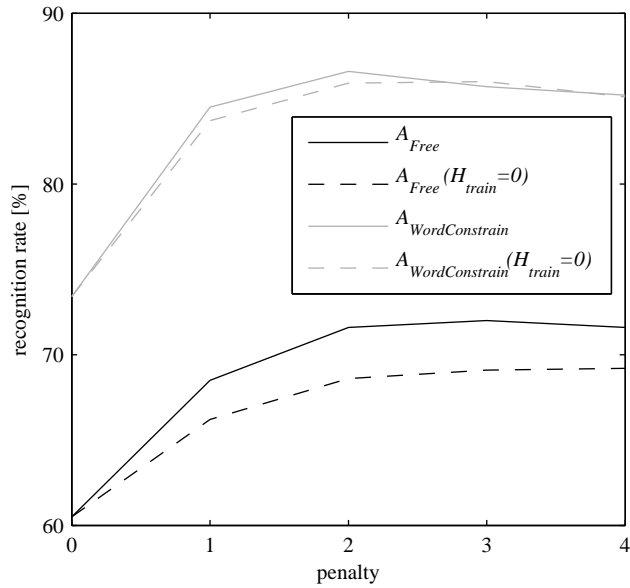
Figure 3.8: *Influence of transition penalties on the recognition rate of a French speaker-dependent scenario. The abstract acoustic element types $\mathcal{A}_{WordConstrain}$ and $\mathcal{A}_{Free}$ are tested twice. In the tests denoted "$H_{train}=0$" only H_{test} is varied while H_{train} remains 0. In the other tests H_{test} and H_{train} were varied.*

based recognizers (our German recognizer has 47 phonemes and the French recognizer has 40). If we consider however, that each phoneme is composed of three states, the number of Gaussian mixture models in monophone-recognizers is quite close to 128.

3.6.6   Comparison of Algorithms to Find a Sequence of Sub-Word Units from Several Utterances

In Section 3.1.2 we sketched two algorithms to determine the optimal sequence of sub-word units for several utterances and described them in detail in Section 3.2. We will now evaluate these algorithms.
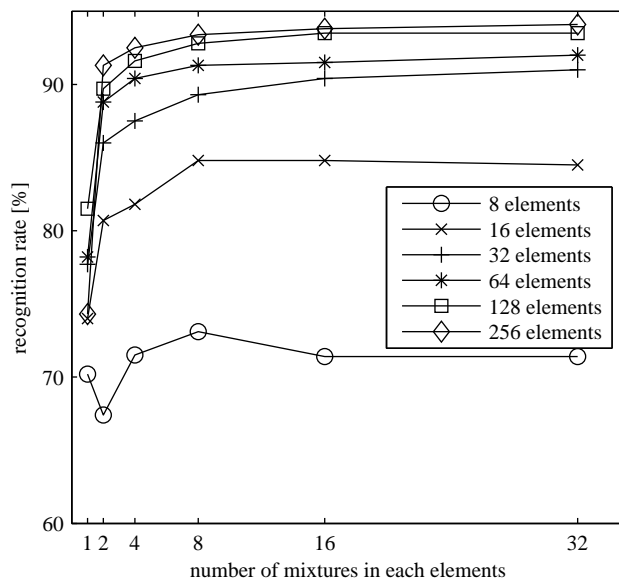
Figure 3.9: *Recognition rate for German in a speaker-dependent scenario with different numbers of abstract acoustic elements and mixtures.*

In the following we will refer to the exact algorithm which is presented in Section 3.2.2 as Alg_{exact} and to the approximative algorithm presented in Section 3.2.4 as Alg_{approx}. All tests were performed with abstract acoustic elements of type $\mathcal{A}_{WordConstrain}$.

Compared Algorithms

Additionally to the algorithms Alg_{exact} and Alg_{approx} we tested two alternative algorithms to determine the best sequence of sub-word units:

- $Alg_{chooseBest}$: This algorithm falls in the first category of algorithms described in Section 3.2.1. It was implemented along the lines of [MJ99]. We determined for each utterance the 20 best sequences of abstract acoustic elements with the token-passing-
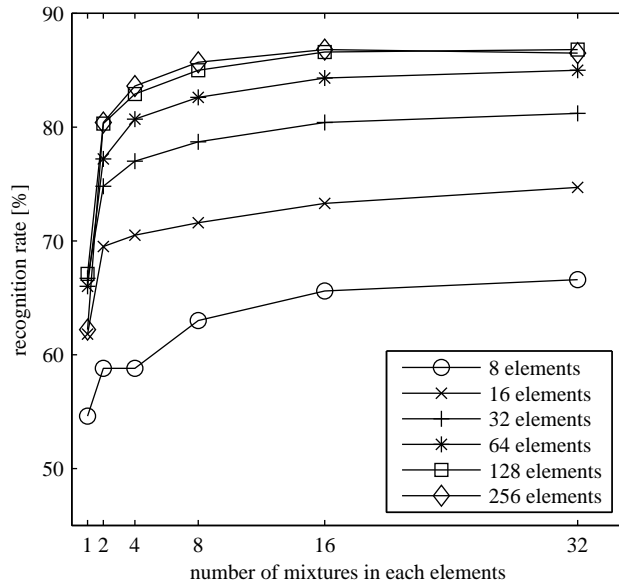
Figure 3.10: *Recognition rate for French in a speaker-dependent scenario with different numbers of abstract acoustic elements and mixtures.*

based Viterbi algorithm (see [You89]) with ten tokens per node. From the resulting sequences, we used that as reference, which best described all reference utterances.

- $Alg_{DtwAlign}$: In this approximative algorithm the available reference utterances of a vocabulary word were aligned with DTW in order to get a time-synchronous version of the sequences. All sequences were mapped on the sequence which had the smallest DTW-distance to all others. For these time-aligned sequences it was straight-forward to perform a Viterbi search by using the joint observation probability of the aligned feature vectors to find the best sequence of abstract acoustic elements.

Tests in Speaker-Dependent IWR

The first experiments show how the different algorithms performed in a speaker-dependent scenario (tasks 1 and 4, cf. Appendix E.2). In Table 3.4 the results of several algorithms are listed as a function of the number of utterances which were used to form each word model. The data available for the speaker-dependent scenario allowed to use only up to three utterances per vocabulary word.

language	German			French		
# utterances	1	2	3	1	2	3
Alg_{exact}		96.2	97.0		93.4	95.2
Alg_{approx}	93.5	96.2	96.8	86.6	93.4	94.8
$Alg_{chooseBest}$		95.6	96.4		90.9	92.8
$Alg_{DtwAlign}$		94.7	95.0		89.6	91.9

Table 3.4: *Evaluation of different algorithms to find the optimal sequence of sub-word units for several utterances in a speaker-dependent scenario. Recognition rates are given in %.*

Tests in Speaker-Independent IWR

In a speaker-independent scenario (tasks 3 and 6, cf. Appendix E.2) we could also evaluate how the algorithms perform with more utterances per vocabulary word. The recognition rates as a function of the number of available utterances per reference word are given in Figure 3.11. We evaluated the same algorithms as in the speaker-dependent case except for the algorithm Alg_{exact}, which had a too high computational complexity to be applied for 20 utterances.

A comparison of the algorithms showed that Alg_{exact} yielded better results than $Alg_{chooseBest}$ and $Alg_{DtwAlign}$. Moreover Alg_{approx} yielded almost as good results as Alg_{exact}. As expected the recognition rate with $Alg_{chooseBest}$ increased with the number of available utterances. Its recognition rate was however also with 20 utterances per reference word still considerably lower than the one with Alg_{approx}. It is a welcome property of the Alg_{approx} systems that the recognition rates are
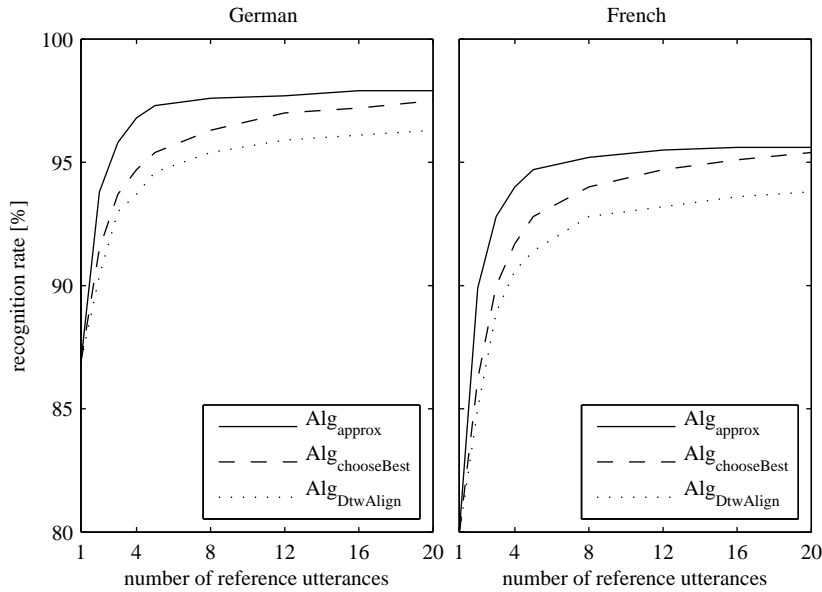
Figure 3.11: *Recognition rate as a function of the number of reference utterances in a speaker-independent scenario (tasks 7 and 8).*

already with a few utterances quite high, since a smaller effort is necessary to record all utterances.

Another important aspect is the computational complexity of the algorithms. The exact algorithm Alg_{exact} has a computational complexity of $\mathcal{O}(T^K)$ for K utterances of average length T. The algorithms $Alg_{chooseBest}$ and $Alg_{DtwAlign}$ have a computational complexity of $\mathcal{O}(T^2)$. The algorithm Alg_{approx} is linear with respect to T.

3.7 Concluding Remarks

We have shown that abstract acoustic elements are more suitable subword units than phonemes to build word HMMs by concatenating sub-

word units according to utterances of the words. The superiority of abstract acoustic elements is particularly evident in a cross-language scenario.

We could also show that the K-dimensional Viterbi and the approximation thereof which were presented in Section 3.2 are better for the task of determining the optimal sequence of abstract acoustic elements than other methods described in the literature.

Abstract acoustic elements $\mathcal{A}_{WordConstrain}$ for which orthographic annotations are necessary perform considerably better than abstract acoustic elements $\mathcal{A}_{Free}$ for which no annotations are necessary. This is disadvantageous for languages for which only unannotated training data is available.

Even though the language mismatch had a smaller impact on the recognition rate when abstract acoustic elements were used than when phonemes were used, it still resulted in a performance loss.

We have seen however that abstract acoustic elements $\mathcal{A}_{WordConstrain}$ which are trained on annotated training data of another language still performed better than abstract acoustic elements $\mathcal{A}_{Free}$ which are trained on data of the target language.

Chapter 4

Comparison of Different Isolated Word Recognition Techniques

This chapter compares the best-performing systems both of the DTW-approach described in Chapter 2 and of the HMM-approach described in Chapter 3 in speaker-dependent, cross-speaker and speaker-independent scenarios. All scenarios are compared for the intra-language case in which the training material was taken from the target language and for the cross-language case in which the training material was taken from another language. In Section 4.2 the recognizers with an utterance-based vocabulary are compared with a state-of-the-art dictionary-based recognizer.

4.1 Comparison of Recognizers with an Utterance-based Vocabulary

Two recognizers were based on DTW pattern matchers as described in Section 2. DTW_{Eucl} used a standard Euclidean distance measure

whereas DTW_{VMLP} used the posterior-feature transformation and a VMLP as a distance measure. HMM_{AE} was a HMM-recognizer based on abstract acoustic elements $\mathcal{A}_{WordConstrain}$ as described in Section 3.5.3. To build a word HMM from several utterances the approximative K-dimensional Viterbi as described in Section 3.2.4 was used.

First the results of the speaker-dependent, cross-speaker and speaker-independent scenarios are given in Section 4.1.1 and then discussed in Section 4.1.2.

4.1.1 Results of Recognizers with an Utterance-based Vocabulary

Speaker-Dependent Scenario

This section gives the results of the speaker-dependent tasks 1 and 4. In Table 4.1 the results of several recognizers are listed in dependence of the number of utterances which were used to form each word representation.

	language	German			French		
	# utterances	1	2	3	1	2	3
	DTW_{Eucl}	86.6	89.3	91.6	67.5	71.8	76.8
intra.-lang.	DTW_{VMLP}	93.1	92.8	94.6	79.1	80.2	84.1
	HMM_{AE}	93.5	96.2	96.8	86.6	93.4	94.8
cross-lang.	DTW_{VMLP}	92.9	92.4	93.7	76.9	78.5	82.5
	HMM_{AE}	91.5	94.4	96.2	76.3	85.4	88.3

Table 4.1: *Evaluation of different recognizers in a speaker-dependent scenario. Several utterances were used to construct the word representations. Recognition rates are given in %.*

Cross-Speaker Scenario

Table 4.2 shows the performance of the various recognizers for cross-speaker tasks 2 and 5. The results are given in Table 4.2.

	language	German			French		
	# utterances	1	2	3	1	2	3
	DTW_{Eucl}	62.1	64.6	66.4	49.7	52.8	56.4
intra.-	DTW_{VMLP}	80.6	80.9	82.8	67.5	69.3	72.6
lang.	HMM_{AE}	86.3	90.6	92.0	80.6	88.6	90.8
cross.-	DTW_{VMLP}	75.6	79.6	80.9	63.9	65.9	69.6
lang.	HMM_{AE}	80.1	84.7	86.8	66.3	74.8	78.1

Table 4.2: *Evaluation of different recognizers in a cross-speaker scenario if several utterances were used to construct the word representations. Recognition rates are given in %.*

Speaker-Independent Scenario

These experiments have shown how a recognizer performed if the word representations are speaker-independent as in tasks 3 and 6. In these tasks up to 20 utterances per word were available. The utterances were randomly chosen from a large number of utterances spoken by a big population of speakers. The language-dependent (black lines) and cross-language (grey lines) recognition rates in dependence of the number of available utterances per reference word are given in Figure 4.1.

4.1.2 Discussion of Recognizers with an Utterance-based Vocabulary

Comparing the results of the speaker-dependent and the cross-speaker scenario clearly showed that the inter-speaker variability had an adverse effect on the recognition rate. This effect was however much more severe for DTW_{Eucl} than for DTW_{VMLP} and HMM_{AE}. This shows that the developed techniques enhance the robustness against inter-speaker variability.

If the word representations could be build from three utterances of several speakers, as is the case of the speaker-independent scenario, the recognition rates were only slightly worse than in the speaker-dependent case. If 20 utterances could be used the results were even better than
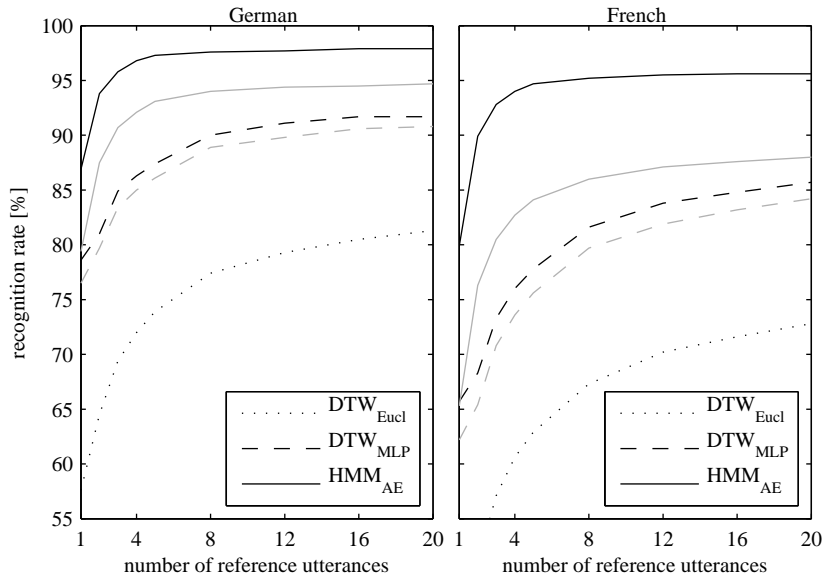
Figure 4.1: *Recognition rate as a function of the number of reference utterances for speaker-independent tasks 3 and 6. The black lines give the results for systems which use resources (VMLPs, abstract acoustic elements) trained on the target language. The grey lines give results for cross-language experiments, i.e. experiments in which the resources were taken from the other language.*

in the speaker-dependent scenario with three utterances – it is indeed likely that more utterances are available in the speaker-independent case since they do not need to be collected for every user separately.

Among the DTW-based recognizers DTW_{VMLP} performed much better than DTW_{Eucl}. The largest gain of a VMLP used in the DTW-based approach was observed in difficult tasks such as the French recognition task with one reference utterance per word. The recognizer based on abstract acoustic elements HMM_{AE} performed however again much better. The superiority of abstract acoustic element recognizers was particularly high if several observation sequences were used.

In the cross-language case the systems performed quite well in the speaker-independent scenario even though the effect of language-mismatch was clearly observable. For the HMM_{AE} systems with 20 utterances the recognition rate decreased from 97.9 % to 94.7 % for German and from 95.6 to 88.0 for French. This decrease may seem quite high, but if we consider that the recognition rates of DTW_{Eucl} were only 81.3 % for German and 72.8 % for French, these figures are still satisfactory. More concretely, the cross-language systems HMM_{AE} yielded a decrease of the error rate relative to DTW_{Eucl} of more than 70 % for German and almost 60 % for French. For the language-dependent HMM_{AE} systems these numbers were almost 90 % for German and more than 80 % for French.

We have seen that HMM_{AE} generally performed better than DTW_{VMLP}. In the cross-language case the difference between the techniques was however considerably smaller than in the language-dependent case. This suggests a higher robustness against language-mismatch of the DTW/VMLP-based technique.

4.2 Comparison with Dictionary-based Recognizers

It remains to evaluate how close the devised techniques come to speaker-independent isolated word recognizers (IWR) which are possible to construct for resource-rich languages. The two languages German and French which we used to test our methods both fulfill this requirement. We therefore tested all tasks with our standard phoneme-based speaker-independent recognizer as described in Appendix D. The results are listed in Table 4.3 along with the recognition rates of the best performing systems based on an utterance-based vocabulary. The results are given for three reference utterances in all scenarios and additionally for 20 reference utterances in the speaker-independent scenario.

First we compared the HMM system based on abstract acoustic elements concatenated according to utterances to the dictionary-based system in a speaker-dependent scenario in the intra-language case. On a first glance this comparison could be considered as unfair since the

language	num. of refs.	scenario	DTW_{Eucl}	HMM_{AE}		dict.-based
				intra.-lang.	cross.-lang.	
German	3	spkr.-dep.	91.6	96.8	96.2	96.9
		cross-spkr.	66.4	92.0	86.8	
		spkr.-indep.	69.4	95.8	90.7	97.4
	20		81.3	97.9	94.7	
French	3	spkr.-dep.	76.8	94.8	88.3	96.7
		cross-spkr.	56.4	90.8	78.1	
		spkr.-indep.	57.1	92.8	80.5	96.7
	20		72.8	95.6	88.0	

Table 4.3: *Comparison of recognition rates in % between different recognizers. The results are given for all scenarios. For HMM$_{AE}$ results for intra-language (abstract acoustic elements from the target language) and cross-language (abstract acoustic elements not from the target language) experiments are given. For the recognizers with an utterance-based vocabulary three utterances were used in the all scenarios and in the speaker-independent scenario additionally the results with 20 utterances are given.*

dictionary-based recognizer is speaker-independent. There may however be situations where one has to build an IWR for a single speaker and thus both, an utterance-based speaker- and language-dependent and a dictionary-based speaker-independent recognizer are viable options. If there is only a single utterance available for each vocabulary word, the best utterance-based recognizer yielded an error rate which was around twice as high than the one of a dictionary-based recognizer for German and around around three times as high for French (cf. Tables 4.1 and 4.3). These results clearly suggest to use the dictionary-based recognizer in this case. With as few as three utterances per vocabulary word the utterance-based recognizer had a similar performance as the dictionary-based recognizer for German while it was a bit worse for French. A reason for the better performance of the utterance-based system in German was probably the test-speaker population. Most speakers were of Swiss German mother tongue. This caused a dialectal influence of their pronunciations which therefore dif-

fered from the standard German pronunciations which were used in the dictionary-based recognizer.

Next we compared the results of speaker-independent tasks 3 and 6. With around 20 utterances from several speakers per reference word the best utterance-based method presented in this thesis reached a similar performance as a dictionary-based recognizer if models of the target language were available.

For a cross-language usage, the recognizer with an utterance-based vocabulary achieved a lower recognition rate than a recognizer which is built for the target-language. In the speaker-independent scenario with 20 reference utterances the HMM_{AE} recognizer had an around twice as high error rate than a language-dedicated dictionary-based recognizer for German (2.6 % vs. 5.3 % absolute error rate) and an around 3.5 fold higher error rate for French (3.3 % vs. 12.0 % absolute error rate).

The results of the French tasks are consistently worse than the results of the German tasks. One reason for this is the higher average length of the utterances in the German tasks than the one in the French tasks.

4.3   Conclusion

We could see that the HMM-based approaches to IWR with a utterance-based vocabulary performed better than the DTW-based approaches even though the use of a VMLP instead of the Euclidean distance yielded a substantial performance gain. The abstract acoustic elements which exploit orthographic annotations performed by far better than abstract acoustic elements which are trained in a completely unsupervised way even if applied in another language than the training language of the elements. The abstract acoustic elements performed also better than phonemes.

By using the HMM-based technique the error rate of recognition with an utterance-based vocabulary could be reduced by around 65 % compared to a standard template-based recognizer even if the abstract acoustic elements were not trained on the target language.

In the intra-language and speaker-dependent scenarios the best recognizer with utterance-based vocabulary presented in this thesis achieved almost the same performance as a dictionary-based recognizer if about three utterances of each word in the vocabulary were available to build the word models. In a speaker-independent scenario a word recognizer with an utterance-based vocabulary could compete with a dictionary-based recognizer only if the abstract acoustic elements were language-dependent. If abstract acoustic elements of another language were used it produced considerably more errors than a dictionary-based recognizer. The difference is however that in contrast to the training of phonemes which are necessary for a dictionary-based recognizer, only acoustic training data but no pronunciation dictionary is necessary for the training of abstract acoustic elements.

In terms of resource requirements the dictionary-based recognizer needs both a pronunciation dictionary and annotated training data from the target language. For the intra-language case the utterance-based technique achieves a similar performance to the dictionary-based recognizer but requires only orthographic annotations and no pronunciation dictionary. In the cross-language case which requires no resources of the target language the recognition rates are lower than the ones of a dictionary-based recognizer but still considerably higher than the ones of a standard utterance-based recognizer.

Chapter 5

Other Application Scenarios

In previous chapters we have investigated isolated word recognition for languages that do not allow to build a dictionary-based recognizer because no pronunciation dictionary is available and therefore only the use of a vocabulary which is based on sample utterances of the words is a viable alternative. We have presented two techniques which are considerably better than dynamic time warping which is usually used for this task.

There are other applications which can profit from these techniques in that these applications need to compare two utterances in some way. Some of these applications are

- *Utterance Verification:* In some applications it has to be verified whether two utterances are equally worded. With the techniques developed in this thesis this task can be performed without even knowing which language is spoken in the two utterances. The techniques promise in particular a higher speaker-independence compared to standard DTW-based techniques. Such a task arises for example in educational applications if it has to be verified whether a student has pronounced the same word or phrase as in

a reference recording of a teacher and that he has pronounced it
correctly.

- *Acoustic Data Mining:* Acoustic keyword spotting, also known
  as spoken term detection, is nowadays mostly based on word lat-
  tices ([Wei95]), phone lattices ([JY94]) or both ([MRS07], [Mil07])
  which are generated with a large vocabulary speech recognition
  (LVCSR) system. This makes these systems language-dependent.
  The techniques presented in this thesis facilitate the construction
  of systems which are language-independent.

- *Speaker Verification:* Speaker verification systems which are
  based on the comparison of words which occur both in the ref-
  erence and the test utterance usually use a LVCSR to find the
  common words. This makes the systems language-dependent.
  The developed techniques open new possibilities to find common
  words in two speech signals in a language-independent way.

In this thesis we investigated acoustic data mining with a task to
find common segments such as words or phrases in two signals. These
investigations will be presented in Section 5.1. An application of this
task in a speaker verification application is presented in Section 5.2.

5.1 Acoustic Data Mining

In this section we investigate a task of acoustic data mining, namely to
find common segments such as words or syllables in two speech signals
with techniques that are language-independent, i.e. techniques which
can be applied for speech signals of any language.

The amount of literature available for this task is scarce. The only
algorithm we are aware of is *Segmental DTW* presented in [PG05]. It
bears indeed some similarity to the algorithm which we present in Sec-
tion 5.1.1 since it is also based on a distance matrix. The algorithm
partitions the distance matrix in overlapping bands in diagonal direc-
tion and finds the optimal path for each band. From this optimal path
a subsegment with a low distance is sought with an algorithm presented
in [LJC02]. A disadvantage of the algorithm is that it finds only one

matching segment in a diagonal band. This has a big impact if the two signals are almost identically worded but have some inserted, substituted or deleted words. In this case the common segments are expected in a few bands around the diagonal of the distance matrix.

How a derivate of the DTW algorithm in combination with verification multilayer perceptrons (VMLP) as presented in Section 2.2 can be used for this task is shown in Section 5.1.1. In Section 5.1.2 we show how HMMs based on abstract acoustic elements can be used for the task. The two approaches are compared in Section 5.1.3.

5.1.1   Seek Similar Segments with Modified DTW

Distance / Probability Matrix

In speech applications the DTW algorithm is usually used to find the best alignment between two observation sequences which are hypothesized to contain the same wording. In the usual case of a distance matrix it is expected that there is a valley of small local distances approximately along the diagonal through the whole distance matrix if the hypothesis is true (i.e. both observation sequences contain the same word). If a VMLP as described in Section 2.2 which outputs the probability that the two input frames are from the same phoneme for each frame pair is used, a ridge in diagonal direction is expected.

In the task we are aiming to solve we do not expect that the two signals are equally worded. Therefore there will be no ridge across the whole probability matrix. If there are however segments of the two signals which match, there will be ridges in approximately diagonal direction at the corresponding locations. An example of a probability matrix is shown in Figure 5.1.

Search Algorithm

Given a probability matrix of the type described above we need an algorithm to detect the ridges. We have therefore devised an algorithm which is related to DTW but is not restricted to find one optimal path through the full probability matrix.
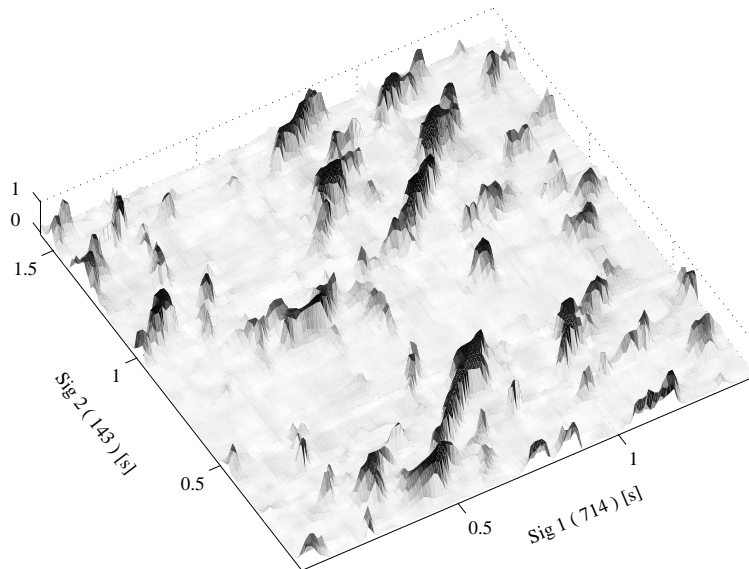
Figure 5.1: *Probability matrix spanned by the German numbers 714 ("siebenhundertvierzehn") and 143 ("hundertdreiundvierzig") uttered by two different speakers. Common segments show as ridges, e.g., the word "hundert" which occurs in both numbers shows as a ridge between 0.6 and 1 s in signal 1 and between 0.1 and 0.4 s in signal 2.*

In contrast to the standard application of DTW where processing starts and ends at more or less constrained starting and end points and is unidirectional, we use bidirectional DTW that starts at the peak of a potential ridge and proceeds in both directions until the ridge ends. More precisely the detection is as follows:

1. The processing starts at the point where P_{ij} is maximal.

2. From this initial point DTW is used to follow the potential ridge, i.e. to construct a warping curve in both directions. As usual, slope constraints are used that allow to compensate a maximum local speaking rate difference of the two speech signals of a factor of two.

3. The ridge ends where P_{ij} is smaller than threshold P_b for two consecutive DTW steps.

4. The points of P_{ij} constituting the found ridge are excluded from further processing.

5. If the found ridge is longer than the minimum length L_m, the partial warping curve with the frame pairs is kept as a part of the result of the algorithm.

6. If the maximum of all not yet excluded P_{ij} is greater than threshold P_a, processing loops back to step 1.

The parameters of the ridge detection, i.e. the thresholds P_a for starting a ridge and P_b for ending a ridge and the minimum length L_m have to be optimized by means of an appropriate data set. If P_a and P_b are chosen too high the probability of missing a common segment increases whereas low P_a and P_b may lead to spurious common segments. Also the choice of L_m is a tradeoff between missing short segments for a long L_m and finding unreliable matches for a short L_m.

5.1.2 HMMs to Seek Similar Segments

By means of abstract acoustic elements the common segments in the observation sequences $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ from the two speech signals S_1 and S_2 are detected as follows: In the first step the optimal sequence

$\hat{Z}^{(1)}$ of acoustic elements for $\mathbf{X}^{(1)}$ is determined. In the second step we seek segments of the observation sequence $\mathbf{X}^{(2)}$ which are sufficiently well described by model subsequences of the sequence $\hat{Z}^{(1)}$.

Decoding the First Signal

To determine the optimal sequence $\hat{Z}^{(1)}$ of $\mathbf{X}^{(1)}$ we use an HMM as shown in Figure 3.2 in the same way which is used to determine the word-HMMs for isolated word recognition as described in Section 3.1. This results in a sequence of abstract acoustic elements as e.g.

$$
\begin{aligned}
\hat{Z}^{(1)} &= \varphi_{66}\varphi_{15}\varphi_{43}\ldots\varphi_{97} & (5.1) \\
&= \hat{z}_1^{(1)}\hat{z}_2^{(1)}\hat{z}_3^{(1)}\ldots\hat{z}_U^{(1)}, \quad \hat{z}_u \in \Phi & (5.2)
\end{aligned}
$$

For all elements $\hat{z}_u$ in $\hat{Z}^{(1)}$ we save also the time $t_b(\hat{z}_u)$ when the element was entered and $t_e(\hat{z}_u)$ when the element was left.

Find Similar Segments in the Second Signal

Now we extend the HMM of Figure 3.2 with the model sequence $\hat{Z}^{(1)}$ which results in an HMM as shown in Figure 5.2. This HMM consists of two parts. The part on the right side is identical to the one used to decode the first signal. The part on the left side represents the sequence $\hat{Z}^{(1)}$ of acoustic elements found for $\mathbf{X}^{(1)}$. These elements constitute first of all a linear HMM. There are additional connections through the non-emitting states that have got again penalties. The motivation for theses penalties H_a and H_b is similar as in keyword-spotting, where penalties are used to tune the operation point of the system to minimize false alarms and missed detections (see e.g. [RP90]). Here the penalties control the number and the lengths of the detected segments. The penalties are implemented by subtracting the value H_a or H_b from the accumulated path log likelihood if a transition with a penalty is taken. In the absence of such penalties the Viterbi decoder would almost never use a transition $\hat{z}_u \rightarrow \hat{z}_{i+1}$ because $\mathbf{X}^{(2)}$ can't be better modeled than with a free sequence of acoustic element models as represented by the right-hand side of the HMM of Figure 5.2. In this HMM the penalties H_b are higher (i.e. a larger log-likelihood value is subtracted in the
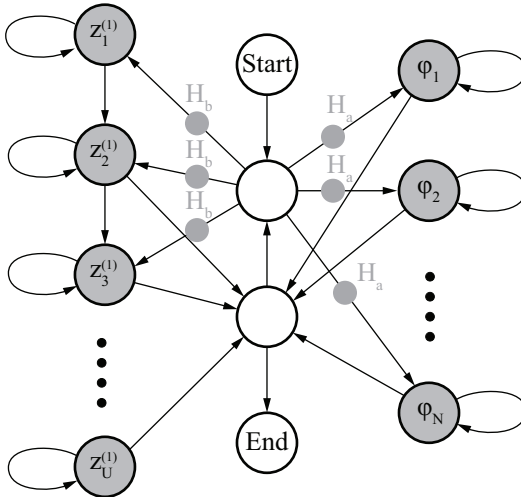
Figure 5.2: *Special Markov model to find segments of the observation sequence $\mathbf{X}^{(2)}$ that match parts of the sequence $\hat{Z}^{(1)}$. All emitting states (printed in grey) are modeled by acoustic elements. Either directly through $\varphi_n \in \Phi$ or indirectly through the $\hat{z}_u^{(1)}$. Non-emitting states are printed in white. H_a and H_b are additional penalties associated with some transitions.*

corresponding transitions) than the penalties H_a in order to prevent the detection of very short segments.

The Viterbi decoder delivers a sequence of abstract acoustic elements $\hat{Z}^{(2)}$ through the HMM of Figure 5.2 for the observations sequence $\mathbf{X}^{(2)}$. This sequence $\hat{Z}^{(2)}$ could look like

$$\hat{Z}^{(2)} = \varphi_7 \ldots \varphi_{95} \hat{z}_{18}^{(1)} \hat{z}_{19}^{(1)} \hat{z}_{20}^{(1)} \hat{z}_{21}^{(1)} \varphi_{17} \ldots \varphi_{14} \qquad (5.3)$$

In sequence $\hat{Z}^{(2)}$ the subsequences consisting of elements $\hat{z}_u^{(1)}$ represent the desired common segments. In the example above the subsequence of interest is $\hat{z}_{18}^{(1)} \hat{z}_{19}^{(1)} \hat{z}_{20}^{(1)} \hat{z}_{21}^{(1)}$. With the entry times $t_b(\hat{z}_{18}^{(1)})$ and the leaving times $t_e(\hat{z}_{21}^{(1)})$ of $\hat{Z}^{(1)}$ and $\hat{Z}^{(2)}$ we can determine the positions of the segment in the two signals.

A handicap of this method is that it can detect for a segment of $\mathbf{X}^{(2)}$ only one matching segment in $\mathbf{X}^{(1)}$ (the other way round is no problem). To alleviate this problem we split the state sequence $\hat{Z}^{(1)}$ into several overlapping windows and perform the algorithm described above for every window of $\hat{Z}^{(1)}$.

5.1.3 Experimental Comparison

Objectives

In the problem of finding common segments in two sequences there are two objectives which should be optimized.

- The found segments should be correct

- All potentially available segments should be detected

The devised methods are likely to optimize one objective at the cost of the other. Which objective will be predominantly fulfilled depends on the parameters P_a, P_b and L_m for the DTW-based algorithm presented in Section 5.1.1 and on H_a and H_b for the HMM-based algorithm presented in Section 5.1.2.

In order to evaluate the quality of the segment search algorithm it was therefore suitable to test the systems with various parameter-sets and determine the Pareto-front of the algorithms for the two objectives.

In the next sections we will show how the two objectives were measured.

Matching Quality

We generated phonetic segmentations of all utterances in the test set with a forced-alignment using only one pronunciation variant per word. In this way we could check for all frame pairs from the found common segments, whether the two frames were assigned the same phoneme label. If the phoneme labels were matching, we assigned the value 1. If only the phoneme label in a directly neighboring frame of the other

signal matched we assigned the value 0.3. All other frame pairs were assigned the value 0. We then calculated the total matching score as the mean of the values assigned to all frame pairs.

Completeness of the Found Sequences

The other objective is the ratio between the number of detected common segments and the number of common segments which are effectively available in the two speech signals. Therefore we determined potential matches by seeking common phoneme sequences in the phonetic segmentations of the signals. We required the phoneme sequences to be at least four phonemes long. With the phonetic segmentations also the locations of every common phoneme sequence in the two speech signals were known. A potential match determined in this way was considered as detected if the tested algorithm detected a common segment at roughly the same positions in the two signals.

Test Setup

We used recordings of German three-digit numbers. Every recording contained four three-digit numbers. The recording pairs in which the common segments were sought never contained the same three-digit numbers. Thus only parts thereof could match. The recordings were taken form speakers of the $\mathcal{S}_{G,digit,3}$ set as described in Appendix E.1. This yielded a total of 1299 tested record pairs for speaker-dependent tests and a total of 1189 record pairs for cross-speaker tests.

We used the features $Feat_{cms}$ as described in Appendix C.1. The VMLPs and the abstract acoustic elements were trained with data of the $\mathcal{S}_{G,poly,1}$ and $\mathcal{S}_{F,poly,1}$ speaker sets as described in Appendix E.1. We used abstract acoustic elements $\mathcal{A}_{WordConstrain}$ trained as described in Section 3.5.3 and VMLPs with one hidden layer containing 209 neurons.

Results

First we evaluated the intra-language case, i.e. if the VMLPs and the abstract acoustic elements were trained in the target language. Due to the lack of suitable testing data for French we tested only with German words.

Intra-Language Case

The Pareto fronts for the DTW-based system as described in Section 5.1.1 and the HMM-based system as described in Section 5.1.2 are shown in Figure 5.3. The parameter L_m of the DTW-based system was fixed at 120 ms and also for the HMM based systems only segments longer than L_m were accepted. For the DTW-based system the parameter P_a was varied between 0.9 and 0.98, and P_b between 0.8 and 0.97. In the HMM-based systems the parameter H_a was varied between 4 and 13 and H_b between 12 and 40.

From the resulting plots in Figure 5.3 we concluded that seeking common segments was a much more difficult task if the two speech signals were not from the same speaker. We always performed the tests for pairs of different speakers with the same parameters which we also used for testing the signal pairs from different speakers. We could see that mostly the completeness of the found segments was impaired if the signals are from different speakers and not so much the quality.

For both, signal pairs from the same speaker and signal pairs from different speakers the DTW-based systems performed better but the difference was a bit bigger for signal pairs from the same speaker.

Cross-Language Case

In order to investigate whether the suggested algorithms to seek common segments can also be used if there is no training material available for the target language or if the target language is not even known we tested the algorithms in a cross-language scenario. The VMLPs and the abstract acoustic elements were trained with French data and the tests were performed on a German test set.
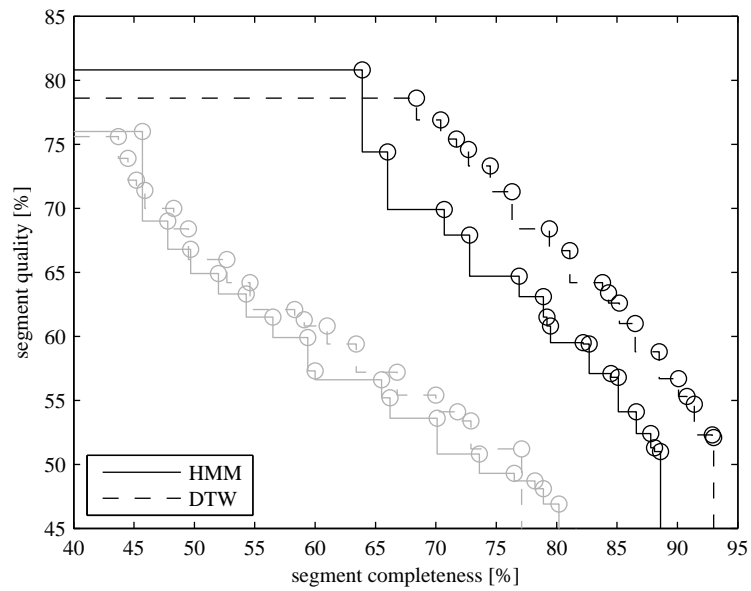
Figure 5.3: *Pareto fronts for the two objectives completeness and matching quality in intra-language experiments. The tests in which the two recordings in a pair are from the same speaker are plotted in black, the tests in which the recordings were from different speakers are plotted in grey.*
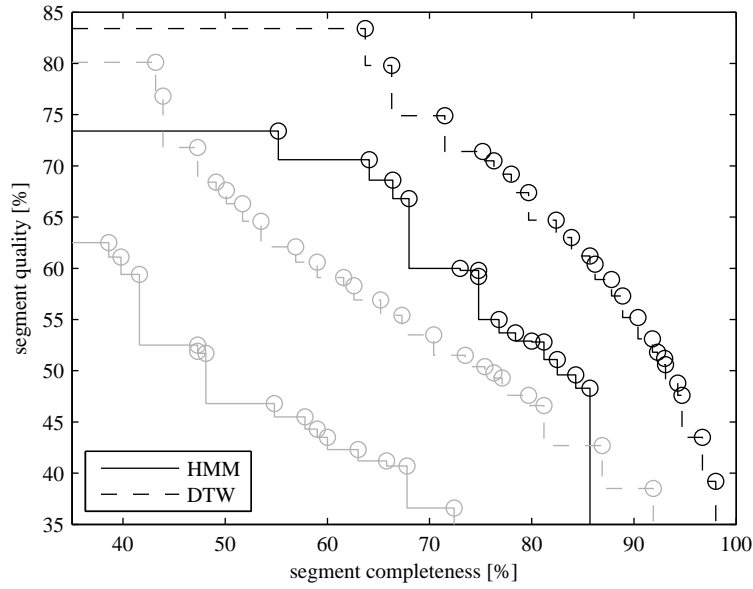
Figure 5.4: *Pareto fronts for the two objectives completeness and matching quality in cross-language experiments. The tests in which the two recordings in a pair are from the same speaker are plotted in black, the tests in which the recordings were from different speakers are plotted in grey.*

The result is again illustrated with the Pareto fronts shown in Figure 5.4. The fronts for the DTW-based systems were quite similar as in the intra-language experiments, indicating that the VMLPs were not very language-dependent, at least not within the two languages German and French. For the HMM-based systems the performance dropped much more.

5.2 Speaker Verification

In this section we investigate a novel approach to text-independent speaker verification as an application scenario for the methods to seek common segments which were presented in the Section 5.1. The target is to make speaker-verification based on pattern matching (PM), which yields very good results for text-dependent speaker verification, applicable for text-independent scenarios. This is achieved by detecting phonetically similar segments in two speech signals with the methods devised in this thesis. The PM approach is then applied for the found common segments. Since the algorithm requires common segment to be available in two speech signals it is more correctly termed quasi text-independent speaker verification.

The devised speaker verification method is related to other methods as outline in Section 5.2.1 but distinguishes itself by the fact that it is not dependent on a LVCSR system and is therefore language-independent.

5.2.1 Related Work

Speaker verification is the task of verifying whether a test signal S_{test} is from the same speaker as a reference signal S_{ref}. The first step in most approaches is that observation sequences $\mathbf{X}^{(ref)}$ and $\mathbf{X}^{(test)}$ are extracted from both signals by short-time analysis. The approaches differ in the way they compute a score which reflects the probability that the two signals are from the same speaker.

We can distinguish between text-dependent speaker verification and text-independent speaker verification. In text-dependent speaker-

verification the signals S_{ref} and S_{test} contain the same text. This allows verification techniques which are not possible for text-independent speaker-verification, where the content of the signals is unknown.

Text-dependent Speaker Verification

In our lab text-dependent speaker verification is used for forensic purposes because of the high confidence which can be achieved (see for example [PB03]). The system is based on the distances between the frame pairs along the warping curve between $\mathbf{X}^{(ref)}$ and $\mathbf{X}^{(test)}$ which is determined by means of DTW. In [NP04] it was shown that the error probability could be significantly reduced by using an appropriately trained verification multilayer perceptron (VMLP) (cf. Section 2.2).

Text-independent Speaker Verification

Text-independent speaker verification systems can be roughly grouped into system which consider the observations of both observation sequences $\mathbf{X}^{(ref)}$ and $\mathbf{X}^{(test)}$ individually and neglect their temporal order and systems which consider the observations in their natural order.

The first group of approaches, also termed bag-of-frames methods, is used more often because of its good performance at a moderate computational cost. The standard approach is the GMM-UBM approach which is well described in [RQD00]. The universal background model (UBM) is a GMM which models the feature distribution of a large population of background speakers. From the observation sequence of the reference speaker $\mathbf{X}^{(ref)}$ a speaker-dependent GMM is adapted with maximum a posteriori adaptation. The score is then computed as the ratio between the likelihood that the observations of $\mathbf{X}^{(test)}$ were produced by the speaker dependent model and the likelihood that the observations were produced by the UBM.

More recently support vector machines have been used for the bag-of-frames approach. Some approaches are based on specially developed sequence kernels ([Cam02], [MB07]). Other approaches such as the one presented in [CSR06] have their origin in the GMM-UBM approach. They train a support vector machine for each speaker by using the

supervector (concatenation of the mean vectors of all mixture components) as input vector.

A big source of errors in speaker verification is channel mismatch. By using factor analysis it was possible to distinguish between differences caused by a different speaker and differences caused by a channel mismatch ([KD04]). This benefit of factor analysis could be used to achieve better results for speaker verification in [MSFB07] and [CSRS06] by integrating factor analysis into support vector machine supervector based systems.

The method presented in this thesis belongs the second group of approaches which considers the frames of $\mathbf{X}^{(ref)}$ and $\mathbf{X}^{(test)}$ within their context. Most of these approaches are based on the output of a LVCSR system which makes them language-dependent.

Some of these systems calculate a verification score only based on higher-level sequential information such as speech prosody in [AH03] or phoneme n-gram statistics in [EP07].

The systems presented in [PCC00] or [PJ02] use individual GMMs for phonetic classes or more acoustically motivated classes. The systems described in [CE98], [HH03] or [GSP05] go one step further regarding the use of sequential information by aligning the frames belonging to the same phoneme with DTW and using the average distance as system output. In [New96] a system was presented which uses the ratio of the likelihood of a speaker-adapted LVCSR system to the likelihood of a speaker-independent LVCSR system as verification score. Usually this family of systems showed a performance gain especially if they were used in combination with a global GMM system (see for example [PJ02] or [WPN+00]).

Other systems which are based on LVCSR systems calculate a score on a predefined set of keywords. In [Stu02] individual GMMs are trained for each keyword found sufficiently often in S_{ref}. The approach chosen in [BP04] is similar but used word-HMMs to model each keyword. As shown in [MH05] these approaches are successful for long signals and if used in combination with a GMM system. In [ABA04] the keywords are detected with a DTW-based system and also the final verification score is computed from the DTW-scores.

The system described in [PG06] resembles to our approach since it is based on the comparison of common segments, which are detected with a DTW-based algorithm and not with a LVCSR system. The differences are however that only the best matching segment is used and that it uses the same Euclidean distance metric both for detecting the common segments and for determining the speaker verification score.

5.2.2   System Description

The speaker verification approach presented in this thesis involves three steps. First phonetically matched segments (e.g. common words or common syllables) are sought in the two speech signals. This step provides a series of frame pairs where both frames in a pair are phonetically matched. In a second step the probability that the two frames in a pair come from the same speaker is computed for every frame pair with a VMLP as presented in Section 5.2.4. Finally, a global indicator that the two speech signals were spoken by the same speaker can be calculated from these frame-pair-level probabilities. This is described in Section 5.2.5.

5.2.3   Seeking Equally Worded Segments

We need an algorithm which seeks common segments in two speech signals which have a minimum length. In Section 5.1 we devised two such algorithms, one based on HMMs, the other based on DTW.

There we have formulated the task as a problem with two objectives – one being the completeness of the found segments and the other the correctness of the found segments. In this task there is only one objective: the performance of the speaker verification algorithm. The speaker verification is impaired by a bad matching quality and by a high rate of missed detections of available common segments. If the matching quality is bad, the MLP will be fed with vector pairs of non-matching frames, which will deteriorate its performance. If on the other hand not all common segments are detected, the final decision does not consider all available information.

A further requirement is that the system is speaker-independent, or

more precisely formulated that the systems works equally well on two signals which are from different speakers as it works on two signals of the same speaker. In Section 5.1 we have seen that this is not totally fulfilled. The completeness is considerably smaller if the signals are from different speakers. This means that information which may be beneficial for speaker verification is lost. This also implies that the optimal parameters can only be determined by optimizing the speaker-discriminating ability of the system.

5.2.4   VMLP-based Probability Estimation

We use a VMLP to calculate for each frame pair on the warping curve the posterior probability that the two frames are from the same speaker. This speaker verification MLP is trained to output the posterior probability that the two observations $\mathbf{x}_1$ and $\mathbf{x}_2$ are from the same speaker given they are from the same phoneme:

$$P_{local} = P(spkr(\mathbf{x}_1) = spkr(\mathbf{x}_2)|\mathbf{x}_1, \mathbf{x}_2, phon(\mathbf{x}_1) = phon(\mathbf{x}_2)) \quad (5.4)$$

The VMLP has shown good results for text-dependent speaker-verification in [NP04] and has been further developed in this thesis.

Training Data for Speaker Verification MLPs

The preparation of training data for speaker verification is similar to the approach described in Section 2.3.3, but both the positive and the negative observation pairs are taken from observation pairs on the warping curve – the positive observation pairs from warping curves of signals from the same speaker and the negative observation pairs from warping curves of signals from different speakers.

5.2.5   Final Decision

Finally we evaluate from the scores of all frame pairs the global score that can be used to decide whether the two speech signals were spoken by the same speaker or not. The global score is the average over the frame-level scores of all found common segments.

5.2.6 Experiments

Compared GMM System

As a baseline system we used a GMM system as described in [RQD00]. We found 1024 Gaussians to be optimal. For each speaker verification trial a speaker model was created by adapting the UBM with the first of the two given signals using a maximum a posteriori adaption (see [GL94]). We have seen, that the system performed best if only the means of the Gaussians were adapted. This is in accordance with the results given in [RQD00]. The log-likelihood (LL) of the second signal was calculated for the adapted speaker model and for the UBM. The difference of the two LL values is the desired output of the GMM system. For the training of the UBM, the maximum a posteriori adaption and the calculation of the LL we have discarded silence frames.

System Combination

In order to verify whether the devised PM system and the GMM system make use of complementary information, we also evaluated a system combination. We used a weighted average of the scores delivered by the two systems. We empirically evaluated equal weights to be optimal. The scores of both systems were normalized to zero mean and unit variance before the fusion.

Experiment Setup

For the speaker verification part, i.e. as input for the GMM system and as input to the speaker discriminating VMLP we used features $Feat_{spveri}$ as described in Appendix C.1. Only for the algorithm to seek common segments we used $Feat_{cms}$.

The VMLPs used for speaker verification (both for static and delta features) had one hidden layer with 209 neurons. In experiments not shown in this thesis we have found this a suitable size.

For the DTW-based algorithm (cf. Section 5.1.1) the parameters P_a and P_b were 0.95 and 0.9 respectively. For the HMM-based algorithm

(cf. Section 5.1.2) the parameters H_a and H_b were 9 and 27 respectively. In both algorithms segments shorter than 120 ms were discarded.

For the experiments we used recordings with German three-digit numbers, here called words. The average duration of a spoken three-digit number is 1.5 s.

All speaker verification trials were conducted with signals containing one to seven words taken from speaker set $\mathcal{S}_{G,digit,3}$ as described in Appendix E.1. None of the words of the first signal occurred in the second one. Therefore only common digit names could be detected, not whole numbers.

The speaker sets $\mathcal{S}_{G,digit,1}$ and $\mathcal{S}_{G,digit,2}$ were used to train the speaker verification MLPs and the UBM of the baseline system. Since these speaker sets are disjoint from $\mathcal{S}_{G,digit,3}$, the VMLPs and the UBM were used in a speaker-independent way.

The abstract acoustic elements and the phoneme verification MLPs which were used in the algorithms to find common segments were trained with data of the polyphone databases and were therefore also speaker-independent. The training data for the intra-language experiment was taken from the Swiss German polyphone database whereas the training data for the cross-language experiments was taken from the Swiss French polyphone database.

Results

We tested the two PM systems PM_{DTW} which used the DTW-based segment search algorithm and PM_{HMM} which used the HMM-based search algorithm along with the GMM system.

Segment Search in the Intra-Language Case

First we tested the speaker verification systems which use models of the target language (i.e. German). The results are given in Figure 5.5.

It could be observed that for only one word the GMM system was clearly superior to the PM based systems. The performance of the PM systems increased however with the number of words and was finally
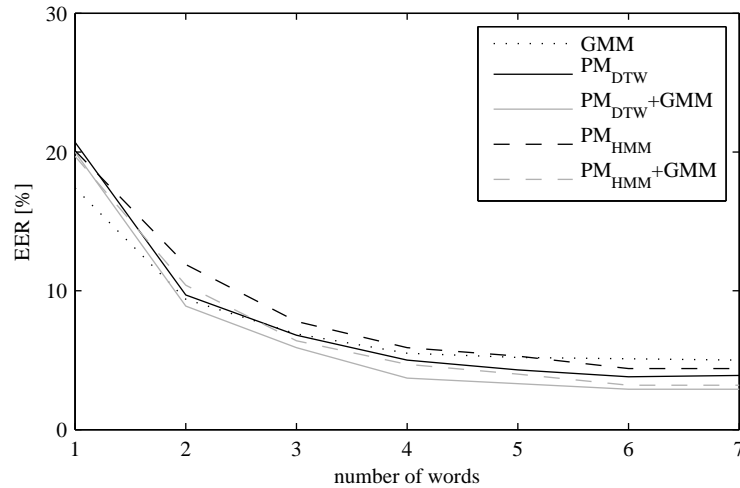
Figure 5.5: *EER of various systems as a function of the number of used words in intra-language experiments.*

better than the GMM system. With seven words the significance level of the superiority of both PM system over the GMM system was $< 0.1\,\%$ according to the McNemar test (see for example [GC89]).

Further considerable improvements could be achieved if the systems were fused. This allows the conclusion that the PM system uses complementary information to the GMM system to compute the probability that two signals are from the same speaker.

Segment Search in the Cross-Language Case

In Section 5.1.3 we have seen that the segment detection rate was impaired if abstract acoustic elements or VMLPs not of the target language were used. It remained to show what impact this language-dependence has on the speaker verification based on these segment-search algorithms. The results are shown in Figure 5.6.

The results of the PM_{HMM} system alone were clearly inferior to the
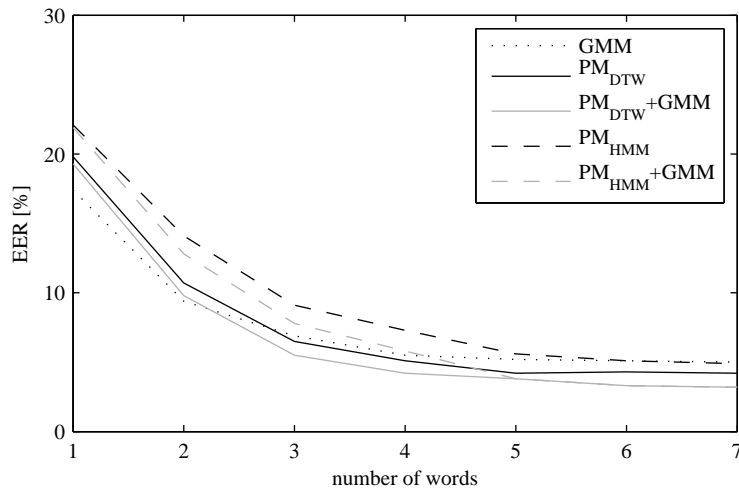
Figure 5.6: *EER of various systems as a function of the number of used words in cross-language experiments.*

ones achieved in the intra-language experiments. With seven available words a system combination yielded however still an improvement over the GMM system alone (significance level < 0.1 %). The PM_{DTW} system was more immune against language-dependence effects which is in line with the results obtained in Section 5.1.3. The PM_{DTW} system alone was still significantly better than the GMM system (significance level < 0.1 %).

Note on the Parameter Selection

The parameters were all constant. It is likely that better results could be achieved if the parameters were dependent on the length of the available utterance. The PM systems could for example be given a higher weight if the available utterances are long. We optimized the parameters for recordings of four words.

5.3   Concluding Remarks

In this chapter we have presented how the techniques developed in this thesis can be used for other applications than isolated word recognition. For both, DTW- and HMM-based approaches we presented algorithms to use them to find similar segments in two speech signals. The common segments of both approaches could be used in a pattern-matching based approach to quasi text-independent speaker verification.

For this application the DTW-based approach was superior to the HMM-based approach whereas the opposite was true for isolated word recognition. Since the basic techniques such as abstract acoustic elements and VMLPs were identical, the difference is likely to be in the algorithms to detect the segments. We had indeed to use some trickery to make segment search work satisfactorily with the HMM-approach. An example is the necessity to divide the reference signal into several overlapping windows.

Chapter 6

Conclusion

In this thesis we have developed techniques which allow to construct reliable isolated word recognizers (IWR) which have an utterance-based vocabulary. The advantage of this recognizer approach is that it has smaller resource requirements than dictionary-based recognizers and can therefore be used in any language or dialect even if no pronunciation dictionary is available. For the tested languages German and French a big improvement of the recognition rate relative to a standard approach to utterance-based recognition could be achieved with the developed techniques even if no resources of the target language were used (i.e. if resources of the other language were used). An even better performance was however possible if annotated acoustic training data from the target language was used.

Furthermore, the developed techniques could also be successfully used for other applications such as acoustic data mining in languages which do not allow to build a large vocabulary speech recognizer because of limited resources such as dictionaries and annotated speech databases.

6.1 Advances in Isolated Word Recognition with an Utterance-based Vocabulary

With the developed techniques the error-rate of recognizers with an utterance-based vocabulary could be more than halved in comparison with a standard DTW-approach even if exclusively resources of another language were used to train the necessary models and multilayer perceptrons.

If the VMLPs presented in this thesis were used in a DTW recognizer instead of the Euclidean distance, the IWR error rate could be greatly reduced. The DTW/VMLP recognizer was however outperformed by the developed HMM-based approach to recognition with an utterance-based vocabulary.

If abstract acoustic elements could be trained with data of the target language, only three utterances of each word sufficed that the utterance-based approach reached a similar performance as a dictionary-based recognizer in the speaker-dependent case. Around 20 utterances per word were necessary in the speaker-independent case. This result also showed the power of the presented extension of the Viterbi algorithm, which allows to find the sequence of sub-word units which optimally describe several utterances and could thus be used to build suitable word models. The relevance of these good results for languages with limited resources lies in the smaller resource requirements for the abstract acoustic element-based recognizer. Only acoustic data with orthographic annotations but no pronunciation dictionary is necessary to train abstract acoustic elements. In this intra-language case a different reduction of necessary training data could be achieved than with the approaches summarized in Section 1.3.2. With those approaches the amount of necessary annotated training data could be reduced while a pronunciation dictionary was still necessary. With the approach presented in this thesis annotated training data is still necessary for the intra-language case but there is no need for a pronunciation dictionary.

If the abstract acoustic elements were used in a cross-language scenario the approaches with an utterance-based vocabulary did not reach

the performance of dictionary-based recognition. The application of the abstract acoustic elements from another language is however an attractive choice if no resources for a language or dialect are available.

6.2   Benefits for Other Applications

The developed techniques also allow to improve techniques such as utterance verification or acoustic data mining for languages with scarce resources.

In this thesis we have presented two novel approaches in a rarely-treated field of acoustic data mining: to find similar segments such as word or syllables in two speech signals. Except for the method presented in [PG05] these are to our knowledge the only approaches which tackle the problem of finding similar segments in a language-independent way.

The approaches to find common segments could be successfully employed in a quasi text-independent speaker verification system.

6.3   Comparison of DTW- and HMM-based Approaches

We could see in IWR and in data mining tasks that the HMM-based approach was much more sensitive to data mismatch than the DTW-based approach. This was especially evident in the performance-decrease if German models were used instead of French models for a French task. This indicated that the acoustic space of French was not sufficiently covered by German acoustic elements. Since using French abstract acoustic elements for German was almost as good as using German models we argue that the abstract acoustic elements are suitable suitable sub-word units to be concatenated to word models according to sample utterances of the words.

6.4   Outlook

This thesis is rather a broad investigation for various techniques which aim at improving speech recognizers with an utterance-based vocabulary than an in-depth evaluation of a specific technique. This work may motivate further work intended to improve a particular approach.

In this work we have not performed an in-depth search for the best performing features but have relied on standard features. Research in this area might enhance the performance of both DTW- and HMM-based approaches. For the HMM-based techniques we see further possibilities for improvement by using model-adaptation techniques to adapt basic models for a specific language or a specific speaker population. This may also help to cope with the observed performance-loss due to data-mismatch which was observed for the HMM-based approaches.

In this thesis we have used German and French as experimental languages which we think to be a reasonable choice because the two languages are from two language families (Germanic and Romance) and have an only partially overlapping phoneme inventory. It is however interesting to investigate how the devised approaches work for a broader set of languages including non-Indo-European languages.

If a broader set of languages is available it is also possible to pool training data from several languages in order to build a set of abstract acoustic elements with a better coverage of the acoustic space and therefore also with a higher degree of language-independence. It may also be beneficial to have different sets of abstract acoustic elements for different language families. If the language families are not chosen too small it is likely that they contain some languages with abundant resources. The models produced from these languages may then also be suitable for resource-poor languages of the same family.

Appendix A

Performance of Verification Multilayer Perceptrons

In order to show that a VMLP has the capability to perform the verification task both for the classes which were seen during the training and for unseen classes we used two test steps. In a first step we showed the optimality of the VMLP for a closed set of classes with a KNN approach. In a second step we showed that the VMLP also generalizes for unseen classes with a real-world verification problem.

A.1 Reformulation as a Classification Problem

In order to evaluate a VMLP, we measure its verification error rate for a given dataset and compare it to a reference error rate which is optimal in a certain sense. By formulating our verification task as a classification problem, we can use the Bayes error as a reference. The Bayes error is known to be optimal for classification problems given the

distribution of the data.

To reformulate a verification task as a classification problem, each pair of vectors is assigned to one of the following two groups:

G_S group of all vector pairs in which the two vectors are from the same class

G_D group of all vector pairs in which both vectors are from different classes

Provided that the same classes which are present in the tests are used to estimate the distributions of G_S and G_D, the Bayes error is optimal, since the two distributions are modeled properly. Otherwise there is a mismatch which leads to ill-modeled G_S and G_D and thus the Bayes classifier is not necessarily optimal any more.

In the case of synthetic data it is possible to calculate the Bayes verification error since the data distributions are given in a parametric form. For real-world problems the data distributions are not given in a parametric form and hence the Bayes verification error can't be computed directly. In this case we can use a k nearest neighbor (KNN) classifier to asymptotically approach the Bayes error as described below.

The KNN approach is a straightforward means of classification. The training set for the KNN algorithm consists of training vectors with known classification $(a_{tr,i}, b_{tr,i})$ where $a_{tr,i}$ is the training vector and $b_{tr,i}$ is its associated class. A test vector $a_{tst,j}$ is classified by seeking the k nearest training vectors $a_{tr,i}$ and it is assigned to the class which is most often present among the k nearest neighbors. The KNN classifier is known to reach the Bayes error if an infinite number of training vectors is available (see e.g. [DHS01]) and is therefore a means to approximate the Bayes error if the data distributions are not known in a parametric form.
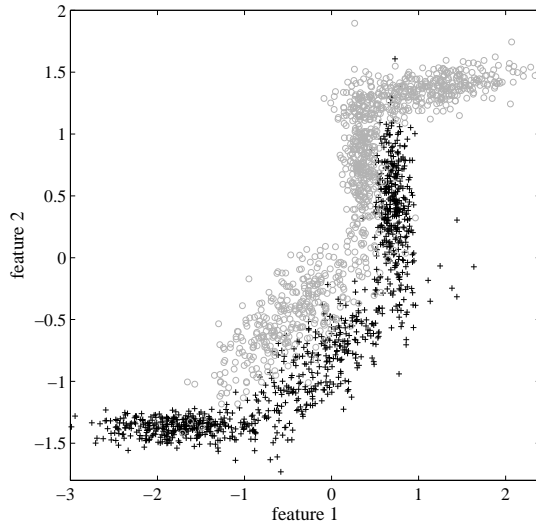
Figure A.1: *Synthetic data: two classes with two-dimensional non-Gaussian distributions.*

A.2   Synthetic Data

The aim of the experiments with synthetic datasets, i.e. datasets with known data distributions, was to evaluate if the VMLP achieves the lowest possible verification error from a Bayesian point of view. The data sets had two to four classes and were two- to five-dimensional. We illustrate these investigations by means of an experiment with a two-dimensional dataset with two classes that were distributed as shown in Figure A.1.

The number of training epochs which were necessary to train the VMLP depended largely on the type of the dataset. We observed the following properties:

- If only a few features carried discriminating information and all other features were just random values the VMLP learned quickly
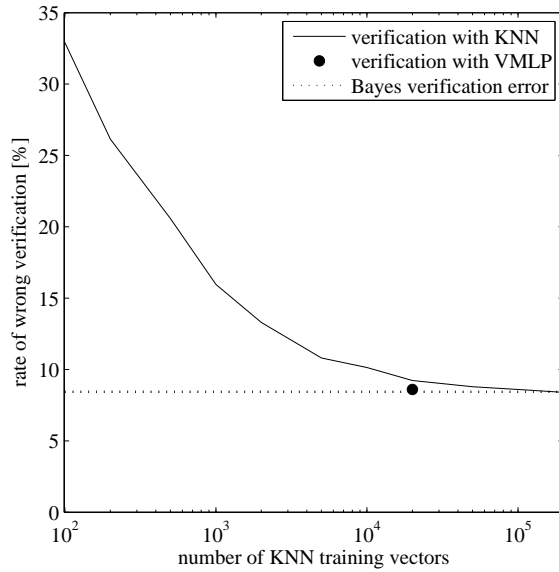
Figure A.2: *Class verification error for the test set as shown in Figure A.1: the KNN verification error is shown in function of the training set size. As expected, with increasing size it approximates the Bayes limit which is indicated with the dotted line. The error rate of the VMLP is close to the Bayes error.*

which features were useful and which ones could be neglected.

- The shape of the distributions strongly influenced the number of epochs that were necessary for the training. For example, two classes distributed in two parallel stripes or classes that had a nonlinear Bayes decision boundary, such as those shown in Figure A.1, required many epochs.

Figure A.2 shows the error rates of different verification methods for data distributed as shown in Figure A.1. It can be seen that the error of the VMLP was almost as low as the Bayes error. The used VMLP had two hidden layers with 20 hidden neurons in the first hidden layer and 10 in the second.

For the KNN-algorithm the error rate was evaluated as a function of the number of training vectors to see the asymptotic behavior which allows to estimate the Bayes error. For the VMLP we were only interested in the best possible verification error for a given task and not in the verification error in function of the number of training vectors. Therefore the VMLP training set was chosen as large as necessary. In the case of the synthetic data shown in Figure A.1 this is 20'000 vector pairs.

For all investigated datasets the verification error achieved with the VMLP was not significantly higher than the Bayes verification error.

A.3   Speech Data

Even though we have seen that VMLPs performed well for synthetic data distributions with a quite complex decision boundary as shown in Figure A.1 it is not sure whether they also provide good results for real world data which have often much more dimensions. Therefore we performed experiments with two tasks from speech processing; namely phoneme verification and speaker verification.

With speech data it was also possible to investigate how the VMLPs perform if they were used for a verification task with classes (i.e. phonemes or speakers) which were not seen during the training.

Phoneme Verification

Features $Feat_{noCms}$ as described in Appendix C.1 were extracted from German words spoken by 4000 speakers and from French words spoken by 3600 speakers. From these speakers, disjoint speaker sets were formed as described in Appendix E.1. Within all sets vector pairs are composed as described in Section 2.3.3. The German VMLPs were trained with the vector pairs of $\mathcal{S}_{G,poly,4}$ and the training was stopped when the verification error reached a minimum on the validation set $\mathcal{S}_{G,poly,5}$. The French VMLPs were trained analogously with $\mathcal{S}_{F,poly,4}$ and $\mathcal{S}_{F,poly,5}$. The used VMLPs had one hidden layer with 140 neurons. For the KNN experiments the validation sets $\mathcal{S}_{G,poly,5}$ and $\mathcal{S}_{F,poly,5}$ were
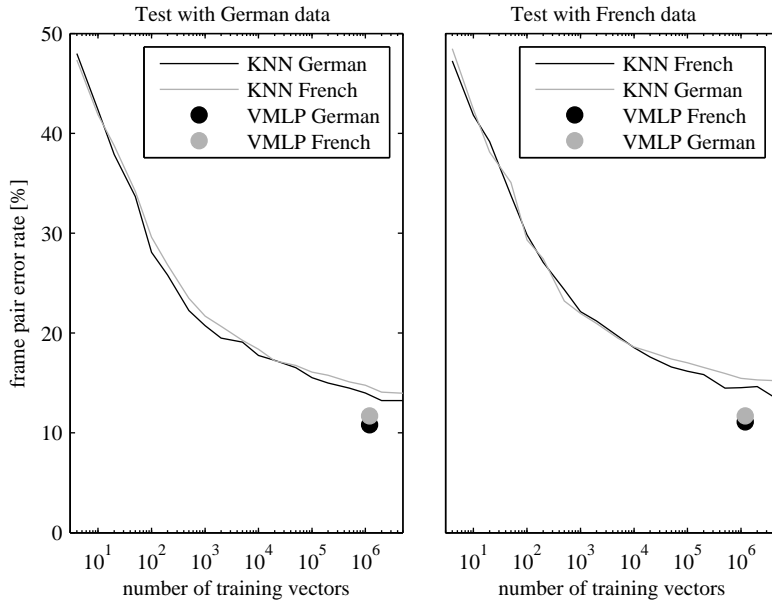
Figure A.3: *Frame pair phoneme verification rate for German test frame pairs on the left and for French test frames pairs on the right. The KNN verification is plotted as a function of the number of KNN training vectors. Results with data and VMLPs from the same language are in black, cross-language results in grey.*

used to determine the best k and the training sets to do the actual classification.

For both languages tests were performed by assigning test vectors either to group G_S or to group G_D as described in Section A.1. Intra-language experiments were performed by using training, validation and test data from the same language (e.g. $\mathcal{S}_{G,poly,4}$, $\mathcal{S}_{G,poly,5}$ and $\mathcal{S}_{G,poly,6}$). In order to evaluate also the cross-language performance, the VMLPs trained for a given language were also evaluated with the task of the other language.

A comparison of the VMLPs and KNN in the intra-language experiments (black line and black dots in Figure A.3) confirmed that the VMLPs yielded results which are close to optimality in a Bayesian sense even though the convergence of the KNN could only be guessed due to the lack of a sufficient number of training vector pairs.

The cross-language KNN experiments (grey line in Figure A.3) showed that the mismatch between the data of the two languages was not very big since the performance was only slightly worse than on data of the same language. Also the VMLPs used in a cross-language scenario (grey dots in Figure A.3) were only marginally worse.

Speaker Verification

We then looked at the verification of speakers, a task which was likely to have a bigger mismatch between training and test data. The data which we used for these experiments was split into three sets. Since the speaker sets from the polyphone database were not suitable for speaker verification experiments we had to use the three-digit database which contains much less speakers. The speaker sets $\mathcal{S}_{G,digit,1}$, $\mathcal{S}_{G,digit,2}$ and $\mathcal{S}_{G,digit,3}$ were taken as training-, validation-, and test set, respectively. The vector pairs were extracted as described in Section 5.2.4. The KNN-verification converged quite slowly if the input vector pairs were a simple concatenation of the two feature vectors $\mathbf{p}_{in} = (\mathbf{x}_1, \mathbf{x}_2)$ as also used for phoneme verification. Therefore we tested additionally a coded version of the input vector pairs $\mathbf{p}_{in} = (|\mathbf{x}_1 - \mathbf{x}_2|, \mathbf{x}_1 + \mathbf{x}_2)$ (see [NP04] for details about the input coding). The used VMLPs had two hidden layers with 70 neurons in the first and 18 in the second layer. Here we used the features $Feat_{spveri}$ as described in Appendix C.1.

The results are shown in Figure A.4. It can be seen that the KNN verification error in function of the training set size decreased much less steeply than in the experiments done with synthetic data and did not even get as low as the verification error of the VMLP. This was possible since the training and test set had some mismatch because the speaker sets are disjoint. Here it could be seen very well that the KNN which is based on coded vector pairs converged with much less training vectors. The VMLP which used coded vector pairs was a bit worse however.
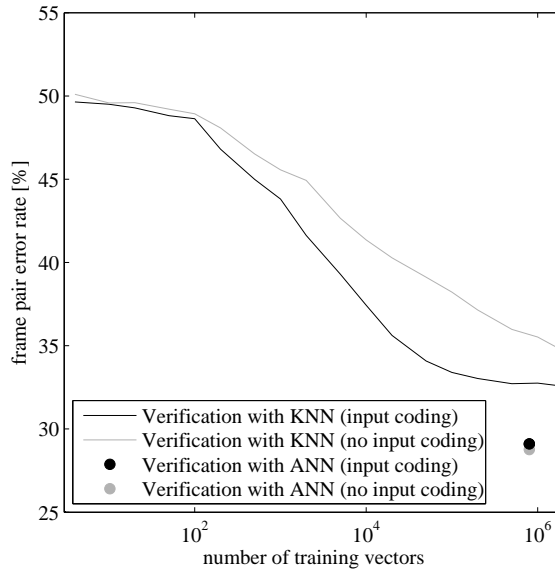
Figure A.4: *Frame pair speaker verification: The KNN error rates decrease with increasing number of KNN training vectors. The error rates of the VMLPs are shown as dots. The error rates for both, KNN and VMLP are given for coded and uncoded input vectors.*

A.4   Concluding Remarks

We could show that the VMLPs have an optimal performance in the Bayesian sense if data with the same distribution was used for training and for testing. In experiments with unmatched data, in particular if discriminating between speakers not seen in the training of the VMLP, we could see that the VMLP rather learned to discriminate between classes of a given task than the actual class distributions.

Appendix B

Qualitative Experiments with the Extended Viterbi Algorithm

This appendix gives qualitative results of the algorithms which are described in Sections 3.2.2 and 3.2.4 to provide an impression of the result of the algorithms.

B.1   Automatic Phonetic Transcriptions

We estimated phonetic transcriptions of words from one or several utterances of each word. For this we used a phoneme loop, i.e. a composite HMM which is a full connection of elementary HMMs which model the German monophones. The monophones are described in Appendix D.

B.1.1 Automatic Transcriptions of Seven Words

The automatically generated transcription of the German names of the weekdays are listed in Table B.1 along with the transcriptions from a pronunciation dictionary ([Dud05]). The transcriptions are given as they were determined from the individual three observation sequences with the normal Viterbi algorithm and as they were determined from the exact K-dimensional algorithm and its approximation from the three available observation sequences.

In the transcriptions generated from individual observation sequences the word were often hardly recognizable. The transcriptions which were generated from several observation sequences look much more appropriate even though they were not completely identical to the transcriptions from the pronunciation dictionary. Noteworthy is also that the transcriptions from several observations often differed much from any of the individual transcriptions. The transcription from the approximative algorithm was not identical to the one of the exact algorithm but both transcriptions looked similarly appropriate. More in depth comparisons were made in Section 3.6.6.

B.1.2 Alignment of Three Utterances

In this example we have a particular look at the determination of the sequence of phonemes which optimally describes three utterances of the German word *Montag* (Monday) spoken by two female and one male speaker. This examples also demonstrates the alignment of the phonemes with the signals as it is done by the exact K-dimensional Viterbi algorithm. In Figure B.1 the signals of the three utterances are plotted. In Figure B.1 a) the joint optimal sequence of phonemes for the three signals is shown and the corresponding positions of the phonemes is indicated with the vertical lines. In Figure B.1 b1) – b3) the optimal sequences of phonemes for the individual signals are given.

Word	Viterbi	K-dim. Viterbi	approx K-dim. Viterbi	pronunc. dictionary
	1 Utterance	3 utterances	3 utterances	
Montag	moʊlŋpaɪ̯k mʊnntha:k mo:mudva:gr	mo:nnta:k	mo:ʊnŋta:k	mo:nta:k
Dienstag	di:nsdapk djufda:kə t͡sgi:ŋstaha:k	di:usdaa:k	di:nstdaa:k	di:nsta:k
Mittwoch	ʔertvɔxʔ mi:tɔx vɪtvox	vɪtvɔx	vɪtvɔx	mɪtvɔx
Donnerstag	rɔo:spda:ks duʋloesta:k dɔmərssda:p	dɔnərsda:k	dɔʊnərsta:k	dɔnɐsta:k
Freitag	dvaɪ̯rpdaʊ̯k t͡szvaɪ̯kpdɛ:ɛrp fvaɪ̯sta:k	fvaɪ̯ta:k	t͡sfvaɪ̯ta:k	fraɪ̯ta:k
Samstag	baʊstha:t ptanxdaa:k zanstææa:k	zaʊnstʔa:k	zaʊnstʔa:k	zamsta:k
Sonntag	fɔʊnpva:k tsvɔʊntaps dɔʊnta̯ua:pk	zoʊnta:pk	zoʊnta:pk	zɔnta:k

Table B.1: *Automatically generated transcriptions of the German names for weekdays. The results of the exact K-dimensional Viterbi algorithm and its approximation as described in Section 3.2.4 are listed along with the pronunciations determined from individual observation sequences and the transcriptions from a pronunciation dictionary.*
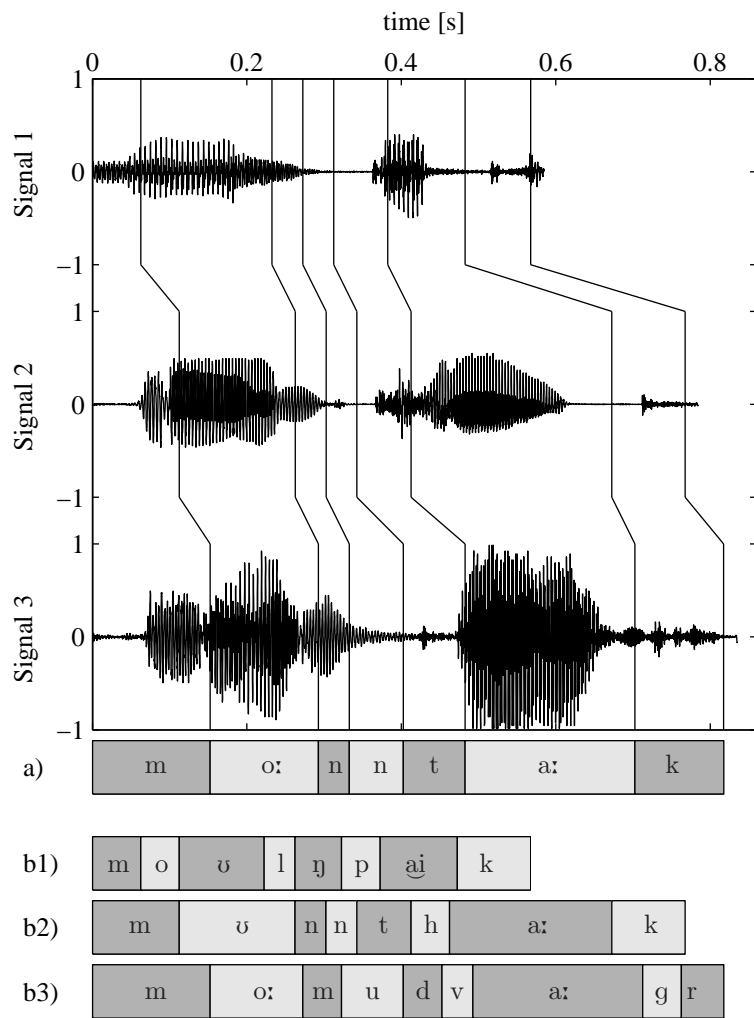
Figure B.1: *Optimal sequence of phonemes for three utterances of the German word* Montag.

Appendix C

Features

This appendix gives the detailed parameters of the features which were used in this thesis in Section C.1. Investigations about the subtraction of cepstral means are presented in Section C.2.

C.1   Feature Description

The main focus of this thesis was on different modeling approaches of isolated word recognition and not on the evaluation of different features for this task. Therefore we used features which are commonly known to yield good results for speech recognition, especially in clean-speech scenarios. We have decided to use Mel cepstral coefficients introduced in [DM80]. Besides LPC cepstral coefficients they are probably the most widely used features for speech recognition.

For two reasons we worked with three variants of Mel frequency cepstral coefficients:

- In [Rey94] was shown that higher cepstral coefficients have a beneficial influence for speaker verification. We could experimentally confirm this result. Therefore we used more cepstral coefficients for the speaker verification features $Feat_{spveri}$.

- Very often the mean values of the cepstral coefficients are computed for a signal and subtracted from the cepstral coefficients. This compensates linear channel distortions. For the case of isolated word recognition this can however be quite dangerous since the signal which the cepstral mean can be computed from is small (i.e. only the signal length of a word is available unless knowledge of a wider context around the word is available). Therefore the cepstral mean may contain phoneme-discriminating information which should not be discarded. Whether the cepstral mean is subtracted or not is therefore a tradeoff between channel compensation and discarding of useful information. Experiments to examine this issue are described in Section C.2.

We always used a short-time analysis with a window length of 37.5 ms and a window shift of 10 ms. All feature vectors were a concatenation of Mel frequency cepstral coefficients and their first temporal derivatives.

The parameters of all feature extractions are listed in Table C.1.

	$Feat_{noCms}$	$Feat_{cms}$	$Feat_{spveri}$
preemphasis coeff.	-0.9		
window	Hamming		
Mel scale break freq.	700 Hz		
number of filters	24	24	34
used coefficients	0–12		1–16
total vector length	26		32
cepstral mean subtr.	no	yes	

Table C.1: *Parameters of the used feature extractions.*

C.2   Investigation of Cepstral Mean Subtraction

For the reasons explained in Section C.1 we tested whether the subtraction of the cepstral mean values is beneficial for isolated word recognition.

C.2.1   Experiments

Speaker-dependent and cross-speaker experiments for French and German were performed according to the tasks 1, 2, 4 and 5 as defined in Appendix E.1. Since these tasks all involve mostly short words we performed additional experiments with German three-digit numbers (tasks 7 and 8) since the cepstral means can be estimated more reliably from these utterances which are on average 1.5 s long.

The DTW recognizer described in Chapter 2 was based on the Euclidean distance measure. The HMM recognizer described in Section 3 used abstract acoustic elements $\mathcal{A}_{WordConstrain}$ of the target language as described in Section 3.5.3.

	word type	DTW		HMM	
		no CMS	CMS	no CMS	CMS
speaker-dependent	long German	85.8	97.8	97.8	99.2
	short German	86.6	86.5	93.5	84.4
	short French	67.5	67.1	86.6	72.7
cross-speaker	long German	72.3	89.2	93.5	96.9
	short German	62.1	65.1	86.3	72.3
	short French	49.7	51.3	80.6	63.1

Table C.2: *Feature evaluation. The recognition rates in % for different languages and scenarios are listed for features which have the cepstral mean subtracted (CMS) and features which don't (no CMS). Results are given both for a DTW recognizer and a HMM recognizer.*

The results are listed in Table C.2. Two big differences can be observed: one between long and short utterances, the other between DTW and HMM recognizer. The difference between long and short utterances comes with no surprise since the cepstral means can be estimated much more accurately on a longer speech segment. Therefore the benefit from the channel compensation dominated over the information loss for longer words. Here it should also be noted that in the database used for the longer utterances one single user used different telephones whereas in the database used for short utterances one speaker used only one telephone. Therefore there was a smaller channel mismatch for the short utterances. This effect was however not present in the

cross-speaker experiments.

The difference between DTW and HMM recognizer is more interesting. It seems that the channel mismatch had a big impact on the DTW recognizer. The channel mismatch could be dealt with much better by the GMMs used in the HMM approach. This is at least partly an effect of the temporal derivatives present in the used feature vectors. These derivatives are also free of linear channel distortions because of the logarithmization in the calculation of the Mel frequency cepstral coefficients.

Since utterance durations in the order of the ones in the tasks for short utterances are much more realistic we performed all other isolated word recognition experiments in this thesis with the mean-uncompensated features $Feat_{noCms}$.

Appendix D

Dictionary-based Recognizer and Used Phonemes

This appendix briefly describes the phone model inventories and the dictionary-based recognizer which were used in this thesis.

D.1   Phonemes

Every phoneme was modeled with a linear three-state HMM. The observation probability density function of each state was modeled with 32 Gaussian mixtures. The Gaussians were parameterized with the weights, the mean vectors and diagonal covariance matrices. Every phoneme was modeled independently of the context, i.e. the models were monophones.

The phone models for both French and German were trained on the respective Polyphone databases (cf. Appendix E.1.1). The phonetic transcriptions were automatically generated from the available orthographic annotations and a pronunciation dictionary. The models

were initialized with a flat start. The training was then continued with Viterbi training (cf. Section 3.4.2). The number of mixtures in each GMM was iteratively doubled until the final 32 mixtures were reached. For every number of mixtures the expectation maximization process was iterated four times, except for 32 mixtures where it was iterated nine times. Except for the very beginning of the training additional silence models were inserted at the beginning and at the end of the sentences and speech pause models were inserted between the words.

No transition probabilities were used in the recognizer.

D.1.1 Phone Model Inventories

The phonemes are listed in Table D.1.

German	ə ç ɛ ɛː ɣ ɪ ŋ ɔ ɔy ʃ ʊ ɣ a aː a͜i a͝u b d
	e eː f g h i iː j k l m n o oː ø øː p
	ʔ r s t t͜s u uː v x y yː z
French	ʃ a (ə) d ɛ ø j ʒ y ɛ̃ u ã k g ɛː t e ɲ
	iː ʔ v s ɥ œ̃ b aː z ɔ̃ w ə f i n ɔ
	œ m l p ʀ o

Table D.1: *List of used phonemes.*

D.2 Dictionary-based Recognizer

Every word of the isolated word recognizer was modeled with a sequence of phonemes according to a pronunciation dictionary. We have used only one pronunciation per word since the systems which we compared the recognizer with also used only one pronunciation per word.

Appendix E

Test Data and Tasks

This appendix gives a detailed description of the experimental data used in this thesis and the speaker subsets which were created from it in Section E.1. The test tasks for isolated word recognition experiments in various sections of the thesis are described in Section E.2.

E.1 Used Databases

E.1.1 Polyphone Database

The Swiss German and Swiss French Polyphone databases ([PSG], [PSF]) were recorded in the German- and French-speaking part of Switzerland, respectively. Both databases were recorded over various telephones including a few mobile phones. We used recordings from 4000 speakers of the German database and from 3600 speakers of the French database. The recordings of each speaker were captured in a single session over a single telephone.

The polyphone databases are available as utterance recordings with orthographic annotations. The utterances have different content types. Some contain single words, but most contain several words (e.g. complete sentences, sequences of numbers or proper names and their

spellings). The Swiss German database contains totally 130500 utterances and the used part of the Swiss French database contains totally 167328 utterances.

The polyphone database was not originally designed to perform test which need several utterances of the same word spoken by a single speaker. To build the utterance-based vocabularies this was however required. Therefore the speakers who had enough occurrences of at least ten words in the available utterances had to be selected from the test speaker set. Therefore data from less speakers was used in the test tasks than there were speakers in the test sets.

To train the abstract acoustic elements and to train the verification multilayer perceptrons several utterances of the same word was needed from the speakers of the training speaker-set. The number of utterances available from each word depended very much on the word. Some words (e.g. number words) were very frequent while from other words only one utterance was available.

Since the exact word positions are not given in the polyphone databases we used forced alignment to extract the words from the databases. This may have led to a few not very precise word boundaries.

Data Sets

Several speaker subsets were formed from the whole speaker population according to Table E.1.

The first three speaker subsets of each language were mutually disjoint and also the last three sets were mutually disjoint.

E.1.2   German Three-Digit Numbers Database

Signals containing 15 different natural numbers spoken in German were recorded by our laboratory. All numbers had three digits (e.g. 398 - dreihundertachtundneunzig). The recordings from the various speakers were recorded over telephone line in several sessions. The speakers were asked to use various telephones including mobile phones. The utterance

purpose	German		French	
	name	number of speakers	name	number of speakers
training	$\mathcal{S}_{G,poly,1}$	3000	$\mathcal{S}_{F,poly,1}$	2400
validation	$\mathcal{S}_{G,poly,2}$	500	$\mathcal{S}_{F,poly,2}$	600
test	$\mathcal{S}_{G,poly,3}$	500	$\mathcal{S}_{F,poly,3}$	600
training	$\mathcal{S}_{G,poly,4}$	1500	$\mathcal{S}_{F,poly,4}$	1500
validation	$\mathcal{S}_{G,poly,5}$	1500	$\mathcal{S}_{F,poly,5}$	1500
test	$\mathcal{S}_{G,poly,6}$	1000	$\mathcal{S}_{F,poly,6}$	600

Table E.1: *List with the names of the different speaker sets derived from the polyphone database and the number of speakers in each set.*

boundaries were manually corrected. The utterances had an average length of about 1.5 seconds.

Data Sets

Several speaker sets were formed from the data of the German three-digit numbers database. These sets are listed in Table E.2. The speaker sets $\mathcal{S}_{G,digit,1}$, $\mathcal{S}_{G,digit,2}$ and $\mathcal{S}_{G,digit,3}$ were disjoint.

purpose	name	number of speakers	gender
training	$\mathcal{S}_{G,digit,1}$	26	male
validation	$\mathcal{S}_{G,digit,2}$	10	male
test	$\mathcal{S}_{G,digit,3}$	13	male
test	$\mathcal{S}_{G,digit,4}$	25	male & female

Table E.2: *List with the names of the different speaker sets and the number of speakers in each set for the three-digit numbers database.*

E.2 Test Tasks for Isolated Word Recognition

We used speaker-dependent, cross-speaker and speaker-independent scenarios both for German and French. This resulted in six main tasks. Additionally there were two tasks for speaker-dependent and cross-speaker scenarios with longer German words (three-digit number).

- *German tasks:* For the German tasks 835 vocabularies were formed by randomly choosing ten words from a pool of mostly short German words. For every vocabulary several words were chosen as test words. The utterances from which the word representations in the vocabulary were formed and the utterances used for each test word were chosen according to the scenario as described below:

  - *Task 1 – speaker-dependent:* The word-representations of each vocabulary were formed from utterances of one speaker. The vocabularies were formed for 64 speakers of the $\mathcal{S}_{G,poly,3}$ speaker set. The utterances of the test words were chosen from the same speaker. Totally 7885 tests were performed.

  - *Task 2 – cross-speaker:* The vocabularies with the same word representations as in task 1 were used. Now the utterances for the test words were however taken from utterances of other speakers than the speaker whom the vocabulary utterances are from. Totally 23655 tests were performed.

  - *Task 3 – speaker-independent:* The word-representations of each vocabulary were formed from utterances of randomly chosen speakers of the $\mathcal{S}_{G,poly,3}$ speaker set except from speakers of the test utterances. The same test utterances as in task 2 were used. Therefore also here 23655 tests were performed.

- *French tasks:* For the French tasks 1882 vocabularies were formed by randomly choosing ten words from a pool of mostly short French words. For every vocabulary several words were chosen

as test words. The utterances from which the word representations in the vocabulary were formed and the utterances used for each test word were chosen according to the scenario as described below:

- *Task 4 – speaker-dependent:* The word-representations of each vocabulary were formed from utterances of one speaker. The vocabularies were formed for 465 speakers of the $\mathcal{S}_{F,poly,3}$ speaker set. The utterances of the test words were chosen from the same speaker. Totally 33371 tests were performed.

- *Task 5 – cross-speaker:* The vocabularies with the same word representations as in task 4 were used. Now the utterances for the test words were however taken from utterances of other speakers than the speaker whom the vocabulary utterances are from. Totally 66742 tests were performed.

- *Task 6 – speaker-independent:* The word-representations of each vocabulary were formed from utterances of randomly chosen speakers of the $\mathcal{S}_{F,poly,3}$ speaker set except from speakers of the test utterances. The same test utterances as in task 5 were used. Therefore also here 66742 tests were performed.

- *German tasks with long words:*

    - *Task 7 – speaker-dependent:* Word recognition experiments were performed for 25 speakers taken from the $\mathcal{S}_{G,digit,4}$ set of the three-digit numbers database. For every speaker around three vocabularies were formed by randomly choosing ten number words. Every vocabulary was tested with approximately 70 numbers spoken by the same speaker. This resulted in a total of 5037 performed tests.

    - *Task 8 – cross-speaker:* Word recognition experiments were performed for 25 speakers taken from the $\mathcal{S}_{G,digit,4}$ set. For every speaker four vocabularies were formed by randomly choosing ten number words. Every vocabulary was tested with approximately 110 number words spoken by each of the other 24 speakers. This resulted in a total of 98440 performed tests.

In order to provide an impression of the words occurring in the test
tasks we give random extracts from the word lists of all test tasks in
Table E.3.

language	tasks	sample words
German	1, 2, 3	aber, auf, das, des, eine, ende, ich, kann, lina, mit nicht, null, raute, sechs, sie, siebzehn, uhr, was, wiederholen, zurück
German	7, 8	siebenhundertvierzehn, sechshundertsiebenundzwanzig
French	4, 5, 6	accent, anna, bien, cent, comme, deux, espace, grave, inconnu, madame, millions, nous, point, quoi, seize, société, tiret, trait, vous, école

Table E.3: *List with sample words from all test tasks.*

Bibliography

[ABA04]   H. Aronowitz, D. Burshtein, and A. Amir. Text inde-
           pendent speaker recognition using speaker dependent word
           spotting. In *Proc. of Interspeech 2004*, 2004.

[AH03]   A. G. Adami and H. Hermansky. Segmentation of speech
           for speaker and language recognition. In *Proc. of Eu-
           rospeech 2003*, pages 841–844, 2003.

[APB10]   A. Asaei, B. Picart, and H. Bourlard. Analysis of phone
           posterior feature space exploiting class specific sparsity and
           mlp-based similarity measure. In *Proc. of ICASSP 2010*,
           2010.

[AVB06]   G. Aradilla, J. Vepa, and H. Bourlard. Posterior-based fea-
           tures and distances in template matching for speech recog-
           nition. In *Proc. of Interspeech 2006*, 2006.

[Bak76]   R. Bakis. Continuous speech recognition via centisecond
           acoustic states. *The Journal of the Acoustical Society of
           America*, 59(S1):S97, April 1976.

[Bau70]   L. E. Baum, et al. A maximization technique occurring in
           the statistical analysis of probabilistic functions of Markov
           chains. *The Annals of Mathematical Statistics*, 41(1):164–
           171, 1970.

[BB93]   L. R. Bahl and P. F. Brown, et al. A method for the
           construction of acoustic Markov models for words. *IEEE*

Transactions on Speech and Audio Processing, 1(4):443–452, October 1993.

[BBdSP88] L. R. Bahl, P. F. Brown, P. V. de Souza, and M. A. Picheny. Acoustic Markov models used in the Tangora speech recognition system. In *Proc. of ICASSP 88*, 1988.

[Bea03]   F. Beaufays, et al. Learning name pronunciations in automatic speech recognition systems. In *Proc. of the 15th IEEE International Conference on Tools with Artificial Intelligence*, 2003.

[Ben07]   M. Benzeghiba. Automatic speech recognition and speech variability: A review. *Speech Communication*, 49:763–786, 2007.

[Bey98]   P. Beyerlein. Discriminative model combination. In *Proc. of ICASSP 98*, 1998.

[BGM97]   P. Bonaventura, F. Gallocchio, and G. Micca. Multilingual speech recognition for flexible vocabularies. In *Proc. of Eurospeech 1997*, 1997.

[BJ04]   K. Bartkova and D. Jouvet. Multiple models for improved speech recognition for non-native speakers. In *Proc. of SPECOM'04*, 2004.

[BM94]   H. Bourlard and N. Morgan. *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers, 1994.

[BN01]   M. Bisani and H. Ney. Breadth-first search for finding the optimal phonetic transcription from multiple utterances. In *Proc. Interspeech 2001*, 2001.

[BP04]   K. Boakye and B. Peskin. Text-constrained speaker recognition on a text-independent task. In *Proc. of Odyssey 2004 - The Speaker and Language Recognition Workshop, Toledo*, June 2004.

[Byr00]   W. Byrne, et al. Towards language independent acoustic modeling. In *Proc. of ICASSP 2000*, 2000.

[Cam02]   W. M. Campbell. Generalized linear discriminant sequence kernels for speaker recognition. In *Proc. of ICASSP 2002*, volume 1, pages 161–164, 2002.

[CE98]   A. Corrada-Emmanuel, et al. Progress in speaker recognition at Dragon Systems. In *Proc. of ICSLP, Sydney, Australia*, 1998.

[CHS06]   P. Charoenpornsawat, S. Hewavitharana, and T. Schultz. Thai grapheme-based speech recognition. In *Proc. of the Human Language Technology Conference of the NAACL Companion Volume: Short Papers*, pages 17–20, June 2006.

[CSR06]   W. M. Campbell, D. E. Sturim, and D. A. Reynolds. Support vector machines using GMM supervectors for speaker verification. *IEEE Signal Processing Letters*, 3(5):308–311, May 2006.

[CSRS06] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff. SVM based speaker verification using a GMM supervector kernel and NAP variability compensation. In *Proc. of ICASSP 2006*, 2006.

[DAB98]   P. Dalsgaard, O. Andersen, and W. B. Barry. Cross-language merged speech units and their descriptive phonetic correlates. In *Proc. of ICSLP, Sydney, Australia*, 1998.

[DB95]   S. Deligne and F. Bimbot. Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams. In *Proc. of ICASSP 1995*, 1995.

[Dem99]   K. Demuynck, et al. Optimal feature sub-space selection based on discriminant analysis. In *Proc. of Eurospeech 1999*, 1999.

[DHS01]   R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2001.

[DM80]   S.B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Speech and Audio Processing*, 28(4):357–366, August 1980.

[Dud05]   Duden.   *Duden - Aussprachewörterbuch,*   6. Auflage.
            Mannheim, Dudenverlag, 2005.

[DW04]   M De Wachter, et al. A locally weighted distance measure
            for example based speech recognition. In *Proc. of ICASSP
            2004*, 2004.

[DW07]   M. De Wachter, et al. Template-based continuous speech
            recognition. *IEEE Transactions on Speech and Audio Pro-
            cessing*, 15(4):1377–1390, May 2007.

[EB04]   D. Elenius and M. Blomberg. Comparing speech recogni-
            tion for adults and children. In *Proc. of FONETIK*, 2004.

[EP06]   A. El Hannani and D. Petrovska-Delacrétaz. Using data-
            driven and phonetic units for speaker verification. In *Proc.
            of Odyssey 2006*, 2006.

[EP07]   A. El Hannani and D. Petrovska-Delacrétaz.   Compar-
            ing data-driven and phonetic n-gram systems for text-
            independent speaker verification. In *Proc. of First IEEE
            International Conference on Biometrics*, 2007.

[Fon97]   V. Fontaine, et al. Nonlinear discriminant analysis for im-
            proved speech recognition. In *Proc. of Eurospeech 1997*,
            1997.

[FSM03]   J. E. Flege, C Schirru, and I. R. A. MacKay. Interaction
            between the native and second language phonetic subsys-
            tems. *Speech and Communication*, 40:467–491, July 2003.

[GBP07]   M. Gerber, R. Beutler, and B. Pfister.   Quasi text-
            independent speaker verification based on pattern match-
            ing. In *Proc. of Interspeech*, pages 1993–1996. ISCA, 2007.

[GC89]   L. Gillick and S. Cox. Some statistical issues in the com-
            parison of speech recognition algorithms. In *Proceedings of
            the ICASSP*, pages 532–535, 1989.

[GKP07]   M. Gerber, T. Kaufmann, and B. Pfister. Perceptron-based
            class verification. In *Proc. of Non Linear Speech Processing
            (NOLISP)*, 2007.

[GL94]   J.-L. Gauvain and C.-H. Lee. Maximum a posteriori esti-
          mation for multivariate Gaussian mixture observations of
          Markov chains. *IEEE Transactions on Speech and Audio
          Processing*, 2(2):291–299, April 1994.

[GL98]   C. Godin and P. Lockwood. DTW schemes for continuous
          speech recognition: A unified view. *Computer Speech and
          Language*, 3:169–198, 1998.

[GP05]   M. Gerber and B. Pfister. Quasi text-independent speaker
          verification with neural networks. MLMI'05 Workshop, Ed-
          inburgh (United Kingdom), July 2005.

[GP08]   M. Gerber and B. Pfister. Fast search for common seg-
          ments in speech signals for speaker verification. In *Proc. of
          Interspeech*, pages 375–378, Brisbane (Australia), Septem-
          ber 2008.

[GSP05]   D. Gillick, S. Stafford, and B. Peskin. Speaker detection
          without models. In *Porc. of the ICASSP 2005*, 2005.

[Har01]   S. Harbeck. *Automatische Verfahren zur Sprachdetektion,
          Landessprachenerkennung und Themendetektion*. PhD the-
          sis, Universität Erlangen-Nürnberg, 2001.

[Hay99]   S. Haykin. *Neural Networks: A Comprehensive Foundation
          (2nd Edition)*. Prentice-Hall, 1999.

[HES00]   H. Hermansky, D. P. W. Ellis, and S. Sharma. Tandem
          connectionist feature extraction for conventional HMM sys-
          tems. In *Proc. of the ICASSP 2000*, volume 3, pages 1635–
          1638, 2000.

[HH92]   M. Hwang and X. Huang. Subphonetic modeling with
          Markov states - Senone. In *Proc. of ICASSP 1992*, 1992.

[HH03]   M. Hébert and L. P. Heck. Phonetic class-based speaker
          verification. In *Proc. of Interspeech 2003*, pages 1665–1668,
          2003.

[Hie93]   J. L. Hieronymus. ASCII phonetic symbols for the world's
          languages: Worldbet. *Journal of the International Phonetic
          Association*, 1993.

[HM05]   A. Hagen and A. Morris. Recent advances in the multi-
          stream HMM/ANN hybrid approach to noise robust ASR.
          Computer Speech and Language, 19(1):3–30, 2005.

[HO99]   S. Harbeck and U. Ohler. Multigrams for language identi-
          fication. In *Proc. of Eurospeech 1999*, 1999.

[Hua01]   C. Huang, et al. Analysis of speaker variability. In *Proc. of
          Eurospeech 2001*, 2001.

[HW99]   T. Hain and P. C. Woodland. Dynamic HMM selection
          for continuous speech recognition. In *Proc. of Eurospeech*,
          1999.

[Ita75]   F. Itakura. Minimum prediction residual principle applied
          to speech recognition. *IEEE Transactions on Acoustics,
          Speech, and Signal Processing*, 23(1):67–72, February 1975.

[Jel76]   F. Jelinek. Continuous speech recognition by statistical
          methods. *Proceedings of the IEEE*, 64(4):532–556, April
          1976.

[Jel98]   F. Jelinek. *Statistical Methods for Speech Recognition*. The
          MIT Press, 1998.

[JR90]   B.-H. Juang and L. R. Rabiner. The segmental k-means al-
          gorithm for estimating parameters of hidden Markov mod-
          els. *IEEE Transactions on Acoustics, Speech, and Signal
          Processing*, 38(9):1639–1641, September 1990.

[JY94]   D. A. James and S. J. Young. A fast lattice-based ap-
          proach to vocabulary independent wordspotting. In *Porc.
          of ICASSP 1994*, 1994.

[KA98]   N. Kumar and A. G. Andreou. Heteroscedastic discrimi-
          nant analysis and reduced rank HMMs for improved speech
          recognition. *Speech Communication*, 26:283–297, 1998.

[KD04]   Patrick Kenny and Pierre Dumouchel. Experiments in
          speaker verification using factor analysis likelihood ratios.
          In *Proc. of Odyssey 2004*, 2004.

[KN02]   S. Kanthak and H. Ney. Context-dependent acoustic mod-
           eling using graphemes for large vocabulary speech recogni-
           tion. In *Proc. of ICASSP*, 2002.

[KSS03]   M. Killer, S. Stüker, and T. Schultz.   Grapheme based
           speech recognition. In *Proc. of Eurospeech 2003*, pages
           3141–3144, Geneva, September 2003.

[Kun04]   S. Kunzmann, et al.   Multilingual acoustic models for
           speech recognition and synthesis. In *Proc. of ICASSP 2004*,
           pages 745–748, 2004.

[LBG80]   Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector
           quantizer design. *IEEE Transaction on Communications*,
           28(1):84–95, January 1980.

[LHG03]   A. D. Lawson, D. M. Harris, and J. J. Grieco. Effect of for-
           eign accent on speech recognition in the NATO n-4 corpus.
           In *Proc. of Eurospeech 2003*, 2003.

[Lip87]   R. P. Lippmann. An introduction to computing with neural
           nets. *IEEE ASSP Magazine*, 3(4):7–25, April 1987.

[LJC02]   Y.-L. Lin, T. Jiang, and K.-M. Chao. Efficient algorithms
           for locating the length-constrained heaviest segments with
           applications to biomolecular sequence analysis. *Journal of
           Computer and System Sciences*, 65:570–586, 2002.

[Mat04]   M. Matton, et al. A discriminative locally weighted distance
           measure for speaker independent template based speech
           recognition. In *Proc. of Interspeech 2004*, 2004.

[MB07]   J. Mariéthoz and S. Bengio. A kernel trick for sequences
           applied to text-independent speaker verification systems.
           Pattern Recognition, 40(8):2315–2324, August 2007.

[ME91]   N. Merhav and Y. Ephraim. Hidden Markov modeling us-
           ing a dominant state sequence with application to speech
           recognition. *Computer Speech & Language*, 5(4):327–339,
           1991.

[MH05]   N. Mirhafori and A. O. Hatch, et al. ICSI's 2005 speaker recognition system. In *Proceedings of ASRU 2005*, 2005.

[Mil07]   D. R. H. Miller, et al. Rapid and accurate spoken term detection. In *Proc. of Interspeech 2007*, 2007.

[MJ99]   H. Mokbel and D. Jouvet. Derivation of the optimal set of phonetic transcriptions for a word from its acoustic realizations. *Speech Communication*, 29:49–64, September 1999.

[MRG85]   J. Makhoul, S. Roucos, and H Gish. Vector quantization in speech coding. *Processings of the IEEE*, 73(11):1551–1588, November 1985.

[MRS07]   J. Mamou, B. Ramabhadran, and O. Siohan. Vocabulary independent spoken term detection. In *Proc. of the 30th annual international ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.

[MSFB07]   D. Matrouf, N. Scheffer, B. Fauve, and J.-F. Bonastre. A straightforward and efficient implementation of the factor analysis model for speaker verification. In *Proc. of Interspeech 2007*, 2007.

[New96]   M. Newman, et al. Speaker verification through large vocabulary continuous speech recognition. In *Proc. of ICSLP 1996*, 1996.

[NP04]   U. Niesen and B. Pfister. Speaker verification by means of ANNs. In *Proc. of ESANN 2004, Bruges (Belgium)*, pages 145–150, April 2004.

[PB03]   B. Pfister and R. Beutler. Estimating the weight of evidence in forensic speaker verification. In *Proc. of Eurospeech 2003*, pages 701–704, Geneva, September 2003.

[PCC00]   D. Petrovska-Delacrétaz, J. Cernocký, and G. Chollet. Segmental approaches for automatic speaker verification. *Digital Signal Processing*, 10(1-3):198–212, Janaury 2000.

[PG05]   A. Park and A. Glass. Towards unsupervised pattern discovery in speech. In *Proc. of ASRU 2005*, 2005.

[PG06]   A. Park and J. R. Glass. A novel DTW-based distance
           measure for speaker segmentation. In *Proc. of the Spoken
           Language Technology Workshop*, 2006.

[Pic09]   B. Picart. Improved phone posterior estimation through
           k-NN and MLP-based similarity. IDIAP research report,
           IDIAP, 2009.

[PJ02]   A. Park and Hazen T. J. ASR dependent techniques for
           speaker identification. In *Proc. of ICSLP 2002*, 2002.

[PSF]   swiss-french-polyphone. ELDA (Evaluations and Language
           resources Distribution Agency).
           (http://www.elda.org/catalogue/en/speech/S0030.html).

[PSG]   swiss-german-polyphone. IDIAP research institute.
           (http://www.idiap.ch/scientific-research/resources/swiss-
           german-polyphone).

[PV00]   R. Paredes and E. Vidal. A nearest neighbor weighted mea-
           sure in classification problems. In *Proc. of the VII SNRFAI*,
           2000.

[Rab86]   L. R. Rabiner, et al. A segmental k-means training pro-
           cedure for connected word recognition. *AT&T technical
           journal*, 64(3):21–40, May 1986.

[Rab89]   L. R. Rabiner. A tutorial on hidden Markov models and
           selected applications in speech recognition. *Proceedings of
           the IEEE*, 77(2):257–286, February 1989.

[Rey94]   D. A. Reynolds. Experimental evaluation of features for
           robust speaker identification. *IEEE Transactions on Speech
           and Audio Processing*, 2(4):639–643, October 1994.

[RLS83]   L. R. Rabiner, S. E. Levinson, and M. M. Sondhi. On
           the application of vector quantization and hidden Markov
           models to speaker-independent, isolated word recognition.
           The Bell System Technical Journal, 62(4):1075–1105, April
           1983.

[RP90]   R. C. Rose and D. B. Paul. A hidden Markov model based
            keyword recognition system. In *Proc. of ICASSP*, volume 1,
            pages 129–132, 1990.

[RQD00]   D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker
            verification using adapted Gaussian mixture models. *Digi-
            tal Signal Processing*, 10:19–41, 2000.

[Sao00]   G. Saon, et al. Maximum likelihood discriminant feature
            spaces. In *Proc. of ICASSP*, 2000.

[SC04]   S. Salvador and Ph. Chan. FastDTW: Toward accurate
            dynamic time warping in linear time and space. In *KDD
            Workshop on Mining Temporal and Sequential Data*, 2004.

[SH90]   F. K. Soong and E.-F. Huang. A tree-trellis based fast
            search for finding the n best sentence hypotheses in con-
            tinuous speech recognition. In *Proc. of the Workshop on
            Speech and Natural Language*, 1990.

[Sin08]   S. M. Siniscalchi, et al. Toward a detector-based universal
            phone recognizer. In *Proc. of ICASSP 2008*, 2008.

[SM07a]   F. Stouten and J.-P. Martens. Dealing with cross-lingual
            aspects in spoken name recognition. In *Proc. of ASRU
            2007*, 2007.

[SM07b]   F. Stouten and J.-P. Martens. Recognition of foreign names
            spoken by native speakers. In *Proc. of Interspeech 2007*,
            2007.

[SNN01]   G. Stemmer, E. Nöth, and H. Niemann. Acoustic modeling
            of foreign words in a German speech recognition system. In
            Proc. of Eurospeech 2001, 2001.

[SSP95]   T. Svendsen, F. K. Soong, and H. Purnhagen. Optimizing
            baseforms for HMM-based speech recognition. In *Proc. of
            Eurospeech 1995*, 1995.

[Stu02]   D. E. Sturim, et al. Speaker verification using text-
            constrained Gaussian mixture models. In *Proc. of ICASSP
            2002*, 2002.

[SW00]   T. Schultz and A. Waibel. Polyphone decision tree special-
            ization for language adaptation. In *Proc. of ICASSP 2000*,
            2000.

[SW01]   T. Schultz and A. Waibel. Language-independent and
            language-adaptive acoustic modeling for speech recogni-
            tion. *Speech Communication*, 35, 2001.

[Ueb01]   U. Uebler. Multilingual speech recognition in seven lan-
            guages. *Speech Communication*, 35:53–69, 2001.

[VC01]   D. Van Compernolle. Recognizing speech of goats, wolves,
            sheep and ... non-natives. *Speech Communication*, 35:71–
            79, 2001.

[Vin71]   T. K. Vintsyuk. Element-wise recognition of continuous
            speech composed of words from a specified dictionary. *Cy-
            bernetics and Systems Analysis*, 7(2):361–372, 1971.

[VZ70]   V. M. Velichko and N. G. Zagoruyko. Automatic recogni-
            tion of 200 words. *Int. J. Man-Machine Studies*, 2:223–234,
            1970.

[Wei95]   M. Weintraub. LVCSR log-likelihood ratio scoring for key-
            word spotting. In *Proc. of ICASSP 1995*, pages 297–300,
            1995.

[Wel89]   J. C. Wells. Computer-coded phonemic notation of indi-
            vidual languages of the European Community. *Journal of
            the International Phonetic Association*, 19:31–54, 1989.

[Wen97]   F. Weng, et al. A study of multilingual speech recogni-
            tion. In *Proc. of Eurospeech 1997*, volume 1, pages 359–362,
            1997.

[WG99]   J. Wu and V. Gupta. Application of simultaneous decod-
            ing algorithms to automatic transcription of known and
            unknown words. In *Proc. of ICASSP 1999*, 1999.

[WPN+00]   F. Weber, B. Peskin, M. Newman, A. Corrada-Emmanuel,
            and L. Gillick. Speaker recognition on single- and mul-
            tispeaker data. *Digital Signal Processing*, 10(1–3):75–92,
            January 2000.

[WTK98]   G. Williams, M. Terry, and J. Kaye. Phonological elements as a basis for language-independent ASR. In *Proc. of IC-SLP 1998*, 1998.

[You89]   S. J. Young, et al. Token passing: a conceptual model for connected speech recognition systems. Technical report, Cambridge University, 1989.

[You94]   S. J. Young, et al. Tree-based state tying for high accuracy acoustic modelling. In *Proc. of the workshop on Human Language Technology*, 1994.

[YS03]   H. Yu and T. Schultz. Enhanced tree clustering with single pronunciation dictionary for conversational speech. In *Proc. of Interspeech 2003*, 2003.

[ZCMS04]   Q. Zhu, B. Chen, N. Morgan, and A. Stolcke. On using MLP features in LVCSR. In *Proc. of ICSLP 2004, Jeju, Korea*, 2004.

Curriculum Vitae

1975   Born in Wattwil, Switzerland

1983-1990   Primarschule in Wattwil

1990-1992   Sekundarschule in Wattwil

1992-1997   Kantonsschule in Wattwil

1997-2002   Studies in electrical engineering at ETH Zürich

2002   Diploma in electrical engineering (dipl. Ing.)

2002-2004   Engineer at SCANCO Medical AG

2004-2011   Research assistant and PhD student at the Speech Processing Group, Computer Engineering and Networks Laboratory, ETH Zürich