Aalto University
School of Science
Degree Programme of Computer Science and Engineering

Peter Smit

Stacked transformations for foreign accented speech recognition

Master's thesis

Espoo, May 2011

Supervisor: Docent Mikko Kurimo, D.Sc.(Tech.)

Instructor: Janne Pylkkönen, M.Sc.

Aalto University
School of Science

Aalto University School of Science Degree Programme of Computer Science and Engineering	ABSTRACT OF MASTER'S THESIS

Author:	Peter Smit				
Title:	Stacked transformations for foreign accented speech recognition				
Number of pages:	vii + 54	Date:	May 2011	Language:	English
Professorship:	Information and Computer Science			Code:	T-61
Supervisor:	Docent Mikko Kurimo, D.Sc.(Tech.)				
Instructor:	Janne Pylkkönen, M.Sc.				

Abstract:

Nowadays, large vocabulary speech recognizers exist that are performing reasonably well for specific conditions and environments. When the conditions change however, performance degrades quickly. For example, when the person to be recognized has a foreign accent the conditions could mismatch with the model, resulting in high error rates.

The problem in recognizing foreign accented speech is the lack of sufficient training data. If enough data would be available of the same accent, from numerous different speakers, a well performing accented speech model could be built.

Besides the lack of speech data, there are more problems with training a complete new model. It costs a lot of computational resources and storage space to train a new model. If speakers with different accents must be recognized, these costs explode as every accent needs retraining. A common solution for preventing retraining is to adapt (transform) an existing model, such that it better matches the recognition conditions.

In this thesis multiple different adaptation transformations are considered. Speaker Transformations are using speech data from the target speaker, Accent Transformations use speech data from different speakers, who have the same accent as the speech that needs to be recognized. Neighbour Transformations are estimated with speech from different speakers that are automatically determined to be similar to the target speaker.

Novelty in this work is the stack wise combination of these adaptations. Instead of using a single transformation, multiple transformations are 'stacked together'. Because all adaptations except the speaker specific adaptation can be precomputed, no extra computational costs at recognition time occur compared to normal speaker adaptation and the adaptations that can be precomputed are much more refined as they can use more and better adaptation data. In addition, they need only a very small amount storage space, compared to a retrained model.

The effect of Stacked Transformations is that the models have a better fit for the recognition utterances. When compared to no adaptation, improvements up to 30% in Word Error Rate can be achieved. In adaptation with a small number (5) of sentences, improvements up to 15% are gained.

Keywords: automatic speech recognition, foreign accent recognition, linear transformation, stacked transformations

Acknowledgements

This Master's Thesis was made in the Speech Group of the Department of Information and Computer Science at the Aalto University, School of Science. The Adaptive Informatics Research Centre and the EMIME project have funded my position and I'm thankful for that.

I would like to thank Mikko Kurimo, leader of the Speech Group, for being my supervisor and Janne Pylkkönen for being my instructor. Janne was also my office mate, as was Reima Karhila, and I thank them for all discussions we have had together, about speech recognition or otherwise, and all the help they gave when I had questions.

My wife, Maria, has been a great support to me throughout my whole study and an extremely good help in checking all the language and spelling in this thesis. I am so grateful to have her as my wife.

Espoo, May 31st, 2011

Peter Smit

Contents

List of Figures

List of Tables

Symbols and Abbreviations

$o(t)$	Observation vector at time t
μ	Mean vector of a multivariate Normal Distribution
Σ	Covariance matrix of a multivariate Normal Distribution

ASR	Automatic Speech Recognition
CMLLR	Constrained Maximum Likelihood Linear Regression
CV	Cross Validation
DRT	Dimensionality Reduction Technique
DSP	DSP Speech Corpus, Finnish accented speech corpus recorded at Aalto Universities Digital Signal Processing Course in 2010 and 2011
EM	Expectation-Maximization (algorithm)
HMM	Hidden Markov Model
LM	Language Model
LVCSR	Large-Vocabulary Continuous Speech Recognition
MLED	Maximum Likelihood Eigenvoice Decomposition
MLLR	Maximum Likelihood Linear Regression
PCA	Principal Component Analysis
SI	Speaker Independent
SD	Speaker Dependent
UED	University of Edinburgh, referring to the foreign accented corpus recorded there
WSJ	Wall Street Journal (American English speech corpus)
WSJCAM	Wall Street Journal Cambridge (British English speech corpus)

Chapter 1

Introduction

As long as machines have existed, people have been trying to make them understand our language, the spoken word. In comic books and movies often speech can be recognized by computers without a single error. In reality however, computers still have problems recognizing and understanding what we speak to them.

The first speech recognizer was made in 1952 in the Bell Telephone Laboratories (K. Davis, Biddulph, and Balashek 1952), a machine that could recognize spoken digits with high accuracy, between 97 and 99% success rate. Since then a lot has changed. Computers were developed and grew in computation and memory capacity exponentially, with the amount of transistors doubling each two years (Moore's law). Still, no applications exist that can recognize a broad range of speech, in vastly different environments with such a high accuracy and speed as humans can.

There is also good news. When a lot of speech data is available from a single person, and only the speech of this particular person needs to be recognized, or in case the domain of words spoken is limited, speech recognizers can recognize with very high accuracies. Therefore, a mobile phone can recognize which contact to call when a name is said and automatic telephone services can determine with which customer representative to connect to, when told what the problem is. Also, dictation and transcription services exist but almost all books and articles, including this thesis, are still typed and not dictated.

The main reason why digit recognizers or voice command operated applications work is because of the limited domain of words that need to be recognized. Also, the words can appear in limited combinations, mostly defined by some simple rules. When recognizing full sentences, the number of possible words increases greatly and the grammars are not easily described as a set of rules.

Recognition is even more complicated in cases where also humans have sometimes trouble understanding each other. Voices vastly differ from each other, as do speaking styles. When people speak in their own local dialect or people are not native to a language, they are often hard to understand. Still, when it is heard often enough, the dialect or accent will become perfectly understandable.

This thesis tries to find methods of improving recognition by using data of other speakers with similar features. Like it will be easier for a person to understand a foreign accent after hearing multiple persons speaking it, a speech recognizer should be able to take advantage of available data with a foreign accent by training or adapting statistical models that describe this particular accent and utilize this to improve the recognition.

1.1 Goals of this thesis

The goal of this thesis is to research methods to improve the speech recognition results by adapting the recognizer with speech data of speakers other than the target speaker. These speakers could be either selected in supervised (speakers pre-grouped by gender or dialect) or unsupervised (speakers grouped automatically) manner. The findings will be tested on different foreign accent data sets to show whether the methods are generic between accents. Also experiments will be done to measure the influence of the amount of speech data available for adaptation.

The main purpose is to improve adaptation for speakers with only very limited amount of speech data available. This thesis also considers the practical needs for real-world applications, taking into account requirements such as speed and required computational resources.

1.2 Speech Recognition at Aalto University

The Speech Group at Aalto University, where this work was done, is part of the department of Information and Computer Science and is led by Docent Mikko Kurimo.

The group has a long history of speech recognition, especially for the Finnish language. The first PhD thesis in speech recognition was Jalanko (1980), which developed a system for labeling speech, useful in a small vocabulary (1000 words) system. Late 1980's, a phonetic typewriter was developed ("Workstation-based phonetic typewriter"; Torkkola et al. 1991). Kurimo (1997) applied Self Organizing Maps and Learning Vector Quantization (both developed in the same department) for the training of Mixture Model HMMs.

After that a lot of research has been done on Large Vocabulary Finnish Speech recognition. Statistical morph-based large vocabulary speech recognition was introduced in

V. Siivola et al. (2003) and further developed in Mathias Creutz (2006); Hirsimäki et al. (2006).

The group has developed it's own Speech Recognizer which is used for research purposes. The focus of research has been mainly on the Finnish Language. Finnish is an agglutinative language, meaning that words can be split easily into smaller parts, named morphemes. The Speech Group has, together with the Natural Language Processing group, made vast advances in developing morphological language models. Besides Finnish these methods also have been applied on Estonian and Turkish (Kurimo, Puurula, et al. 2006).

Currently the group researches on methodologies for many aspects of large vocabulary speech recognition and also speaker adaptation for speech synthesis. One sub group focuses on noise robust speech recognition, in which methods are developed to enable recognition for noisy environments. Another focus area is in acoustic model training and adaptation algorithms, where this thesis belongs to. The other research areas are training and adaptation methods for language modeling and the applications of large vocabulary speech recognition in synthesis, retrieval and translation of speech.

1.3 The EMIME project

This work was funded by the European Community project for Effective Multilingual Interaction in Mobile Environments which had the goal of developing personalized speech to speech translation on a mobile device. The potentials of this project are high, as it would enable people to naturally speak on the phone with each other, even though they don't speak or understand the language of their conversation partner.

Important part is also the personalization of the speech synthesis. When talking with an elderly women, it is unnatural to hear a voice of a young boy speaking. Or when speaking to a man that is careful with his words, speaking slowly, and a deep voice, it is unnatural to hear a fast speaking voice on a higher tone. A third example is maybe even more important, when we hear Finnish men speaking English, we expect some degree of a Finnish accent in the English speech, which helps us to recognize this person's voice.

The personalization problem is related to the synthesis part of the application, but can be also translated to speech recognition, where similar models are used. As it is hard to personalize a voice in synthesis, for example with an accent, the equivalent problem in speech recognition is to recognize this particular type of voice.

This thesis focuses on recognizing foreign accented speech, but these methods could later also be applied to synthesis, potentially making to easily create accented synthesized voices and provide rapid speaker adaptation.

Chapter 2

Components and models in a speech recognition system

The task of an Automatic Speech Recognition (ASR) system is to automatically transcribe speech to text or commands. ASR systems are for example used in phone services or dictation services. Where in TV-series and movies computers seem to be able to perfectly and instantaneously recognize any speech in any language, in reality the performance in speed and accuracy depends heavily on the conditions, the speaker, the type of speech, and the vocabulary and language used.

For small vocabulary recognition systems, such as digit recognition or command recognition, systems exist that can recognize with a very high accuracy. Therefore, current research is mostly focused on so-called Large-Vocabulary Continuous Speech Recognition (LVCSR) systems that can recognize anything said in a certain language. Also for LVSCR already well-performing recognizers exist, but they are often very sensitive for change in speaker, noise-conditions or differences in speaking style (e.g. dictation vs. free speech). Improving the robustness of recognizers, or enabling recognizers to adapt to new conditions is one of the focus areas of the ASR community (Baker, Deng, Glass, et al. 2009). Another focus is the usage of high volume corpora. Some people try to create enormous well transcribed, high quality corpora, while others try to utilize the enormous amount of speech data available on the Internet. (Baker, Deng, Khudanpur, et al. 2007).

State-of-the-art ASR recognition systems consist out of four separate models. First, Feature Extraction transforms a speech signal into sequential data vectors, called feature vectors. The feature vectors are matched with their corresponding sounds in the Acoustic Model. The Acoustic Model describes for every sound in the language, how the feature vectors are distributed. These sounds often generally match with the phones used in a phonetic alphabet. Words are described in the Lexicon, which is a list of all different

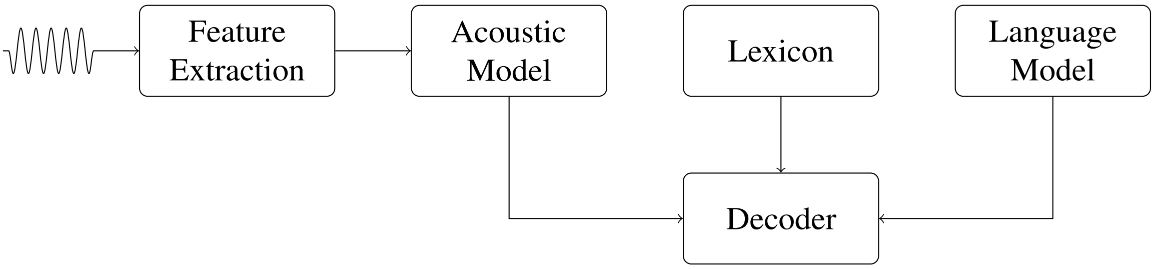

Figure 2.1: The main components of an ASR system

pronunciations for each word that must be recognized by the system. The last part is the language model that describes how the language that is recognized is structured. In English for example, after the word ‘red’, it is possible that the word ‘wine’ is used. However, after the word ‘wine’, the word ‘red’ wouldn’t be on the right place in the sentence. The Language Model describes constraints of the language, to make recognition of normal sentences possible. These four different models are shown in Figure 2.1. The fifth component of the ASR system is the decoder. The decoder finds the sequence of words that fit the best in the combination of the acoustic and language model, given an input audio signal.

In the next sections, all the four different models are described. After that the decoding with the whole system is explained. The last part is the metrics used for evaluation of ASR systems. Because this thesis focuses on improving the acoustic model for foreign accented speech, the acoustic model will be described most extensively.

2.1 Speech Signal

Sounds, including speech signals, are transported as pressure differences in the air. The pressure changes differ in frequency and these frequencies are perceived by the human ear as tones and their amplitude corresponds with loudness. To digitize the sounds a microphone records a high number of samples each second of the pressure differences. The number of samples per second is called the sample rate and the number of bits precision used to record the pressure differences is called the bit-size. Figure 2.2 shows an example of a sound signal for the words “speech recognition”. Here a sample rate of 16 kHz and a precision of 16 bits is used for the accuracy of each frame (meaning each frames can have $2^{16} = 65536$ different values).

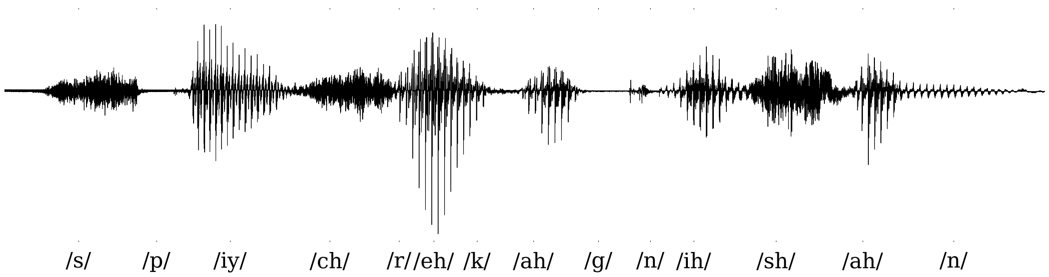

Figure 2.2: The sound signal for the words “speech recognition”

2.2 Feature Extraction

The digital speech signal is not immediately useful for ASR. As speech is formed by the frequencies of the sound, the speech should be transformed from the time domain to the frequency domain. Because sound is a mixture of signals with different frequencies, the signal in the frequency domain will be multidimensional, with every dimension describing the amplitude of the components in some frequency range.

The most common way of extracting frequencies are Fourier Transforms and in this case, because we have digitized data, the Discrete Fourier Transform (DFT) (Cooley, Lewis, and Welch 1969). As our signal is quasi-stationary (O’Shaughnessy 1987, p. 40), i.e. stationary over a short time span, we calculate the transform over short, overlapping time windows, which is also called the Short Time Fourier Transform (STFT). To smooth the overlap between windows, Hamming windows are used. Typically the window size used in ASR is between 20 and 25 milliseconds. A common overlap for the windows is 10 ms. This results in 100-125 frames per second, with each frame being a large vector of Fourier coefficients. In Figure 2.3 the STFT spectrum (the energies of different frequencies) is shown for the same data as used in Figure 2.2. Here a window of 25 ms is used.

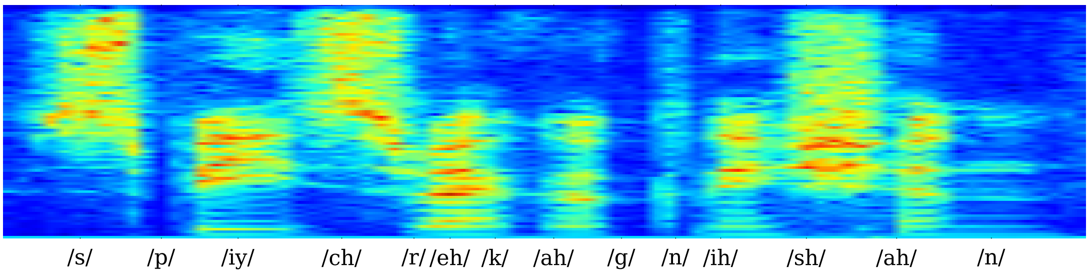

Figure 2.3: STFT spectrum for the words “speech recognition”

Not all frequencies are equally important when recognizing speech. Therefore, also the human ear is more sensitive to certain frequencies. Inspired by this detail the mel-scale was developed by Stevens, Volkmann, and Newman (1937) to model the human ear. A mel-scale filter bank can be used to transform a frequency spectrum into a mel-scale spectrum. The mel filterbank defines a number of 'bins', that are equally spaced on the mel-scale. The signal is divided over these bins, and as the bins for the most important frequencies are smaller, these frequencies are amplified. On the frequency scale, triangular windows are used to assign the frequencies. An example of the windows is shown in Figure 2.4. Commonly between 24 and 40 bins are used as it utilizes the effect of the mel-filter bank, but also reduces the dimensionality of the data. The formulas are summarized in X. Huang, Acero, and Hon (2001, pp. 316–318). The resulting mel-spectrum is shown in Figure 2.5.

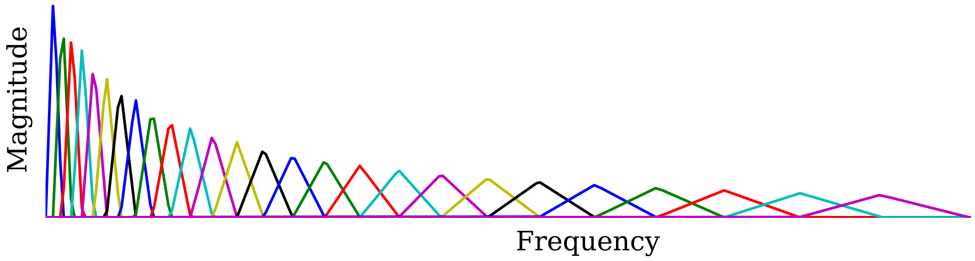

Figure 2.4: Melfilterbank with 26 components

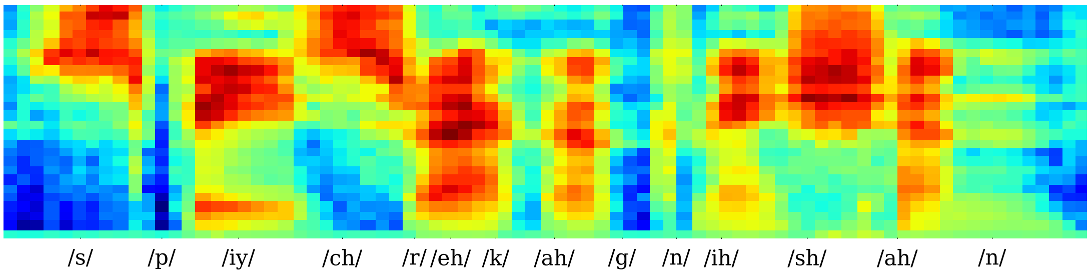

Figure 2.5: Mel spectrum for the words "speech recognition"

It is desirable for the acoustic model to operate on lower-dimensional, de-correlated features. Thus, the logarithm is taken of the features and then the Discrete Cosine Transform (DCT) is used to transform the mel spectrum into the mel cepstrum. As a consequence the energy will be collected in the lower indices with the first component being only the energy. Often the number of final components is thirteen. In Figure 2.6 the result after the DCT is shown. Visually the different phonetic units are not as easily recognizable as with the spectrum, but for the acoustic models these vectors are much better.

These 13-dimension vectors are called the Mel-Frequency Cepstral Coefficients (MFCC). MFCC's were first used in Mermelstein and S. Davis (1980) for speech recognition. Currently, often the first and second derivatives are added to retain more temporal information,

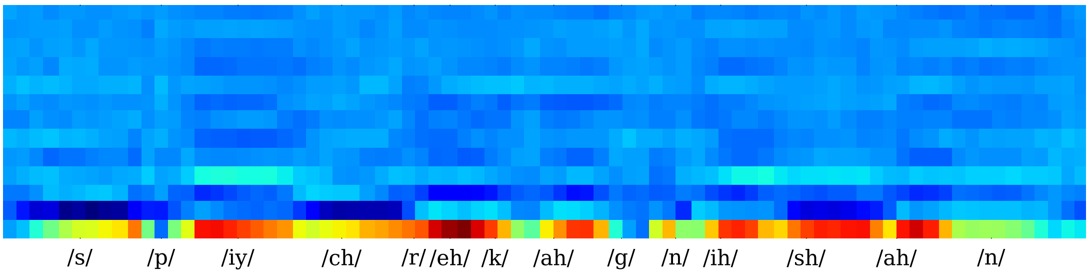

Figure 2.6: Mel cepstrum for the words “speech recognition”

improving recognition performance (Furui 1986), which makes the feature vectors 39-dimensional.

2.2.1 Feature level normalization

Multiple methods exist to enhance the robustness of the features for different environments, such as noise on the background or type of microphone. A basic method is Cepstral Mean Subtraction (CMS), which is used for all experiments done in this thesis. CMS is a simple technique that normalizes for each utterance the features. As a result, the conditions that are constant in an utterance, such as microphone type, are normalized away (Atal 1974).

2.3 Acoustic Model

As there are many options for choosing the feature extraction method, multiple different acoustic models are possible. Most of the current state of the art systems use Hidden Markov Models (HMM) with Gaussian Mixture Model (GMM) emission distributions.

This section will first define how phonemes are selected and then how their acoustics can be modeled with GMMs and their temporal properties with HMMs.

2.3.1 Phoneme set

An important decision for the acoustic model is which phonetic alphabet to use. Unlike for example Finnish, the English language does not always have a straight translation between a letter in the normal alphabet and it’s pronunciation. For example the words toy and cow both have an ’o’ as middle letter, still the pronunciation is completely different. Further a word can have different pronunciations depending on its meaning or the dialect or accent of the speaker.

Multiple phonetic alphabets have been developed that can be used to transcribe a word into phonemes that have a direct translation with a pronounced sound. The most known one is the phonetic alphabet of the International Phonetic Association (IPA[1]) (Ladefoged 1990) which describes all sounds used by languages all over the world. Any single language will only use a subset of these sounds. Another commonly used alphabet for English is the ARPABET, which is used for the recognizers in this thesis. The phonemes and example words for ARPABET are shown in Table 2.1, where the first section is the original set published by the Advanced Research Projects Agency (ARPA, nowadays DARPA) and the second section the extension made for British English. For clarity, to distinguish normal letters from pronunciation phonemes, the phonemes are enclosed by '/' signs.

When a phoneme set is used, there must exist a lexicon for the words to be recognized. The lexicon is a list of words in a language and all their possible phonetic transcriptions. Most of the time not all words in a language are represented in the lexicon, but at least all words that are used in the training data and language model. In a lexicon a single word can have different phoneme transcriptions, to illustrate, the word 'the' could be pronounced as 'thee' or 'thuh'. Additionally, different words could share the same phoneme transcription such as 'two' and 'too'. The lexicons used in this thesis are described in Section 5.1.2.

Triphones

Often the pronunciation of a phoneme is affected by the phonemes pronounced before and after it. To model this difference, a model could be created for every different context a phoneme appears in. The phoneme 'a' could expanded to all different contexts of 'a', e.g. 't-a+s' which means an 'a' preceded by a 't' and succeeded by an 's'. If we would have 38 different phonemes plus a silence phoneme (like in the ARPABET), this would mean $39 \cdot 39 = 1521$ models would be used for each phoneme.

Triphones are used to get even more accurate and precise acoustic models. The disadvantage is the size increase of the model and the need for more calculations when decoding. The middle way is to have different emission distributions for often used triphones and shared emission distributions for less used triphones. The process of tying these Gaussians is described in Section 2.3.4. In the next sections the word phoneme is used for both single phonemes (monophones) and triphones.

[1] The acronym IPA is often used as an abbreviation for International Phonetic Alphabet, even though that is technically incorrect

Phoneme	**Example**	**Translation**
/AA/	odd	AA D
/AE/	at	AE T
/AH/	hut	HH AH T
/AO/	ought	AO T
/AW/	cow	K AW
/AY/	hide	HH AY D
/B/	be	B IY
/CH/	cheese	CH IY Z
/D/	dee	D IY
/DH/	thee	DH IY
/EH/	Ed	EH D
/ER/	hurt	HH ER T
/EY/	ate	EY T
/F/	fee	F IY
/G/	green	G R IY N
/HH/	he	HH IY
/IH/	it	IH T
/IY/	eat	IY T
/JH/	gee	JH IY
/K/	key	K IY
/L/	lee	L IY
/M/	me	M IY
/N/	knee	N IY
/NG/	ping	P IH NG
/OW/	oat	OW T
/OY/	toy	T OY
/P/	pee	P IY
/R/	read	R IY D
/S/	sea	S IY
/SH/	she	SH IY
/T/	tea	T IY
/TH/	theta	TH EY T AH
/UH/	hood	HH UH D
/UW/	two	T UW
/V/	vee	V IY
/W/	we	W IY
/Y/	yield	Y IY L D
/Z/	zee	Z IY
/ZH/	seizure	S IY ZH ER
/AX/	aboard	AX B AO D
/EA/	aero	EA R OW
/IA/	fear	F IA R
/OH/	bombs	B OH M Z
/UA/	bourne	B UA N

Table 2.1: The ARPABET phonemes, with in the second section the phonemes exclusively used for British English.

2.3.2 Gaussian Mixture Models

In the acoustic model, sounds are modeled with Gaussian Mixture Models (GMM). Each phoneme could have a single GMM, but when Hidden Markov Models (HMM) are used, even phonemes are split into separate parts, e.g. sub phones, that all have their own GMM.

A GMM is a weighted sum of one or more Multivariate Gaussian distributions. So for a GMM with K components, the distribution is defined as

$$X \sim \sum_{k}^{K} \phi_k \mathcal{N}(\mu_k, \Sigma_k)$$

with μ_k and Σ_k being the mean vector and covariance matrix of Gaussian k and ϕ_k being the weight of Gaussian k.

If there would be no temporal structure between frames, and training a sufficiently different model for each acoustic sound could be done, it would enable recognizing for each the frame the most likely phoneme. In reality, however, sounds are so similar that strict separation of sounds is not possible. Besides, a frame-wise phoneme classification is still not easily transferred to transcription in words as one misclassified frame could mess up the recognition and a single sound could easily have different labellings. The main problem is the lack of temporal information stored in a single GMM. Still, they are very useful, and in fact used as the basic building blocks of more advanced models, e.g. HMMs as described in the next section.

Parameters of a GMM can be trained with the Expectation Maximization (EM) algorithm using the Maximum Likelihood criterion (Bilmes 1997). The EM procedure is an iterative procedure that works in two steps. One step is assign each feature to a mixture component and the second step updates the mixture weights, means and variances. Every iteration of the procedure guarantees an increase in likelihood of the model. EM with HMMs and GMMs is explained in Section 2.3.3.

2.3.3 Modeling temporal structures with Hidden Markov Models

To take into account temporal structure in the feature vectors, Hidden Markov Models (HMM) can be used. HMMs are an extension of a Markov chain.

A Markov chain is a network of states. Every time moment the system will be in one state. The probability of being in a state at a certain time, is only dependent on the state of the previous time frame. A demonstration of a Markov chain with two states; Rain and Sunshine is shown in Figure 2.7. Between the states are arcs with the transition probabilities. States can have self-loops, meaning that a state can appear several consecutive times. In

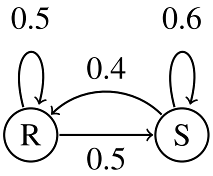

Figure 2.7: Example of a Markov Chain

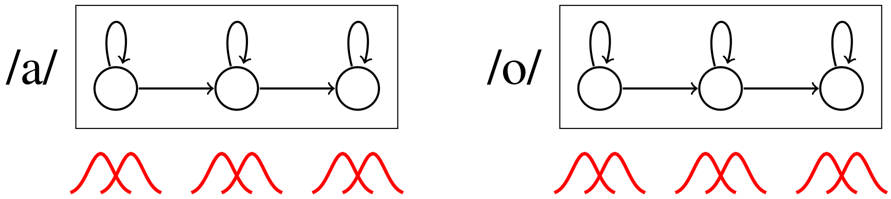

Figure 2.8: Examples of three-state left-to-right Hidden Markov Models for two phonemes

the example, when on day one it rains, the probability for it to be raining on day two is calculated by simply taking the transition probability from Rain to Rain, which is 0.5. The probability for it to be raining on day three is still easy, but involves summing up the options that there is sunshine on day two and raining on day three or raining on both days. The probability for the first is $0.5 \cdot 0.4 = 0.2$ and for the second $0.5 \cdot 0.5 = 0.25$. The total probability is 0.45 for it to be raining on day three, if it rains on day one.

In formulas, it is written that the variable X (which can be rain or sunshine) at time t is only dependent on X of time $t-1$:

$$P(X_t \mid X_1,\dots,X_{t-1}) = P(X_t \mid X_{t-1})$$

Where in a Markov model the state can be observed, in HMM's the real states are not discretely observed, such as the Rain or the Sunshine, instead they have an 'emission distribution'. The Sunshine and Rain example can be in a different country, and the only data available is is the humidity percentage. Even though the Sunshine and Rain can't be observed, from the humidity can be inferred what most likely is the case.

For speech this means that the hidden state is the label of an acoustic sound, a phone or a sub phone and emission distribution is a distribution of the realizations of this speech, the feature vectors. From a feature vector, not directly can be observed what the state is that generated it, but the probability it is emitted from any state can be calculated.

In Figure 2.8 two HMM's for two different vowels is shown. Each vowel is separated in a begin, middle and end part, a so called three-state HMM. The emission distributions are GMMs which were introduced in the previous section.

The notations for a HMM are as follows. The HMM has two kinds of parameters, the transition probabilities between states and the emission probabilities for each state. These parameters together are denoted as λ. Now there are two sequences, a sequence of observations $O = (o_1, \ldots, o_T)$, the feature vectors, and $Q = (Q_1, \ldots, Q_T)$, the sequence of hidden states that has produced (and 'explains') this observations. For recognition, the model is used in such way, that the sequence Q with the highest probability $P(Q|\lambda, O)$ is found.

For the purposes of ASR, left-to-right models are used, meaning that all states are ordered and no arcs exist from a state to any of it's previous states. Logically, a backward arc in a phoneme HMM would mean that the beginning of a phoneme is pronounced again after the middle part, practically left-to-right models are used as it is a requirement for computational feasible decoding with the Viterbi algorithm. Rabiner (1989) gives more thorough examples of HMMs in ASR.

Training with Baum-Welch

An HMM can be trained with the Expectation Maximization (EM) algorithm, which in case of an HMM with GMM emission distributions is called the "Baum-Welch" algorithm (Baum et al. 1970). A more introductory explanation is given in Bilmes (1997). The auxiliary function is reasonably simple:

$$Q(\lambda, \lambda') = \sum_{q \in \mathscr{Q}} \log P(o, q|\lambda) P(O, q|\lambda')$$

Here λ' are our current (previously estimated or guessed) parameters of the HMM and λ the new parameters. $\mathscr{Q}$ are all possible sequences of length T, where O is the observed data, also of length T. The Baum-Welch algorithm increases the value of this function with each iteration, until a maximum is reached.

In the Baum-Welch algorithm, a composite HMM is created. The composite HMM exists out of all HMM's combined after each other, according to the available transcription. For each time frame of the speech data, the occupation probability for each hidden state of the composite HMM is calculated. When the state occupation probabilities are known, the parameters of the output distributions (μ, Σ) can be re-estimated.

2.3.4 Tying of Gaussians

A large ASR system can have a lot of Gaussians. In case the GMM's have 16 Gaussians per mixture and a triphone, three-state model with 20 phonemes is used, there would be

$16 \cdot 3 \cdot 20 \cdot 19 \cdot 19 = 346560$ Gaussians.

A great number of Gaussians has several disadvantages. First, there is not enough data to train every Gaussian and it is unlikely that every triphone occurs so often that $16 \cdot 3 = 48$ Gaussians could be trained. Some triphones such as 't-p-t' may not occur at all. Secondly, a computationally expensive part of the recognition procedure is the calculation of occupation probabilities for each frame from each Gaussian. Hence, a smaller number of Gaussians would speed up calculations.

Reducing the number of Gaussians can be done by tying similar Gaussians together (Odell 1995; Young, Odell, and Woodland 1994). The similarity can be expressed in a supervised manner, as a list of rules about which phonemes are vowels or consonants, nasal or not nasal, stopping etc. The similarity can also be expressed in an unsupervised way, as a distance measure between Gaussian distributions. The supervised or unsupervised rules are used to build a decision tree. Decision trees are similar to the regression class trees used in adaptation which are described in Section 3.2.1

2.4 Language Model

The language model describes the structure of a language. It tries to prevent the construction of malformed sentences and uses the information of its grammar to rank possible output hypotheses.

The most common language model is the *n*-gram language model. In an *n*-gram language model, the likelihood of a word is defined by its $n-1$ previous words.

Table 2.2 shows an excerpt from an *n*-gram model. It shows the log-probability for different words after the sequence 'in new'. Clearly, in the model 'in New York' seems to be the most popular phrase. The phrase 'in new year's' is still likely enough to be in the model, but has a much lower probability than the 'in New York' phrase. Unlikely transcriptions are not mentioned at all in the language model to reduce the size and complexity of the model. An extensive explanation of *n*-gram language models can be found in Manning (1999, pp. 191-224).

In recognition, the likelihood of different hypothesis will be calculated from the language model and joined with the acoustic probabilities to determine to most probable transcription.

-0.0802	in	new	york
-1.5867	in	new	jersey
-1.8131	in	new	orleans
-1.8744	in	new	hampshire
-2.0407	in	new	england
-2.2626	in	new	york’s
-2.2680	in	new	zealand
-2.2734	in	new	capital
-2.3450	in	new	orders
-2.3787	in	new	mexico
-2.4550	in	new	delhi
-2.4550	in	new	haven
-2.4990	in	new	cash
...			
-4.1225	in	new	towers
-4.1225	in	new	twenty
-4.1225	in	new	u.
-4.1225	in	new	visions
-4.1225	in	new	work
-4.1225	in	new	workers
-4.1225	in	new	year’s

Table 2.2: An excerpt from an *n*-gram language model; the log-probabilities for different words after the sequence ‘in new’.

2.5 Decoding

As shown in Figure 2.1 in the beginning of this chapter, the decoding stage in ASR utilizes all parts of the model. From an utterance that needs to be recognized, the feature vectors are extracted and the probabilities for all states are calculated. The decoder builds a search network and generates a number of hypotheses. The hypotheses are ranked according to their joint acoustic and language model probability.

This decoding is done with the Viterbi algorithm which was proposed in Viterbi (1967) and explained in Forney (1973). In the Viterbi algorithm sequence of phonemes can be found that best describes an sequence of observation vectors from a HMM. The probabilities of the language model are taken jointly with the acoustic probabilities to select the path that is both good according to the acoustic and according to the language model.

2.6 ASR evaluation

The evaluation of an ASR system can be done in multiple ways. Some measures can be done to describe the fit of the acoustic and language models, or the whole system can be evaluated at once. For an English LVCSR, the latter is often done with help of the Word Error Rate (WER) of a recognition run. To detect real improvements in the WER, statistical significance tests are used.

2.6.1 Recognition run

Evaluating the whole performance of an ASR system must not use any data that is used in the training of the ASR system. Therefore, the recognition utterances must be from speakers who were not used in the development of the acoustic model and the recognition sentences must be different than the ones used in any training files. Most corpora have predefined sets for training and evaluation, sometimes even a separate development set, and if possible these sets should be adhered to, to make comparisons between different studies possible.

2.6.2 Cross validation

A common technique used in Machine Learning, but less in Speech Recognition is the use of cross validation (CV) (Kohavi 1995). This technique splits the data set in to N parts, and trains and evaluates the models with different subsets of the data. For example, N recognitions are done where the evaluation data is one part, and the other $N-1$ parts are used for training.

The advantage of this technique is that the data is used more efficiently, as there is no need to excluded one big evaluation set from training. On the other hand, more models have to be trained and evaluated, often resulting in more need of computing resources. Consequently, ASR not often uses CV. In many cases, the corpora are so big that only a small part of the data needs to be used for evaluation. Furthermore, training and evaluation of ASR systems are computationally expensive, making CV unattractive. Still, in this thesis CV is used because the bilingual datasets do not have enough different speakers available that there would be enough training data left if a subset of them would be set apart for evaluation.

2.6.3 Word Error Rate

For evaluation of the whole ASR system, the Word Error Rate (WER) can be calculated by comparing the recognized text and the real transcription of the recognition data. The error measure is the minimum number of steps to get from the recognized text to the transcription, where a step is an insertion, deletion, or substitution of a single word. The number of steps is taken as a percentage of the number of words in the reference transcription to calculate the error percentage. In principle it is even possible to get a higher error than 100%, when the length of inserted texts exceed the length of the reference transcription.

2.6.4 Statistical Significance Testing

The WER is a good measure to evaluate different recognition techniques or models. Nevertheless, when comparing two methods it must be taken into account that there are other factors than the quality of the technique or the model that can play a role in the final result. Often there is a factor of randomness and chance involved in such complicated model.

Statistical significance tests are used to evaluate whether two results are significantly different. In case the difference is not significant, the better result could be just the result of chance.

The common way of comparing recognition results are matched-pair tests, as advised in Gillick and Cox (1989). In this work, we use the Wilcoxon signed-rank test. The Wilcoxon signed-rank test determines for how many of the utterances, one model performs better than the other. In case for all utterances the result improves, it is definitely significant. If however for almost half of the utterances the result degrades, and for the other big half improves, the average will be still better, but the result is not significant.

Chapter 3

Adaptation

A speech recognizer can come across a wide variety of utterances. Speakers can be different, noise levels can vary and the topic spoken about can range from the weather to rocket science.

Often a model is trained for a certain type of condition, such as a specific speaker, a certain amount of noise or a particular speaking style. Training models, however, for all different conditions is unfeasible, especially, because there are often no samples available in advance of the exact conditions of the utterances that need to be recognized. To overcome this problem, a model can be adapted to a specific condition. Common adaptations are noise adaptation (Compernolle 1989), speaker adaptation and language adaptation (Bacchiani and Roark 2003). The adaptations in this work are about speaker adaptations, adaptations that account for the variety in speech between different persons. Section 3.1 looks into the differences between different types of models, speaker independent or speaker dependent, and in Section 3.2 speaker adaptation is explained. Section 3.3 reviews different methods for adapting to (foreign) accents.

3.1 Speaker-Independent and Speaker-Dependent systems

There are two different kinds of ASR systems. The first category of systems are Speaker-Dependent (SD) systems, which are able to recognize a single speaker. SD models are trained with speech data from this single speaker, to be able to later very well recognize this single speaker. Except for the advantage that a very high performance can be achieved, there are a lot of drawbacks to this approach. A speaker that wants to be recognized must first speak training sentences, that are transcribed. Depending on the desired accuracy, several hours of speech are needed. For normal end-users this is most of the time not feasible, taking simply too much time and costing too much resources for storage and transcription of files.

The second category of models are Speaker-Independent (SI) models. SI models recognize most speakers reasonably well, but do not excel for any particular speaker. SI models are obtained by training the model with speech collected from a variety of different speakers. In this way the model captures the variety of pronunciation possibilities, even enabling the model to recognize new speakers whose data was not used in training. Compared to SD models, the advantages are remarkable. No training data is needed when a new voice is to be recognized, which makes it easy for recognizers to start recognizing new speakers. In addition, a moderate amount of transcribed training data from multiple speakers is already commonly available and only one model needs to be stored for all speakers. The drawback is the reduced accuracy compared to a SD model.

3.2 Speaker Adaptation methods

Speaker Adaptation tries to find ways to transform, or adapt a Speaker-Independent model to a Speaker-Dependent model, with the help of only a very little, possibly not transcribed, training (adaptation) data. Speaker Adaptation enables to get high accuracy results like in an SD model, without the drawbacks for recording, computation and storage.

As the difference between speakers is often in the acoustic domain, the parameters that are of interest to be adapted are the parameters of the emission distributions, the means and variances of the Gaussians in the GMMs. Even though there could be also optimal values for the transition parameters of the HMMs, these only give relatively small improvements and are hard to calculate with a limited amount of data. Further, adaptation in the language model could be done for single speakers, but a lot more sentences are needed than for acoustic model adaptation. Therefore, language model adaptation is often done topic wise.

Woodland (2001) defines three classes of speaker adaptation techniques; linear regression techniques, maximum a posteriori techniques, and eigenvoice or clustering techniques.

Linear Regression techniques are methods such as Maximum Likelihood Linear Regression (MLLR) and Constrained Maximum Likelihood Linear Regression (CMLLR). These techniques perform a linear transformation on the parameters of the Gaussian models. MLLR and CMLLR are the most common used adaptation techniques. CMLLR is explained in later sections as it is heavily used in the experiments.

Just as in other machine learning applications, instead of Maximum Likelihood, also Maximum a Posteriori (MAP) adaptation can be adopted. In MAP adaptation, the SI model is used as prior and observations of features are used to estimate a posterior distribution (Gauvain and C.-H. Lee 1994). One problem is that for a Gaussian to be updated with MAP, it must be observed in the adaptation data. As a lot of sentences are needed to observe enough data for all Gaussians, more adaptation data is needed for MAP techniques than for example for linear regression techniques. If a lot of data is available MAP will likely perform better than linear regression. Because of the need of much adaptation data, MAP is used less often than linear regression methods.

The third class of adaptation methods are the eigenvoice and clustering techniques (Gales 2000; Kuhn, Nguyen, et al. 1998), which create a space of voices, with every voice being represented as a mixture of base voices. For a new person, maximum likelihood methods can estimate new parameters. In this thesis eigenvoice methods are not used as adaptation method, but as clustering technique and are described more detailed in Section 3.2.2

3.2.1 Linear Regression Methods

Maximum Likelihood Linear Regression

Maximum Likelihood Linear Regression (MLLR) is a method proposed in Gales and Woodland (1996); Leggetter and Woodland (1995), and is a method for doing a linear transform on the mean and variance parameters of the Gaussian distributions in such way that the likelihood of the adaptation data is maximized.

With MLLR, the mean and covariance matrices of the multivariate Gaussians are transformed with different transforms. The mean μ is transformed with matrix $\mathbf{A}$ and vector $\mathbf{b}$ to form the adapted mean $\hat{\mu}$ with

$$\hat{\mu} = \mathbf{A}\mu + \mathbf{b}$$

The variance transform is a bit more complicated. From the covariance matrix Σ the inverse Choleski factor $\mathbf{B}$ is derived, to be transformed with the transformation matrix $\mathbf{H}$. The new covariance matrix $\hat{\Sigma}$ is

$$\hat{\Sigma} = \mathbf{B}^T\mathbf{H}\mathbf{B}$$

Constrained MLLR

In contrast to normal MLLR, Constrained MLLR transforms the means and variances of the Gaussian distributions with the same transformation matrix (Digalakis, Rtischev, and Neumeyer 1995; Gales 1998).

The new transformation matrices are:

$$\hat{\mu} = \mathbf{A}^{-1}\mu + \mathbf{b}$$
$$\hat{\Sigma} = \mathbf{A}^{-1}\Sigma\mathbf{A}^{-1T}$$

Because often diagonal covariance matrices are used, it is favorable to preserve them as it makes computation of likelihoods easier. With CMLLR however, the covariances are always transformed into full matrices. To circumvent this the feature vectors can be transformed instead, before any likelihood calculations are done. Instead of the above equations, the feature transform becomes now

$$\hat{\mathbf{o}}(t) = \mathbf{A}\mathbf{o}(t) - \mathbf{b}$$

Regression Class Trees

Normally in CMLLR and MLLR methods, all distributions are transformed with the same transform, though, this does not have to be the case. MLLR and CMLLR could also be applied to all Gaussians separately. Although it sounds appealing, some complications arise. For 39-dimensional distributions (as common for MFCC vectors), at least a few hundred frames must be used for each Gaussian to give a robust transform that generalizes to new speech. It is unfeasible to collect that much adaptation data, especially because some Gaussians are so rare that they are used only very few times, even in training.

An alternative approach is to group Gaussians together in so-called Regression Class Tree (Gales 1996). All Gaussians will be assigned to exactly one Regression Class, and all Gaussians in the same class are using the same transformation. An example of a Regression Class Tree is given in Figure 3.1. A Regression Class Tree is a binary tree, with only the leaf nodes containing objects. The objects could be Gaussians, or groups of Gaussians; for example all the Gaussians belonging to one phoneme or all Gaussians belonging to one mixture. In the following text the word Gaussian is used, but the procedures are the same for groups of Gaussians.

The Regression Class Tree creation procedure is as follows. First for all Gaussians, the statistics are collected, specifically, by assigning each frame in the training data to the best matching Gaussian. Then all the Gaussians are collected in the root node and an iterative procedure starts. The node with the biggest occupation is split into two siblings. Splitting a node happens with the k-means algorithm ($k = 2$), where the Kullbach-Leibler divergence acts as distance measure between Gaussians. The splitting results in two groups of Gaussians, with the Gaussians in a single group being similar to each other. When the

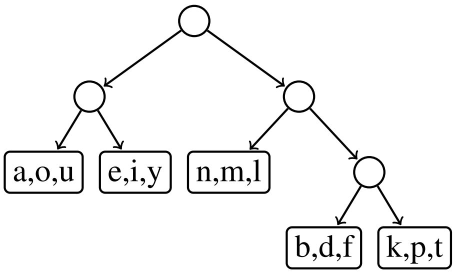

Figure 3.1: Regression Class Tree with phonemes as unit

number of leaf nodes is equal to a predefined number, or when there are no nodes anymore that can be split, the algorithm terminates.

Applying the Regression Class Tree in adaptation works as follows. In (C)MLLR for each adaptation matrix statistics are collected, which will be used for calculating the matrix. Instead of collecting the statistics for the whole model, as would be done when adapting with only one matrix, the statistics are collected separately for each leaf node. When all statistics are collected each leaf node is evaluated whether it has enough statistics. Having enough statistics depends on available utterances that contain frames emitted by the Gaussians in the leaf node. The amount of statistics required is approximately 1000-4000 frames but depends on the model and type of adaptation trained.

If a leaf node does not contain enough statistics, its statistics are merged with the statistics of its sibling. Going back to Figure 3.1, if the node ‘b,d,f’ does not contain enough statistics, the statistics of nodes ‘b,d,f’ and ‘k,p,t’ are merged together to a new node ‘b,d,f,k,p,t’. This procedure repeats itself until all leaf nodes have enough statistics or only the root node is left over. Now for each leaf node an adaptation matrix is calculated according to the normal procedures. When storing the adaptation matrix, also the Gaussians that are applicable for the matrix must be stored in order to know at recognition time what Gaussians to transform with the adaptation matrix.

The size of the tree is not really relevant, as long as it ‘big enough’. The real number of adaptation matrices used only depends on the adaptation data. For normal recognition the number of final adaptation matrices will be quite small.

3.2.2 Eigenvoices

Eigenvoices is an adaptation method, or more precisely a concept of methods, that uses a low dimensional space to describe different speakers. The parameters of a speaker are the weights for a mixture of parameter vectors that construct a recognition model.

Eigenvoices were first proposed in Kuhn, Junqua, et al. (2000); Kuhn, Nguyen, et al.

(1998). The principle is to train Speaker Dependent models for a number of speakers, with each model being of the same structure and dimensions. The parameters of the models are transformed into a ‘supervector’ for each model. Afterwards, a Dimensionality Reduction Technique (DRT) is applied to find the bases of the eigenvoice space (eigenspace). In principle every DRT would work, but Principal Component Analysis (PCA) (Jolliffe 2002) is the most commonly used technique.

When the eigenspace is created, multiple methods can be used to estimate the eigenvoice parameters for the new speakers. The process is called eigenvoice decomposition and the most common method is Maximum Likelihood Eigenvoice Decompostion (MLED). For both eigenspace creation and eigenvoice decomposition there are a wide range of methods available. For example, instead of a DRT, a maximum likelihood estimation of the eigenspace can be made (Nguyen, Wellekens, and Junqua 1999). Another popular variation is to not train separate SD models, but to use MLLR adapted models directly (Botterweck 2000; K. Chen and H. Wang 2001) or indirectly (N.-C. Wang et al. 2001).

In this work eigenvoices are not applied as adaptation, but the eigenvoice parameters are used to find similar speakers by looking to the similar speakers in the eigenspace. Both eigenspace computation and decomposition are utilized in the original way and are therefore more thoroughly explained in the following sections.

Eigenspace computation with (C)MLLR adapted models

To calculate the bases of the eigenspace, it is the most convenient to train an SI model and transform it with a CMLLR adaptation to get an SD model for each speaker. This removes the need of training a separate model for each speaker.

A super vector is created by combining and ordering all the mean parameters of the model. The ordering is arbitrary but must be the same for all speakers. With PCA the supervectors are reduced to a small number of parameters, typically between 3 and 10.

Because acoustic models with triphones have a enormous amount of mean parameters, making PCA infeasible, in this work only the single phone models are used for the eigenvoice computations. Using single phone models also reduces the computational needs for decomposition.

The eigenvoice parameters of the training speakers can be directly reconstructed by using the scores obtained with PCA. Also an Eigenvoice Decomposition method could be applied on the training data and ideally the resulting eigenvoice parameters will be the same for both methods.

Maximum Likelihood Eigen Decomposition

Maximum Likelihood Eigen Decomposition (MLED) finds the eigenvoice parameters that maximize the likelihood of the evaluation data (Kuhn, Junqua, et al. 2000; Kuhn, Nguyen, et al. 1998), by applying the Expectation Maximization (EM) algorithm. As common in EM, an auxiliary function $Q(\lambda,\hat{\lambda})$ is defined, in this case

$$Q(\lambda,\hat{\lambda}) = -\frac{1}{2}P(O \mid \lambda)\sum_{s}\sum_{t}\gamma_s(t)\mathbf{f}(\mathbf{o}_t,s)$$

with $\mathbf{f}(\mathbf{o}_t,s)$ being the Gaussian log-likelihoods, here defined as:

$$\mathbf{f}(\mathbf{o}_t,s) = (n\log(2\pi)+\log|C_s|+\mathbf{h}(\mathbf{o}_t,s))$$

and

$$\mathbf{h}(\mathbf{o}_t,s) = (\mathbf{o}_t-\hat{\mu}_s)^T C_s^{-1}(\mathbf{o}_t-\hat{\mu}_s)$$

where s is a state, n is the number of features, $\mathbf{o}_t$ the observation vector at time t, $C_m^{(s)-1}$ the inverse covariance in state s, $\hat{\mu}_s$ the new adapted mean of state s and $\gamma_s(t)$ the current likelihood $L(s|\lambda,\mathbf{o}_t)$. These likelihoods are the occupation probabilities, calculated with the Baum-Welch algorithm (as explained in chapter 2.3.3).

Q can be minimized analytically, resulting in the closed form

$$\sum_{t}\sum_{s}\gamma_s(t)\mathbf{e}_s^T C_s^{-1}\mathbf{o}_t = \mathbf{w}^T\sum_{t}\sum_{s}\gamma_s(t)\mathbf{e}_s^T C_s^{-1}\mathbf{e}_s$$

where $\mathbf{e}_s$ are the eigenbases in state s and $\mathbf{w}$ are the eigenvoice parameters. This closed form can be easily solved as it has the same number of equations as unknowns.

3.2.3 Supervised and unsupervised adaptation

All mentioned speaker adaptation methods take a number of utterances and their transcription to estimate the adaptation. In real world systems however, not always the transcription of the adaptation sentences is available.

When the transcriptions are known, for example because the speaker read from a prompt or the utterances are manually transcribed, the adaptation is called supervised. When the transcriptions are not known, the baseline recognizer is used to recognize an initial transcription. This transcription is not perfect, but most adaptation algorithms will still work correctly in estimating a good adaptation with them. After that the recognition is done with adaptation to get the better recognition result. In principle this procedure could be even repeated, but often the final result will only improve little and the computational costs are high (the cost another adaptation + recognition run). The latter method is called unsupervised adaptation.

3.2.4 Speaker adaptation in real world recognition scenarios

One of the advantages of speaker adaptation compared with training a speaker dependent model is that it is possible to estimate an adaptation in a short time, possibly while the user is waiting for it. The scenario for a first time user is that a speaker independent model exist, with no knowledge of the speaker that is going to be recognized. The system could now either choose to ask the user for uttering some predefined utterances to do supervised adaptation, or to recognize the first sentences without adaptation and to estimate an unsupervised transform with this initial recognition.

In the second scenario, the user might experience a bad performance in the beginning of the recognition, which only can be improved after a few sentences. These sentences have to be stored until the adaptation is estimated, after which only the adaptation matrix (which only is a few kilobytes) needs to be retained.

Compared to the SD trained model this is a substantial improvement. For the training of an SD model hundreds of sentences are needed, the training takes longer and the storage needs per speaker are megabytes of audio in training phase and hundreds of kilobytes per model.

3.3 Accent Adaptation

Besides techniques for adapting the speech recognition system to a specific target speaker, an ASR system can also be adapted to be suitable for recognizing specific accents or dialects. The most basic solution for recognizing accented speech is to train with either exclusively accented speech or to at least include some accented data for training. The problem with the former is that enough accented data (from the same accent) must be available to train a robust model. Both of these methods have the issue that they need training for every accent that will be encountered in recognition. Often, in real world systems, not every accent can be accounted for in advance.

As with the more generic problem with recognizing speakers whose speech was not included in the training data, adaptation can be used to optimize our model for a particular accent. In contrast to speaker adaptation where some ’standard’ methods such as CMLLR exist, the accent adaptation landscape is more diverse. Dialects can differ in pronunciation, grammar and vocabulary, all these three areas have there own adaptation points. The pronunciation and vocabulary differences can be corrected with dictionary adaptation (C. Huang, Chang, et al. 2000; C. Huang, T. Chen, and Chang 2004) or acoustic model adaptation (Z. Wang, Schultz, and Waibel 2003).

For foreign accents also cross-lingual adaptation is a possibility (Aalburg and Hoege 2004; Bartkova and Jouvet 2007; Karhila and Kurimo 2010; Liu and Fung 2000). Instead of using adaptation data from the same language as the one being recognized speech data from another language, commonly the target speakers mother tongue, is used. As an example, bilingual models can be trained or phonemes can be mapped between different languages.

Clarke and Jurafsky (2006) shows that basic acoustic model adaptations have limitations in how far they can reduce error rates, but still good results can be achieved. Off course for foreign accents the level of proficiency in the language greatly changes the possibilities for recognition improvement (Tomokiyo and Waibel 2003).

Chapter 4

Stacked Transformations

Stacked Transformation is a method that uses multiple different sequential CMLLR adaptations to improve recognition. In first instance it is applied to foreign accented speech, but the method could be transferred to normal recognition and to noise robust recognition as well. The method is also described in Smit and Kurimo (2011) which is a conference paper describing the highlights of this method.

The basic idea is inspired by the fact that an important factor for a bad performance when recognizing foreign accented speech is the mismatch between the acoustic models which are trained on non-accented speech and the real acoustic properties of the foreign accented speech that is recognized. Naturally, it would be possible to train a separate model for recognizing the foreign accented speech, resulting in no, or very little, mismatch.

Unfortunately, big corpora of foreign accented speech are scarce. Often the amount of utterances or the number of speakers is limited, which makes it hard to build a model for the foreign accented speech that is as robust as it could be for native speech. In addition, as model training and storage cost a lot of resources, it would be nice if both native and foreign speakers could be recognized with the same models.

The situation is similar to the need for well performing speaker dependent models as described in Chapter 3.1. To obtain a good SD model, already a good approach exists, like the adaptation of SI models into SD models. Similarly we can use the same techniques to transform an SI model into a Speaker-Independent–Accent-Dependent model. Adaptations need much less data than model training and preserve the robustness of the original model.

Besides Speaker Adaptations and Accent adaptations, other adaptations could be thought out that refine the acoustic model parameters to fit a specific group of speakers, such as speakers with the same gender, or speakers with the same age. Adaptations can even be trained without predefined knowledge about the similarity between speakers, but instead the adaptation groups can be determined automatically.

Having different types of adaptations raises the interesting possibility of combining the adaptations for recognition. Is it useful to use first an accent adaptation and after that still a speaker adaptation? These stacks of transformations will be called "Stacked Transformations".

The following sections will describe the new type of adaptations, the accent adaptation and the near neighbour adaptation and the theory behind stacking these transformation to achieve an even better recognition result.

4.1 Accent Transformations

The first step in creating models that could recognize foreign accented speech well, is to gather some data of people with the same foreign accent. With this data a Speaker-Independent–Accent-Independent (SI-AI) model can be adapted into an Speaker-Independent–Accent-Dependent (SI-AD) model. Because the utterances will contain some common features of the accent, the model will be able to recognize this accent much better. Yet, because multiple different speaker are used, the model will still be able to recognize different speakers even the ones that are not first-timers.

The transformation can be the same as for Speaker Adaptation. A difference is that the transformation can have much more adaptation matrices than the normal amount used for Speaker Adaptation, because of more available data.

In normal CMLLR transformations with Regression Class Trees it is common to have a maximum of tens of adaptation matrices, as not often more data is available to train more. With an accented speech corpora, hundreds of robust adaptations can be trained, making the transformation for each group of Gaussians very detailed and specific.

4.2 Near Neigbour Transformations

Besides using a preselected group (such as people with the same accent, or possibly gender), the closest speakers in a corpus could be used for transforming the model. Obviously a problem of how to determine which speakers are similar to the one that needs to be recognized exists.

As a solution, one method is to determine this with help of the eigenvoice technique, which is described in Section 3.2.2. The idea of using eigenvoices for similarity is not new, as it was previously used in Thyes et al. (2000). An eigenspace can be created with all the data available. The training data can overlap with the speech data used in the training of the model, but also additional data can be used. For all speakers in the training set, the eigenvoice parameters can be estimated with help of Maximum Likelihood Eigen Decomposition (MLED).

The eigenvoice parameters are an obvious choice for determining the similarity between two speakers. For measuring the ‘distance’ between two speakers, the euclidean distance criterion can be used .

The procedure for recognizing a new speaker is as follows. A few utterances of the speaker are recognized (or transcribed) and the transcription is used for determining the eigen parameters of the new speaker with MLED. As MLED can work with very little adaptation data, it can be applied already on only a few utterances. The distance between the eigen parameters of the new speaker and all the training speakers can be easily computed and the speakers with the closest distances are elected as neighbours. Adaptation can be done in the same way as done for speaker and accent adaptations.

In real-life systems, the utterance of the new speaker can immediately be used as neighbour data for other speakers. If later another similar speaker needs to be recognized, the just recognized data can be also a candidate for usage in the neighbour transformation.

4.3 Stacked Transformations

Multiple CMLLR transformations can be applied to the same model. Commonly, this is already done with global and tree transformations. The idea of stacked transformations is to use different adaptation levels, all with different detail and using different data sources for the adaptation.

In the case of combining Accent Transformations and speaker adaptations, first a transformation with a big number of adaptation matrices (hundreds) is done to transform the SI-AI model into a SI-AD model. The second transformation will be less detailed and transform the SI-AD model to a Speaker-Dependent (SD) model. The difference between a direct adaptation from an SI to an SD model and the stacked transformation approach is that the knowledge of the characteristics of the accented data is used for both reducing the mismatch and improving the transcription for the Speaker Adaptation. The Stacked Transformation procedure is visualized in Figure 4.1.

Note that the different transformations must have a different level of detail. To recall from Section 3.2.1, a single transformation of a Gaussian is defined by:

$$\hat{\mu} = \mathbf{A}\mu + \mathbf{b} \tag{4.1}$$

and

$$\hat{\Sigma} = \mathbf{B}^T \mathbf{H} \mathbf{B} \tag{4.2}$$

Normally, when two linear transformations are applied, they can be combined to one linear transformation. In that case, the effects of the first transform would be very limited and its effect almost none.

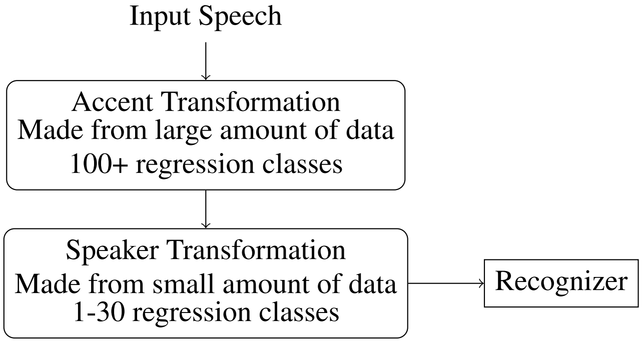

Figure 4.1: Diagram of the recognition procedure with stacked transformations. The accent transformation is trained with data from multiple speakers with the same accent speakers. The speaker transformation is trained with only data of the target speaker.

4.4 Stacked Transformation computation procedures

In comparison to the scenario described in Section 3.2.4 this section explains how Stacked Transformations could be applied in real life scenarios.

A first distinction that needs to be made is between ‘off-line’ and ‘on-line’ computations. Off-line computations are those that are performed before the data of the target speaker is known or even recorded. In normal recognition with Speaker Independent models, this is only the training of the model. All computations that are done involving the speech of our target speaker are on-line computations.

As we assume that a user would like to have as fast as possible results after inputting the recognition data to the system, the on-line computations need to be reduced to as small amount as possible. On the other hand, for off-line computations no users are waiting for the result, meaning that speed is of less importance.

In unsupervised speaker adaptation there are three steps that need to be performed on-line; recognition of the utterance to provide an initial transcription, estimating a transformation with the initial transcription, and after that recognition using adaptation to get the final transcription.

Even though in stacked transformations more adaptations are used, the same number of steps are needed in the on-line phase as for normal speaker adaptation. An Accent Transformation can be completely calculated off-line before any speaker specific data is known. For a Neighbour Transformation all necessary statistics for making the adaptation can be precomputed, leaving only the discovery of neighbours and the aggregation of statistics for the on-line phase. For the Speaker Transformation all the same steps as outlined in the previous paragraph need to be performed.

4.4.1 Storage requirements

The storage needs for adaptation matrices is only a fraction of the space needed to store a complete ASR model. A typical model used in this work is around 40 megabytes in space and a complete transformation only a few kilobytes.

Again there is only a small need for more storage compared to normal adaptation. In normal adaptation a new transformation is stored for every speaker. In Stacked Transformations only one detailed transformation is stored extra for each accent that is occurred.

Chapter 5

Experiments

5.1 Datasets and model setup

5.1.1 Speech Corpora

Multiple English Speech Corpora were utilized in the experiments, as shown in Table 5.1. Firstly, the standard Wall Street Journal (WSJ) dataset was used (Paul and Baker 1992). This dataset is used for standard large vocabulary recognitions. All speakers in the WSJ dataset have an American accent. WSJ0 contains 84 training speakers and a evaluation set, the resulting training set is called WSJ-84.

The second corpus is Wall Street Journal Cambridge (WSJCAM0) (Robinson, Fransen, et al. 1995) a corpus designed to mimic the conditions and size of the WSJ0 dataset, only the speakers have a British English accent instead of an American English accent.

Moreover, multiple corpora with non native dialects were used. The UED_F and UED_G datasets (Wester 2010) and the UED_M (Wester and Liang 2011) contain speech of persons with respectively Finnish, German or Mandarin as their mother tongue. All these datasets were recorded with the purpose of accent evaluations in the context of the EMIME project. Besides the accented English speech the corpora contains parallel recordings in the subject's native tongue. These recordings however, were not used for the experiments.

The last dataset was recorded at Aalto University for the course Digital Signal Processing (DSP) in 2010 and 2011. Only native Finnish persons recorded the sentences. The proficiency in English is lower compared with the UED_F dataset and the background noise levels were higher, hence making it a more challenging dataset to recognize. The combination of the UED_F and the DSP dataset was used to show the effects of the experiments when slightly different data is used for training and evaluation.

		Training			*Evaluation*	
Dataset	Accent	# spkr	Utt. / spkr	# hours	# spkr	Utt. / spkr
WSJ-84	North American	84	85	15	8	41
WSJCAM0	British	92	90	16.4	14	39
UED_F	Finnish	14	115	1.8	14*	30
UED_G	German	14	115	1.8	14*	30
UED_M	Mandarin	14	115	1.8	14*	30
DSP	Finnish	120	22		34	25

Table 5.1: Speech corpora, with the number of speakers, number of utterances per speaker and total hours of speech for training and evaluation. *The UED datasets share the speakers for evaluation and training. Cross-validation is used in experiments that use these datasets in both training and evaluation.

5.1.2 Lexicon

A combination of two different lexicons were used for all experiments. The CMU dictionary (Carnegie Mellon University 2007) is the standard dictionary used with the WSJ speech corpora, containing American English transcriptions. The BEEP dictionary (Robinson 1996) is the equivalent for the WSJCAM corpus. The phoneme set used in these dictionaries is described in Section 2.3.1 and shown in Table 2.1.

The combination of both lexicons is used to enable pronunciations for both native accents being available when recognizing foreign accented speech. Also having the same combined lexicon in all experiments enables fair comparison of the adaptation methods. A last reason for using the combined lexicon is that not all words used in training and evaluation are present in either lexicon by itself. The combined lexicon does contain all words, enabling evaluation without 'out of vocabulary' (OOV) words.

5.1.3 Language Model

The language model utilized in the experiments was the WSJ0 20k words language model. Normally the language model contains a probability for words that are not present in the model, but as the recognition tools do not exploit the probability and all words of the evaluation sets are present in the model, the model is modified to exclude the 'unknown word' probability. As the evaluation sentences are all contained in this language model, OOV words are not playing any role in recognition performance.

5.1.4 Acoustic Models

The different models were all trained with the same procedure, utilizing the tools from the HTK toolkit (Young and Evermann 1997). The models were initialized with the 'flatstart' procedure, where every model is initialized with the same data. The models were started as single phone three-state models, then after multiple re-estimation steps the models were expanded to triphone models and similar Gaussians were tied together. Again after a number of re-estimation steps, the emission distributions were split into Gaussian Mixture Models. The GMM's were not split evenly, the number of mixtures was made proportional to the amount of training data available for that particular GMM. At the end of the training, every model had on average 16 Gaussians per mixture.

5.2 Experiment using accented data in training

One of the methods to improve recognition of accented speech is to include some accented data in training. This experiment combined the basic datasets (WSJ-84 and WSJCAM) with the accented datasets, to see the improvement in recognition result. The results were a baseline for the comparison of the different methods that utilize adaptations. Also the normal WSJ and WSJCAM test sets are evaluated to show the possible degradation in performance for non accented speech.

Because data that is used in training should not be in evaluation, cross-validation was applied for the sets containing UED data. For every speaker, a model was trained of the all speech data, except for it's own. The cross-validated model was used for evaluation. The normal procedure of splitting the data in an training and evaluation set was not possible because of the small number of speakers.

5.2.1 Results

The results for this experiment are shown in 5.2. The variation in results shows that matching training data matters a lot when doing recognition.

First the results for the basic WSJ and WSJCAM evaluation sets are compared. With exactly matching training data they perform well, both around 10% (9.1% for WSJ-84 / WSJ and 10.8% for WSJCAM / WSJCAM). It is no surprise that swapping the models gives much worse results, even tripling the number of errors when WSJCAM is recognized with a WSJ-84 model.

Mixing in accented data has different effects on the WSJ and the WSJCAM evaluations. For WSJ the result improves most of the time slightly, which could be explained by

Model	UED-F	DSP	WSJ-84	WSJCAM
WSJ-84	40.6 / 29.4	50.8 / 43.2	9.1 / 6.3	30.6 / 23.6
WSJ-84+UED_F	33.9 / 25.2	45.2 / 38.3	8.4 / 6.3	24.4 / 19.9
WSJ-84+DSP	34.0 / 24.2	37.4 / 32.5	9.3 / 7.8	25.1 / 19.9
WSJCAM	32.1 / 24.1	46.7 / 39.4	22.9 / 17.2	10.8 / 9.8
WSJCAM+UED_F	26.5 / 21.5	42.8 / 36.3	19.9 / 14.3	11.3 / 9.6
WSJCAM+DSP	26.1 / 20.4	36.7 / 32.9	19.4 / 14.6	11.2 / 9.5

(a) Finnish Accented

Model	UED-G	WSJ-84	WSJCAM
WSJ-84	31.9 / 22.5	9.1 / 6.3	30.6 / 23.6
WSJ-84+UED_G	25.7 / 17.6	8.1 / 6.2	22.6 / 18.4
WSJCAM	22.9 / 16.8	22.9 / 17.2	10.8 / 9.8
WSJCAM+UED_G	19.4 / 14.6	20.2 / 14.4	10.5 / 9.1

(b) German Accented

Model	UED-M	WSJ-84	WSJCAM
WSJ-84	44.8 / 34.9	9.1 / 6.3	30.6 / 23.6
WSJ-84+UED_M	32.6 / 25.4	8.9 / 7.2	25.7 / 20.8
WSJCAM	44.3 / 34.2	22.9 / 17.2	10.8 / 9.8
WSJCAM+UED_M	29.6 / 24.3	20.4 / 15.4	11.4 / 9.9

(c) Mandarin Accented

Table 5.2: Results for using Accented Speech in training. All results are given as 'no adaptation'/'speaker adapted (CMLLR)'. (%WER)

the improved robustness of the model, due to the extra variation present in the training data. With the DSP data added to the WSJ model, the error rate is a bit higher. DSP has more noise than the UED and WSJ sets, so the extra variation in data does not seem to be beneficial. The results are however so close to each other that there is no statistical significant difference between them.

For WSJCAM it has a degrading effect when accented data is mixed with the WSJCAM model, and an improving effect with a WSJ-84 model. A logical explanation is that the WSJCAM model fits so well that extra variation in the training data can only make things worse, and the WSJ-84 model fits badly, so extra variation in training data is beneficial.

The Mandarin accented speech is recognized equally poor with both the plain WSJ-84 model as the WSJCAM model. Adding accented data in the training gives almost 40% improvement when compared to the plain models.

The German and the Finnish accented speech are clearly better recognized with the WSJCAM models. The improvement in adding accented data is relatively smaller than for Mandarin, but still significant. Especially with the WSJCAM models that already performed good, the improvement was smaller.

As there are two datasets with Finnish accented speech with a bit different conditions (worse quality / more noise in DSP) the effects of using one in training and the other for evaluation is studied. As expected, the more similar conditions, the better results. Still, even though the conditions mismatch, it is still beneficial to add the accented data in training, giving good improvement compared to native English models.

Overall, this experiment behaved as expected with no exceptional results.

5.3 Accent Transformation experiment

Instead of adding accented speech in the training, Accent Transformations can be used as described in Section 4.1.

For this experiment a model was trained with both the WSJ-84 and the WSJCAM dataset (WSJ-84+WSJCAM model). For all the three accents two types of adaptations are researched. An adaptation with all data of the accent (excluding the data of the target speaker) and an adaptation with only the data of the same gender. Different numbers of adaptation matrices were experimented, controlled by the size of the Regression Class Tree. The number of nodes is the maximum of adaptation matrices that can be used, and for higher numbers it is likely that only a smaller amount of transformations are really used, because of a lack of data to train all of the nodes in the tree.

5.3.1 Results

In Figure 5.1 the results of the experiment are shown. For all different accents the pattern is approximately the same. The results are greatly improving for the first adaptation matrices, and after that steadily improving until around 250 matrices. After that only very small improvements are seen. As expected, the errors are decreasing and never increasing.

Interesting is that there are little differences between the gender specific and the transforms with both genders. The biggest differences are with a small number of transformations but it is not consistent between accents which of the two options performs better. Therefore, in the rest of the experiments transformations were used with data from both genders.

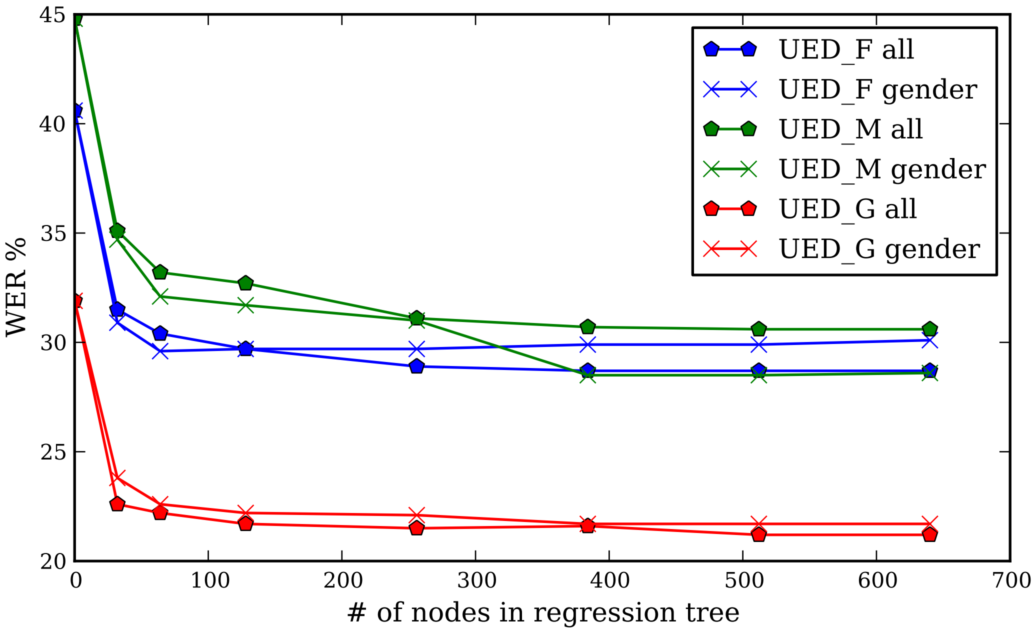

Figure 5.1: Adaptation development by varying the amount of data used for an Accent Transformation. 'gender' transforms only use data of speakers with the same gender and accent as the target speaker.

5.4 Experiment to visually map gender and accents with eigenvoice parameters

The proposed neighbour transform (Section 4.2) uses eigenvoice parameters to find similar speakers in corpora. As the corpora are annotated by accent and gender it is interesting to see whether eigenvoice parameters really separate different groups in a good way. To show the eigenvoice parameters visually, a single model was trained with all training data from the WSJ-84, WSJCAM, UED_F, UED_G and UED_M corpora. After that, MLED was applied and the first two components of the calculated scores were plotted. The different accents are given different shapes and the genders are discriminated by filling the shapes of female speakers and leaving the shapes of male speakers blank.

5.4.1 Results

The generated map is shown in Figure 5.2. The first eigenvoice parameter clearly identifies the gender of the person. All filled markers (the female speakers) are clustered on the left and all open markers (the male speakers) are clustered on the right.

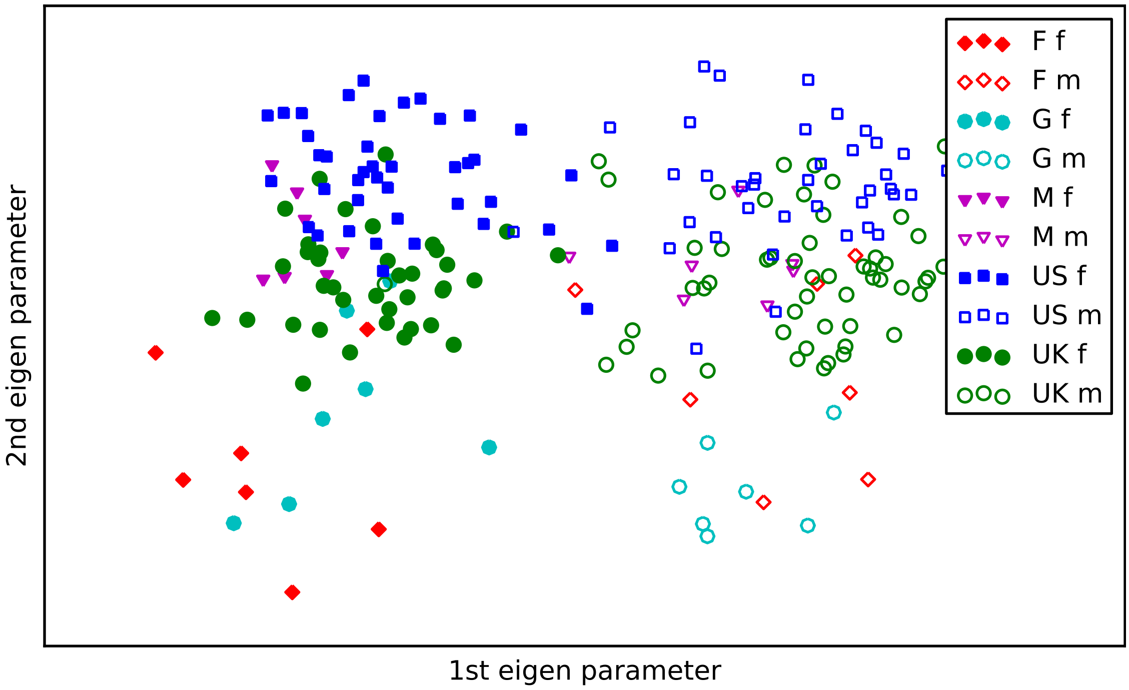

Figure 5.2: Map of voices. Different shapes are used for different corpora (German, Mandarin, Finnish and native corpora WSJ-US and WSJ-UK). The shapes are open for male and filled for female speakers.

Between different accents there is a less clear distinction than between the genders. Still clusterings of datasets are visible, especially for the native accents (UK and US). The other accents are a bit more spread but still close to each others.

Even though the clustering is far from perfect, there seems to be a real relation between eigenvoice parameters and the gender and accent properties of the data. Furthermore, the components of the eigenvoice parameters that are not shown can help in the separation.

In conclusion, whether the eigenvoice parameters are a good similarity measure is of course to be seen in the real recognition experiments but visual inspection looks hopeful.

5.5 Neighbour Transformations experiment

Where the results in Experiment 5.4 showed that the eigenvoice parameters look promising for identifying similar speakers, in this experiment it was tested whether an adaptation with neighbour data is giving better recognition results.

For each combination of model and evaluation set an eigenspace was built with the training data. After that, for every speaker in the training and evaluation sets, the eigenvoice parameters were calculated with MLED. With the eigenvoice parameters, the distances between each evaluation speaker and all the other speakers were calculated to find the closest neighbours. From the selected neighbours, all data was used to estimate a normal CMLLR (Regression Class Tree) transformation.

To reduce the computations needed for the experiment, the target speaker was used in the construction of the eigenspace. Normally, using evaluation data in training can influence the results, but in this case it was only used for defining a base, with an extra step for obtaining the real eigenvoice parameters with only the evaluation data.

There were a lot of parameters that can be tweaked for this experiment. It was chosen to use five eigenvoice parameters and 15 files per speaker to calculate the MLED parameters. As the current focus was primarily on the real adaptation, tweaking the parameters for the similarity measure was left for future research.

Different amounts of neighbours were tried out and compared with normal unsupervised CMLLR adaptation and baseline results.

5.5.1 Results

FF1	FF3	FF5	FF6	FF4	FF2	FF7	20h	209	206	20t
FF2	20t	FF5	01k	20a	FF3	40e	01j	20h	20p	20e
FF3	FF5	FF1	FF4	FF6	FF2	FF7	20h	01k	20p	206
FF4	FF1	FF3	FF6	FF5	FM2	FF7	FM3	206	204	FM1
FF5	FF3	FF1	FF4	FF7	FF6	FF2	20h	01k	206	40p
FF6	FF1	FF4	FF5	FF3	FF2	FM3	FF7	FM2	20t	01k
FF7	20p	206	01k	40p	FF5	011	016	02c	01f	404
FM1	FM3	FM2	FM6	FM5	207	20c	FM7	02b	40a	029
FM2	FM4	20l	FM5	02b	20o	208	20m	FM3	FM6	20v
FM3	FM2	FM1	FM5	FM6	20l	20c	02b	40a	20g	207
FM4	20l	40d	40c	40p	40l	20a	409	208	20i	01y
FM5	20l	20g	40a	015	20m	208	02b	408	FM2	20c
FM6	02b	FM7	FM1	207	20c	029	40n	FM2	FM5	01r
FM7	029	FM6	20c	405	207	40n	40a	02b	01r	403

Table 5.3: An example for the selected neighbours in case of the WSJ-84 model and the UED_F evaluation set. The neighbours are sorted on their distance to the target speaker. The speakers starting with F are Finnish accented, and their second letter is their gender (Male/Female). All other speakers are from WSJ-84

The first result is of the similarity algorithm. Table 5.3 shows the selected neighbours for the Finnish Accented (UED_F) evaluation speakers, for a model trained with only WSJ-84 training data. The possible neighbours were the training speakers from WSJ-84 and all UED_F evaluation speakers (except itself). The results show that often speakers from the same dataset are chosen as close speakers. Nevertheless, there are notable exceptions, for example Finnish female 2 and 7 have primarily American English speakers as neighbours, with the same being true for multiple Finnish males.

It is interesting to compare this list to a listening test determining the degree of foreign accent of the speakers in Wester (2010). In that experiment, the male speakers FM2 and FM5 have highest degree of foreign accent and FM4 the least. Compared to the neighbour results, FM4 has almost all its neighbours from the American English dataset, confirming that he has a low degree of foreign accent. For the other speakers there seems to be no correlation.

			Num neighbours							
UED_F	BL	Adap	1	3	5	7	9	11	13	15
WSJ-84	40.5	29.4	36.5	32.0	32.0	31.6	31.0	31.3	31.5	**30.5**
WSJCAM	32.1	24.1	30.9	28.0	26.7	26.2	26.0	26.8	**25.9**	**25.9**
WSJ-84 + WSJCAM	27.4	19.2	27.9	25.4	23.9	23.9	23.4	23.1	22.7	**22.4**

			Num neighbours							
UED_G	BL	Adap	1	3	5	7	9	11	13	15
WSJ-84	31.9	22.5	29.6	23.7	23.1	23.2	23.3	23.2	23.3	**23.1**
WSJCAM	22.9	16.8	23.0	19.7	**19.1**	**19.1**	19.8	20.2	19.5	19.2
WSJ-84 + WSJCAM	20.2	12.8	20.5	16.4	16.2	15.6	15.6	**15.2**	15.9	15.7

			Num neighbours							
UED_M	BL	Adap	1	3	5	7	9	11	13	15
WSJ-84	44.8	34.9	41.7	36.0	35.0	**34.4**	**34.4**	**34.4**	34.9	**34.4**
WSJCAM	44.3	34.2	39.6	34.3	**32.9**	33.1	**32.9**	33.1	33.6	33.2
WSJ-84 + WSJCAM	34.0	28.1	35.6	30.7	30.1	30.2	29.9	**29.3**	29.8	**29.3**

			Num neighbours							
WSJ0	BL	Adap	1	3	5	7	9	11	13	15
WSJ-84	9.1	6.3	8.8	7.7	7.8	7.5	7.7	7.7	7.7	**7.3**

			Num neighbours							
WSJCAM	BL	Adap	1	3	5	7	9	11	13	15
WSJCAM	10.8	9.8	11.6	11.0	11.0	10.7	**10.6**	**10.6**	10.9	10.7

Table 5.4: Recognition results for using Neighbour Transformations with different amounts of neighbours. BL is the baseline result, recognition without any adaptation. Adap means normal Speaker Adaptation with all evaluation utterances (30 for UED sets, ~40 for WSJ sets) as unsupervised CMLLR Regression Class Tree adaptation. Each table is a different evaluation set, which is written in the left top corner. The best results for each model/evaluation set combination are **bolded** (%WER)

The second set of results is the one of recognition with Neighbour Transformations, shown in Table 5.4. Already from one neighbour on, neighbour adaptation gives improvements compared to the baseline. The error rates drop when more neighbours are added. Depending on the dataset the most gain in accuracy is achieved with 5 to 15 neighbours.

Compared to normal adaptation results, there is often relatively small differences to neighbour adaptation. The general trend is however that normal adaptation performs better, with an exception for the WSJ-84 and the WSJCAM model in combination with the UED_M evaluation set. Looking to the baseline results it can seen that these models are the 'worst' and are therefore likely to gain the most from the neighbour transformation.

Model-wise and accent-wise there are not much differences in the behavior of neighbour transforms, indicating that this is a method that can be generalized easily.

For recognition of native evaluation data there is less to gain with neighbour adaptations. For the WSJ-84 / WSJ combination, neighbour transformations improve the baseline, but can not compete with normal adaptation. The WSJCAM / WSJCAM combination gives even some neighbour transformation results that are worse than the baseline result. This shows that by itself neighbour transformations are not very useful when the conditions between training and evaluation are matching. When this is not the case they can help the performance almost as much as normal adaptations could.

5.6 Stacked Transformations experiment

To evaluate the use of stacked transformations, multiple models were trained and the effect of a normal CMLLR transform, an accent transform and a combination of both were tested.

For each foreign accent, two models were tested. The baseline result and CMLLR adaptation using 5 and 30 adaptation sentences were reported for comparison. For both neighbour transformations and accent transformations three experiments were done; no speaker adaptation and CMLLR speaker adaptation with 5 and 30 adaptation sentences. Chosen was to use 5 neighbours in all experiments, as Section 5.5 showed that this gives good results for all evaluation sets and is the best for some.

The native English sets were evaluated only for normal adaptation and neighbour adaptation, as accent adaptation does not exist for them.

5.6.1 Results

In Table 5.5 the results for this experiment are displayed. Clearly can be seen that Stacked Transformations yield good results for foreign accented datasets. For every combination of model and evaluation set the error rates for S5 (Speaker adaptation with 5 sentences)

Eval	Model	BL	S5	S30	N	N S5	N S30	A	A S5	A S30
UED_F	WSJ-84	40.6	31.2	29.4	32.0	28.5	28.1	29.0	26.7	26.4
UED_F	WSJCAM	32.1	25.7	24.1	26.7	24.5	23.8	26.4	23.8	23.8
UED_G	WSJ-84	31.9	23.8	22.5	23.1	21.4	19.6	21.5	20.2	19.1
UED_G	WSJCAM	22.9	17.9	16.8	19.1	17.5	16.9	19.4	17.5	17.0
UED_M	WSJ-84	44.8	36.8	34.9	35.0	32.4	31.0	33.0	31.0	29.6
UED_M	WSJCAM	44.3	36.6	34.2	32.9	29.9	29.7	33.2	30.0	30.3
WSJ	WSJ-84	9.1	6.4	6.2	7.8	6.2	6.3			
WSJCAM	WSJCAM	10.8	10.0	9.6	11.0	10.1	9.6			

Table 5.5: Results for Speaker adaptation (S, followed by number of adaptation sentences), Neighbour adaptation (N), Accent adaptation (A). Combinations of these are stacked transformations (%WER)

are higher than N S5 (Neighbour adaptation + Speaker adaptation with 5 sentences) and A S5 (Accent Adaptation + Speaker Adaptation). Also with 30 sentences adaptation, the Stacked Transformations are the best, except for UED_G with the WSJCAM model where the results do not differ.

For the WSJ-84 / WSJ and WSJCAM / WSJCAM experiment, the Neighbour Transformations give marginal better results than normal adaptation. The results are not significant, but very promising, considered that the neighbour selection algorithm has not been optimized.

Statistical Significance

The result in Table 5.5 are giving a good indication of performance, but it is important whether there are statistically significant differences between adaptation methods. The Wilcoxon significance test (as explained in Section 2.6.4) was applied for the combinations of all different adaptation methods. The results are shown in Table 5.6.

It is clear that not all differences were actually significant. If we discuss WSJ-84, it turns out that Stacked Transformations of an Accent Transformation and a Speaker Transformation performs always significantly better than only Speaker Adaptation, even when only 5 speaker utterances are used for the Stacked Transformation and 30 speaker utterances for the Speaker Adaptation. With WSJCAM however, this is only the case for Mandarin accented speech.

When a Neighbour Transformation is used, it is always beneficial to use Speaker Adaptation. Despite that, not often the results are significantly better than normal Speaker Adaptation. For the WSJCAM models this is only the case for Mandarin accented speech.

The effectiveness of stacked transformations is clearly dependent on the original model.

	S5	S30	N5	N S5	N S30	A	A S5	A S30
B	F,G,M	F,G,M	F,G,M	F,G,M	F,G,M	F,G,M	F,G,M	F,G,M
S5		F,G	-	G,M	F,G,M	-	F,G,M	F,G,M
S30			-	M	G,M	-	F,G,M	F,G,M
N5				F,G,M	F,G,M	*F*	F,G,M	F,G,M
N5 S5					G	-	-	-
N5 S30						*G*	-	-
A							F,M	F,G,M
A S5								-

(a) WSJ-84

	S5	S30	N5	N S5	N S30	A	A S5	A S30
B	F,G,M	F,G,M	F,G,M	F,G,M	F,G,M	F,M	F,G,M	F,G,M
S5		G	M	M	F,M	M	F,M	F,G
S30			*F,G*	M	M	*F*	M	M
N5				F,G,M	F,G,M	-	F,G,M	F,G
N5 S5					F	*F,M*	-	-
N5 S30						*F,M*	-	-
A							F,M	F,M
A S5								-

(b) WSJCAM

Table 5.6: Statistical significance test matrix for different adaptation methods. If the adaptation technique in a column is significantly better than the technique in the row, the accent is mention in normal font. Significantly worse results are mentioned in *italic*. All other results do not give any statistical significant difference.

When there is a big mismatch, for example with the plain WSJ0 models, there is always a gain in doing an accent adaptation first. When the models are already reasonably good, for example with the WSJCAM+UED models, there is little to no improvement in using both transformations. As expected, when the model is a different dataset than the target dataset, it is always beneficial to do an accent transform, compared to baseline model.

5.7 Experiment of varying the amount of Speaker Adaptation data

The previous experiment only showed the results when using Stacked Transformation with 5 or 30 utterances. In this experiment the development of the error rate was studied when different amounts of utterances are used for normal adaptation and Stacked Transformations.

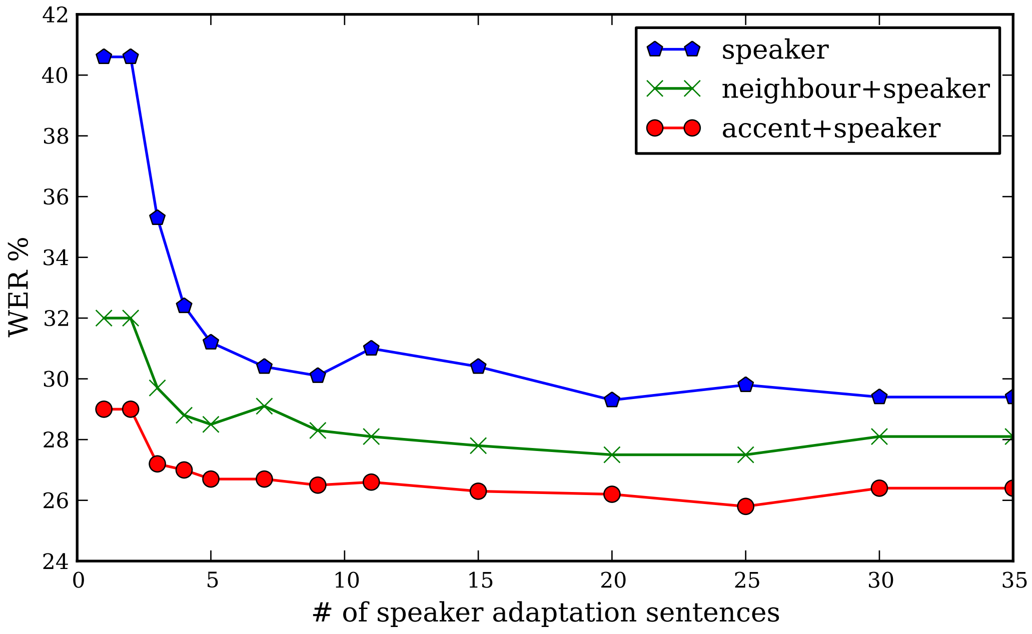

Figure 5.3: Adaptation development for the UED_F evaluation set and the WSJ-84 model

In Figure 5.3 the error developments for the WSJ-84 model and the UED_F evaluation set are shown. It shows clearly that the Stacked Transformations (both with Accent or Neighbour Transformation) need less adaptation utterances to improve the result. For normal adaptation the error rate improves for the first 10 adaptation utterances, for Stacked Transformations the error already converges after 5 adaptation utterances.

In the long term the Stacked Transformation works better than normal Speaker Adaptation. The Stacked Transformation with Accent Transformation works the best.

5.8 Comparison of Accent in Training and Stacked Transformations

Interesting is whether the Stacked Transformations performs better than when accented data is used in training. Using the results of previous experiments, specifically the results in Table 5.2 and Table 5.5, the two methods can be compared.

It is clear that using accented data in training works almost always better. The performance gains achieved with using accented data in training are so big, that stacked transformations can not match up to the improvement, neither with accent nor with neighbour transformations. Noteworthy is however, that the baseline results of the accented speech models are most of the time equaled by the stacked transformations and in some cases even equaled by the plain accent and neighbour transformations.

5.8.1 Combining both methods

As with most methods in ASR, these methods do not have to be exclusive. Stacked Transformations could be applied on models that already contain accented speech. Therefore, normal adaptation and stacked transformations were applied to the models used in Section 5.2. Cross validation was again used to prevent that the target speaker would be present in model training.

Eval	Model	BL	S5	S30	A	A S5	A S30
UED_F	WSJ-84+UED	33.9	26.4	25.2	**27.5**	**24.7**	**23.2**
UED_F	WSJCAM+UED	26.5	23.2	21.5	**24.0**	22.5	21.4
UED_G	WSJ-84+UED	25.7	19.5	17.6	**20.4**	18.3	17.4
UED_G	WSJCAM+UED	19.4	15.6	14.6	18.4	16.2	15.4
UED_M	WSJ-84+UED	32.6	27.7	25.4	**27.8**	25.7	23.9
UED_M	WSJCAM+UED	29.6	26.6	24.3	28.0	25.8	24.9

Table 5.7: Results for Speaker adaptation (S, followed by number of adaptation sentences), Accent adaptation (A) and Stacked Transformations (A S). The results in a row are compared with the result with the same amount of speaker adaptation (BL and A, S5 and A S5, S30 and A S30) and the statistically significant better results are **bolded** (%WER)

The results for the combination of accented training and Stacked Transformations are shown in Table 5.7 . The gains achieved for Stacked Transformations are similar to the experiments with native English models. The accent transformation gives always an improvement over the baseline, even though the differences are a bit smaller. For the models with WSJ-84 and the WSJCAM-UED_F model the results are even significantly better according to the Wilcoxon significance test. Compared with normal speaker adaption (5 sentences), stacked transformations with 5 speaker sentences perform better, except for the WSJCAM+UED_G model recognizing UED_G. For the experiments with 30 speaker adaptation sentences also the WSJCAM+UED_M stacked transformations are performing worse than normal speaker adaptation. A reason for this is not evident, but the results differ little and are not significantly different.

For these models many of the reasons to use Stacked Transformations, low on-line costs and no need of retraining, are not applicable anymore. This experiment showed that Stacked Transformations give better results for most accents and can still be useful in the scenario of models that are trained with accented data.

Chapter 6

Conclusions

This thesis has introduced various methods of adaptation, focused on the application of improving the recognition for foreign accented speech. Besides speaker adaptation, accent and neighbour transformations were introduced. These transformations were also combined, using a technique named stacked transformations.

Accent transformations and neighbour transformations are common Regression Class Tree CMLLR adaptations, estimated with a different set of speech data than that of the target speaker. In case of accent transformations, a whole corpus of speech in the same accent as the target speaker is used. For neighbour transformations a number of similar speakers are selected in an unsupervised manner with help of the eigenvoices technique and their data used for the creation of a neighbour adaptation.

For both Accent and Neighbour Transformations, the big advantage is that a lot of computation can be done in advance of the recognition ('off line'). In case of accent transformation, the complete adaptation can be calculated, leaving only a very small additional load for using the transformation in recognition. For neighbour transformations all possible statistics for adaptation can be calculated in advance, leaving only the similarity detection and the combination of statistics to adaptations for the recognition phase.

Compared to a baseline recognition, both accent and neighbour transformations give big improvements, up to around 25% reduction in Word Error Rate (WER) for recognition of foreign accent speech with a native English model. For neighbour transformations, even a 20% improvement was achieved for recognition of American English on an American English model. When the Accent and Neighbour Transformations are compared to normal speaker adaptation, the results are more diverse. Overall the results are rather similar and often there are no statistically significant differences.

Stacked Transformations combine both an accent or neighbour transform with normal speaker adaptation. The adaptations for the accent and neighbour transform can still be precomputed and they provide a better initial transcription and a model close to the speaker dependent model. The speaker adaptation needs therefore less adaptation sentences to achieve a good speaker dependent model and also when more adaptation is used it outperforms normal speaker adaptation.

Compared to baseline recognition or recognition with only an accent or neighbour transform the error rates always improve when stacked transformations are used. Compared to normal adaptation the results vary greatly between datasets and amounts of adaptation data. Especially the stacked transformations having an accent transformation and a speaker adaptation with only a small amount of adaptation sentences improves the recognition significantly compared to normal speaker adaptation with the same amount of sentences, up to 15%. When both methods use more adaptation sentences, the advantage slowly fades, the results of stacked transformations staying a small bit better than normal adaptation.

Compared to the method of mixing in accented speech in training, stacked transformations do not give better results. The advantage of stacked transformations here is the lower computational cost compared with retraining a complete speech recognition model with new data. Interesting is that when both techniques are used in combination, the error rates can be improved even more.

Future Research

The technique of stacked transformations has not been applied before and therefore allows for a lot of refinement. Especially the type of transformations that can be stacked and the real effect of the intermediate transforms can be researched.

As mentioned in the introduction, Stacked Transformations are developed with a possible application to HMM-based speech synthesis in mind. Stacked Transformations, Accent Transformations and Neighbour Transformations can possibly be applied there for personalizing the output of speech synthesis, without the need of a great amount of adaptation data and without the need to retrain and store big models for each speaker.

Neighbour Transformations were also introduced in this thesis and primarily used in the context of Stacked Transformations. The possibilities for this method are great as they even can be useful for ‘standard’ recognition, when both the model and evaluation data are matching. This is however still future as current results are still 15% worse than normal adaptation.

Bibliography

Aalburg, S. and H. Hoege (2004). “Foreign-accented speaker-independent speech recognition”. In: *Eighth International Conference on Spoken Language Processing*. ISCA.

Atal, B. S. (1974). “Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification”. In: *The Journal of the Acoustical Society of America* 55.6, pp. 1304–1312. DOI: 10.1121/1.1914702.

Bacchiani, M. and B. Roark (Apr. 2003). “Unsupervised language model adaptation”. In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP ’03). 2003 IEEE International Conference on*. Vol. 1, DOI: 10.1109/ICASSP.2003.1198758.

Baker, J. et al. (May 2009). “Developments and directions in speech recognition and understanding, Part 1 [DSP Education]”. In: *Signal Processing Magazine, IEEE* 26.3, pp. 75 –80. ISSN: 1053-5888. DOI: 10.1109/MSP.2009.932166.

Baker, J. et al. (2007). “Historical Development and Future Directions in Speech Recognition and Understanding”. In:

Bartkova, Katarina and Denis Jouvet (2007). “On using units trained on foreign data for improved multiple accent speech recognition”. In: *Speech Communication* 49.10-11. Intrinsic Speech Variations, pp. 836 –846. ISSN: 0167-6393. DOI: DOI:10.1016/j.specom.2006.12.009.

Baum, Leonard E. et al. (1970). “A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains”. English. In: *The Annals of Mathematical Statistics* 41.1, pp. 164–171. ISSN: 00034851. URL: http://www.jstor.org/stable/2239727.

Bilmes, Jeff (1997). *A Gentle Tutorial of the EM algorithm and its application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*. Tech. rep. TR-97-021. ICSI.

Botterweck, H. (2000). “Very Fast Adaptation for Large Vocabulary Continuous Speech Recognition Using Eigenvoices”. In: *Sixth International Conference on Spoken Language Processing*. ISCA.

Carnegie Mellon University, The (2007). *Carnegie Mellon Pronouncing Dictionary (cmudict)*. version 0.7a. URL: http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

Chen, K.T. and H.M. Wang (2001). “Eigenspace-based maximum a posteriori linear regression for rapid speaker adaptation”. In: *IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS SPEECH AND SIGNAL PROCESSING*. Vol. 1. Citeseer.

Clarke, C. and D. Jurafsky (2006). “Limitations of MLLR Adaptation with Spanish-Accented English: An Error Analysis”. In: *Ninth International Conference on Spoken Language Processing*. ISCA.

Compernolle, Dirk Van (1989). “Noise adaptation in a hidden Markov model speech recognition system”. In: *Computer Speech & Language* 3.2, pp. 151 –167. ISSN: 0885-2308. DOI: DOI:10.1016/0885-2308(89)90027-2.

Cooley, J., P. Lewis, and P. Welch (June 1969). “The finite Fourier transform”. In: *Audio and Electroacoustics, IEEE Transactions on* 17.2, pp. 77 –85. ISSN: 0018-9278. DOI: 10.1109/TAU.1969.1162036.

Creutz, Mathias (2006). “Induction of the Morphology of Natural Language: Unsupervised Morpheme Segmentation with Application to Automatic Speech Recognition”. PhD thesis. Helsinki University of Technology.

Davis, KH, R. Biddulph, and S. Balashek (1952). “Automatic Recognition of Spoken Digits”. In: *The Journal of the Acoustical Society of America* 24, p. 637.

Digalakis, V.V., D. Rtischev, and L.G. Neumeyer (Sept. 1995). “Speaker adaptation using constrained estimation of Gaussian mixtures”. In: *Speech and Audio Processing, IEEE Transactions on* 3.5, pp. 357 –366. ISSN: 1063-6676. DOI: 10.1109/89.466659.

Forney G.D., Jr. (Mar. 1973). “The viterbi algorithm”. In: *Proceedings of the IEEE* 61.3, pp. 268 –278. ISSN: 0018-9219. DOI: 10.1109/PROC.1973.9030.

Furui, S. (Feb. 1986). “Speaker-independent isolated word recognition using dynamic features of speech spectrum”. In: *Acoustics, Speech and Signal Processing, IEEE*

Transactions on 34.1, pp. 52 –59. ISSN: 0096-3518. DOI: `10.1109/TASSP.1986.1164788`.

Gales, M.J.F. (1996). *The generation and use of regression class trees for MLLR adaptation*. Tech. rep. Cambridge University Engineering Department.

— (1998). “Maximum likelihood linear transformations for HMM-based speech recognition”. In: *Computer Speech & Language* 12.2, pp. 75 –98. ISSN: 0885-2308. DOI: `DOI:10.1006/csla.1998.0043`.

— (July 2000). “Cluster adaptive training of hidden Markov models”. In: *Speech and Audio Processing, IEEE Transactions on* 8.4, pp. 417 –428. ISSN: 1063-6676. DOI: `10.1109/89.848223`.

Gales, M.J.F. and P.C. Woodland (1996). “Mean and variance adaptation within the MLLR framework”. In: *Computer Speech & Language* 10.4, pp. 249 –264. ISSN: 0885-2308. DOI: `DOI:10.1006/csla.1996.0013`.

Gauvain, J.-L. and Chin-Hui Lee (Apr. 1994). “Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains”. In: *Speech and Audio Processing, IEEE Transactions on* 2.2, pp. 291 –298. ISSN: 1063-6676. DOI: `10.1109/89.279278`.

Gillick, L. and S.J. Cox (May 1989). “Some statistical issues in the comparison of speech recognition algorithms”. In: *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, 532 –535 vol.1. DOI: `10.1109/ICASSP.1989.266481`.

Hirsimäki, Teemu et al. (2006). “Unlimited vocabulary speech recognition with morph language models applied to Finnish”. In: *Computer Speech & Language* 20.4, pp. 515 –541. ISSN: 0885-2308. DOI: `DOI:10.1016/j.csl.2005.07.002`.

Huang, C. et al. (2000). “Accent modeling based on pronunciation dictionary adaptation for large vocabulary Mandarin speech recognition”. In: *Sixth International Conference on Spoken Language Processing*.

Huang, C., T. Chen, and E. Chang (2004). “Accent issues in large vocabulary continuous speech recognition”. In: *International Journal of Speech Technology* 7.2, pp. 141–153. ISSN: 1381-2416.

Huang, Xuedong, Alex Acero, and Hsiao-Wuen Hon (2001). *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. 1st. Upper Saddle River, NJ, USA: Prentice Hall PTR. ISBN: 9780130226167.

Jalanko, M. (1980). "Studies of Learning Projective Methods in Automatic Speech Recognition". PhD thesis. Espoo, Finland: Helsinki University of Technology.

Jolliffe, IT (2002). *Principal component analysis*. Springer Verlag. ISBN: 0387954422.

Karhila, Reima and Mikko Kurimo (Dec. 2010). "Unsupervised cross-lingual speaker adaptation for accented speech recognition". In: *Spoken Language Technology Workshop (SLT), 2010 IEEE*, pp. 109 –114. DOI: 10.1109/SLT.2010.5700831.

Kohavi, R. (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 2*. Morgan Kaufmann Publishers Inc., pp. 1137–1143.

Kohonen, T. "Workstation-based phonetic typewriter". In: *Neural Networks for Signal Processing [1991]., Proceedings of the 1991 IEEE Workshop*. IEEE, pp. 279–288.

Kuhn, Roland et al. (2000). "Rapid speaker adaptation in eigenvoice space". In: *IEEE Transactions on Speech and Audio Processing* 8.6, pp. 695–707. DOI: 10.1109/89.876308.

Kuhn, Roland et al. (1998). "Eigenvoices for speaker adaptation". In: *Fifth International Conference on Spoken Language Processing*. URL: http://www.isca-speech.org/archive/icslp_1998/i98_0303.html.

Kurimo, Mikko (1997). "Using Self-Organizing Maps and Learning Vector Quantization for Mixture Density Hidden Markov Models". PhD thesis. Helsinki University of Technology.

Kurimo, Mikko et al. (2006). "Unlimited vocabulary speech recognition for agglutinative languages". In: *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. HLT-NAACL '06. New York, New York: Association for Computational Linguistics, pp. 487–494. DOI: 10.3115/1220835.1220897.

Ladefoged, P. (1990). "The revised international phonetic alphabet". In: *Language* 66.3, pp. 550–552. ISSN: 0097-8507.

Leggetter, C.J. and P.C. Woodland (1995). “Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models”. In: *Computer speech and language* 9.2, p. 171. ISSN: 0885-2308.

Liu, W.K. and P. Fung (2000). “MLLR-Based Accent Model Adaptation Without Accented Data”. In: *Sixth International Conference on Spoken Language Processing*. ISCA.

Manning, C.D. (1999). *Foundations of statistical natural language processing*. Vol. 59. MIT Press.

Mermelstein, P. and S. Davis (1980). “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 28.4, p. 357.

Nguyen, Patrick, Christian Wellekens, and Jean-Claude Junqua (1999). “Maximum likelihood eigenspace and MLLR for speech recognition in noisy environments”. In: *Sixth European Conference on Speech Communication and Technology*. Citeseer, pp. 2519–2522.

Odell, J.J. (1995). “The Use of Context in Large Vocabulary Speech Recognition”. PhD thesis.

O’Shaughnessy, Douglas (1987). *Speech Communication: Human and Machine*. Addison-Wesley.

Paul, Douglas B. and J. Baker (1992). “The design for the wall street journal-based CSR corpus”. In: *Proceedings of the workshop on Speech and Natural Language*. HLT ’91. Harriman, New York: Association for Computational Linguistics, pp. 357–362. ISBN: 1-55860-272-0. DOI: 10.3115/1075527.1075614.

Rabiner, L.R. (Feb. 1989). “A tutorial on hidden Markov models and selected applications in speech recognition”. In: *Proceedings of the IEEE* 77.2, pp. 257 –286. ISSN: 0018-9219. DOI: 10.1109/5.18626.

Robinson, Tony (1996). *BEEP pronounciation dictionary*. version 1.0. URL: ftp://svr-ftp.eng.cam.ac.uk/pub/comp.speech/dictionaries.

Robinson, Tony et al. (1995). “Wsjcam0: A British English Speech Corpus For Large Vocabulary Continuous Speech Recognition”. In: *In Proc. ICASSP 95*. IEEE, pp. 81–84.

Siivola, V. et al. (2003). “Unlimited vocabulary speech recognition based on morphs discovered in an unsupervised manner”. In: *Proc. Eurospeech’03*. Citeseer, pp. 2293–2296.

Smit, Peter and Mikko Kurimo (May 2011). “Using stacked transformations for recognizing foreign accented speech”. In: *Acoustics Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. Prague, Czech Republic.

Stevens, S.S., J. Volkmann, and EB Newman (1937). “A scale for the measurement of the psychological magnitude pitch”. In: *The Journal of the Acoustical Society of America* 8, p. 185.

Thyes, Olivier et al. (2000). “Speaker identification and verification using eigenvoices”. In: *INTERSPEECH*, pp. 242–245.

Tomokiyo, L.M. and A. Waibel (2003). “Adaptation methods for non-native speech”. In: *Multilingual Speech and Language Processing*, p. 6. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.7.3902.

Torkkola, K. et al. (1991). “Status report of the Finnish phonetic typewriter project”. In:

Wang, N.J.-C. et al. (2001). “Rapid speaker adaptation using a priori knowledge by eigenspace analysis of MLLR parameters”. In: *Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP ’01). 2001 IEEE International Conference on*. Vol. 1, 345 –348 vol.1. DOI: 10.1109/ICASSP.2001.940838.

Wang, Zhirong, T. Schultz, and A. Waibel (Apr. 2003). “Comparison of acoustic model adaptation techniques on non-native speech”. In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP ’03). 2003 IEEE International Conference on*. Vol. 1, DOI: 10.1109/ICASSP.2003.1198837.

Wester, Mirjam (Sept. 2010). *The EMIME Bilingual Database*. Tech. rep. EDI-INF-RR-1388. The University of Edinburgh. URL: http://www.inf.ed.ac.uk/publications/report/1388.html.

Wester, Mirjam and Hui Liang (Feb. 2011). *The EMIME Mandarin Bilingual Database*. Tech. rep. EDI-INF-RR-1396. The University of Edinburgh. URL: http://www.inf.ed.ac.uk/publications/report/1396.html.

Viterbi, A. (Apr. 1967). “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm”. In: *Information Theory, IEEE Transactions on* 13.2, pp. 260 –269. ISSN: 0018-9448. DOI: 10.1109/TIT.1967.1054010.

Woodland, P.C. (2001). “Speaker adaptation for continuous density HMMs: A review”. In: *ISCA Tutorial and Research Workshop (ITRW) on Adaptation Methods for Speech Recognition*. URL: http://www.isca-speech.org/archive/adaptation/adap_011.html.

Young, S. and G. Evermann (1997). *The HTK book*. Vol. 2. Citeseer.

Young, S., J.J. Odell, and P.C. Woodland (1994). “Tree-based state tying for high accuracy acoustic modelling”. In: *Proceedings of the workshop on Human Language Technology*. Association for Computational Linguistics, pp. 307–312. ISBN: 1558603573.