Aalto University
School of Science and Technology
Faculty of Electronics, Communications and Automation
Master's Programme in Electronics and Electrical Engineering

André Mansikkaniemi

Acoustic Model and Language Model Adaptation for a Mobile Dictation Service

Master's thesis

Espoo, 3.2.2010

Thesis supervisor:

Prof. Mikko Sams

Thesis instructor:

D.Sc.(Tech.) Mikko Kurimo

Aalto University School of Science and Technology	ABSTRACT OF MASTER'S THESIS

Author:	André Mansikkaniemi
Title:	Acoustic model and language model adaptation for a mobile dictation service
Date:	3.2.2010 **Language:** English **Pages:** 3+70
Professorship:	Cognitive technology **Code:** S-114
Supervisor:	Prof. Mikko Sams
Instructor:	D.Sc.(Tech.) Mikko Kurimo

Automatic speech recognition is the machine-based method of converting speech to text. MobiDic is a mobile dictation service which uses a server-side speech recognition system to convert speech recorded on a mobile phone to readable and editable text notes.

In this work, performance of the TKK speech recognition system has been evaluated on law-related speech recorded on a mobile phone with the MobiDic client application. There was mismatch betweeen testing and training data in terms of both of acoustics and language. The background acoustic models were trained on speech recorded on PC microphones. The background language models were trained on texts from journals and news wire services. Because of the special nature of the testing data, main focus has been on using acoustic model and language model adaptation methods to enhance speech recognition performance.

Acoustic model adaptation gives the highest and most reliable performance increase. Using the global cMLLR method, word error rate reductions between 15-22% can be reached with only 2 minutes of adaptation data. Regression class cMLLR can give even higher performance boosts if larger sets of audio adaptation data ($>$ 10 min) are available.

Language model adaptation was not able to significantly improve performance in this task. The main problems were differences between language adaptation data and language of the law-related speech data.

Keywords: automatic speech recognition, mobile dictation, acoustic model adaptation, language model adaptation

Aalto-universitetet
Tekniska Högskolan

SAMMANDRAG
AV DIPLOMARBETET

Utfört av:	André Mansikkaniemi		
Arbetets namn:	Adaptering av akustiska modeller och språkmodeller för en mobil dikteringstjänst		
Datum:	3.2.2010	**Språk:** Engelska	**Sidoantal:** 3+70
Professur:	Kognitiv teknologi	**Kod:** S-114	
Övervakare:	Prof. Mikko Sams		
Handledare:	Tekn. dr. Mikko Kurimo		

Automatisk taligenkänning är en maskinstyrd metod genom vilken tal omvandlas till text. MobiDic är en mobil dikteringstjänst som använder ett serverbaserat automatiskt taligenkänningssystem för att omvandla tal inspelat på en mobiltelefon till läsbara och editerbara textdokument.

I detta arbete undersöktes förmågan hos Tekniska Högskolans taligenkänningssystem att omvandla juridik-relaterat tal inspelat på en mobiltelefon med MobiDics klientprogram till korrekt text. Det fanns skillnader mellan test- och träningsdata gällande både akustik och språk. De akutiska bakgrundsmodellerna var tränade med tal som hade spelats in på en datormikrofon. Språkmodellerna var tränade med text från olika tidningar och nyhetstjänster. På grund av testdatans speciella karaktär har tyngdpunkten i arbetet legat på att förbättra taligenkänningsförmågan hos systemet genom adaptering av akustiska modeller och språkmodeller.

Adaptering av akustiska modeller ger de bästa och pålitligaste resultaten i syftet att förbättra taligenkänningsförmågan. Genom att använda den globala cMLLR-metoden och endast 2 minuter av adapteringsdata kan man förminska antalet feltolkade ord med 15-22%. Genom att använda den regressionsklassbaserade cMLLR-metoden kan man uppnå ytterligare förbättringar i taligenkänningsförmågan om det finns större mängder av adapteringsdata ($>$ 10 min.) tillgängligt.

Adaptering av språkmodellen gav ingen betydande förbättring av taligenkänningsförmågan. Det främsta problemet var de stora skillnaderna mellan språkadapteringsdata och språket som förekom i de juridik-relaterade talinspelningarna.

Nyckelord: automatisk taligenkänning, mobil diktering, adaptering av akustiska modeller, adaptering av språkmodeller

Preface

This work was done in the Department of Information and Computer Science of Helsinki University of Technology during 2008-2009. The project has been supported by Mobiter Dicta and Academy of Finland.

I'm very grateful to my instructor Mikko Kurimo for giving me the chance to do my master's thesis in the Speech group and helping me along the way in this project. I also want to thank Mikko Sams for supervising this work.

Special thanks also go out to Janne Pylkkönen and Teemu Hirsimäki in the Speech group for helpful advice about the use and inner workings of the TKK speech recognition system.

Finally, I want to thank my parents, brother and my dear Feifei for support, inspiration and encouragement.

André Mansikkaniemi
Otaniemi, February 3, 2010

Contents

Symbols and abbrevations

$\mathrm{C}(W)$	Number of occurrences of event W
$\mathrm{N}_{1+}(\bullet, W)$	Number of distinct contexts with event W
$\mathrm{P}(X)$	Probability of X
$\mathrm{P}(X\|Y)$	Conditional probability of X given the occurrence of Y

AM	Acoustic model
ASR	Automatic speech recognition
GMM	Gaussian mixture model
HMM	Hidden markov model
LER	Letter error rate
LM	Language model
LVCSR	Large vocabulary speech recognition system
MDI	Minimum discrimination information
MFCC	Mel-frequency cepstrum coefficient
MLLR	Maximum likelihood linear regression
SD	Speaker dependent
SI	Speaker independent
VTLN	Vocal tract length normalization
WER	Word error rate
WSJCAM0	Wall Street Journal Cambridge edition speech corpus

Chapter 1

Introduction

1.1 Automatic Speech Recognition

Automatic speech recognition (ASR) can in general terms be defined as any machine-based method that converts speech to text. Speech recognition as a scientific research field dates back to the 1950's with the development of various types of isolated-word recognizers at research laboratories and universities [1].

Having mainly been a curious academic research topic for many decades, many commercial ASR applications began to see the daylight in the 1990's. Speech recognition technology is today implemented in a variety of different applications, automatic dictation being one of the most used ASR application on personal computers.

The architecture of a modern speech recognition system is shown in Figure 1.1.

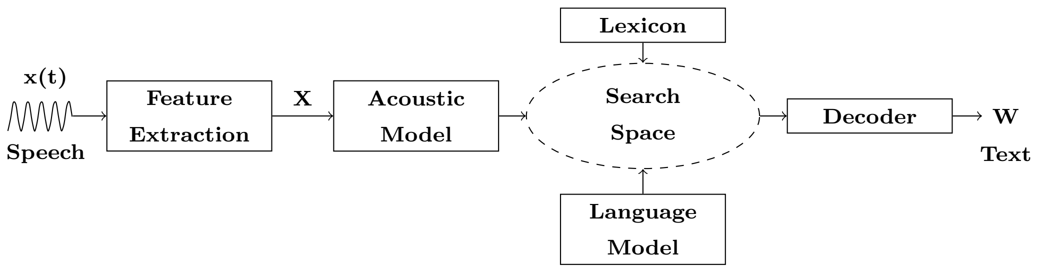

Figure 1.1: The acoustic model, lexicon, and language model define a search space that is used by the decoder to find the most likely word sequence.

Most of today's ASR systems aim to be speaker independent (SI). Features, which are invariant to both the speaker and environment, are extracted from the speech signal *x(t)*. Statistical probability models are used to find the most likely word sequence. The acoustic model defines the probability that a basic sound unit, or phoneme has been uttered. The lexicon contains information of how words are formed from phoneme sequences. The language model defines the probability of the occurrence of a word or a word sequence.

The decoder is the algorithm that tries to find the most likely word sequence $\boldsymbol{W}$ in the search space defined by the statistical models.

1.2 Mobile Dictation

Mobile phones of today have grown to become quite powerful computing devices. The same set of applications, which could only be run on desktop computers in the past, can now be used on many of the latest mobile phone models.

Although the computational power of mobile devices has grown, the small physical size still restricts easy use of many applications. A word processor is able to run smoothly on almost any mobile device but the small size of the input keys makes it quite a painful task to produce any longer documents.

Direct speech input could be an alternative to traditional input mechanisms for mobile devices. MobiDic is an application that has been developed at the University of Tampere [2]. It's a mobile dictation and note-taking application. The application runs on a mobile phone. The user reads in speech to the microphone and gets it back as editable text that can be stored and sent forwards.

MobiDic implements the model of network speech recognition (NSR) [3], shown in Figure 1.2.

The speech is digitally recorded on to the phone and sent to a speech recognition server that converts the speech to text. The text results are then sent back to the user for editing and reviewing.

The benefit of using a client-server architecture is that the greater processing power and larger storage capacity of a server machine can be harnessed to run the resource intensive ASR application.

Automatic dictation applications have mostly been used by specific profes-

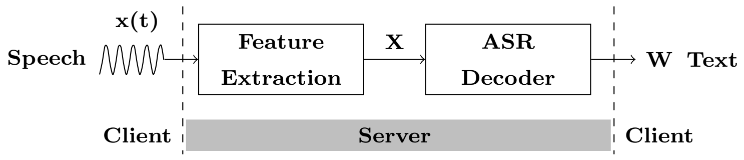

Figure 1.2: Network speech recognition (NSR) architecture. The recorded speech signal is sent from the client to the ASR server for recognition. The output text is then sent back to the client.

sions, such as lawyers and doctors. The target users for MobiDic are professionals who move a lot during their workday.

Because the actual speech recognition is done on a server, in an offline mode, recognition speed is not of big importance. The quality of the recognized text is what matters. The minimum performance requirement could be described as that the time it takes to correct the recognized text isn't longer than the time it would take to write the entire text on the mobile phone.

1.3 Model Adaptation

In this thesis work we will simulate the integration of MobiDic with the large vocabulary speech recognition system (LVCSR) developed in the department of computer and information science at TKK. This will be done by taking real life speech data recorded on the MobiDic client application and perform recognition tests on it with the TKK speech recognizer.

The emphasis in the thesis is on acoustic model and language model adaptation. Acoustic models and language models are usually estimated from large speech and text corpora. The corpus data is chosen so that the estimated models perform well on average for a wide variety of different users.

In order to develop a fully speaker dependent (SD) ASR system, several hours of speech from the user is needed. This much data from one speaker is usually not available and it is also impractical to collect large amounts of data from every user. Acoustic model adaptation uses smaller amounts of data, in the order of a few minutes of speech, to adapt the general acoustic model to better estimate the natural voice characteristics of the specific user. Most acoustic model adaptation methods also adapt to the environment and recording method.

Language model adaptation is done to improve the performance of the ASR system for a specific topic or speaking style. Texts from the selected topic are collected to either estimate a new language model or to adapt the general language model.

The goal of this work is to enhance speech recognition performance for a special user group of MobiDic, through the means of acoustic model and language model adaptation. The intended user group is lawyers. The language model will therefore be adapted to law-related texts. The acoustic model will be adapted to the specific user and recording enviroment. Adaptation is needed because there is mismatch between test data and background models both in terms of acoustics and language. The background acoustic models have been trained on speech recorded on a computer microphone, while the test data has been recorded on a mobile phone microphone. The background language models have been trained on texts from journals and news wire services, while test data is mainly law-related.

The model adaptation framework and the integration of MobiDic with the TKK speech recognition system is illustrated in Figure 1.3.

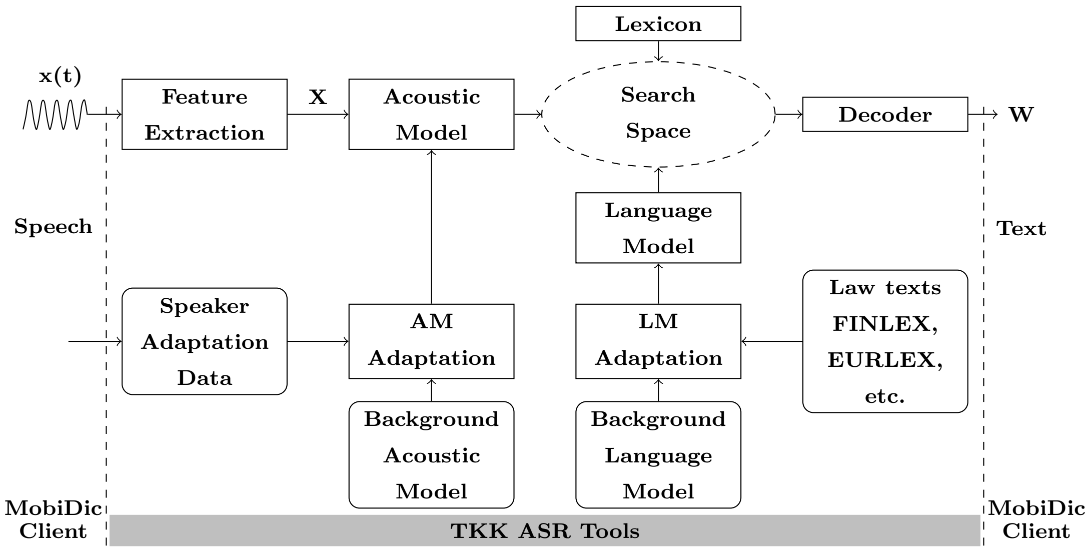

Figure 1.3: TKK ASR system runs on the server-side. The background acoustic model is adapted to the speaker and recording enviroment with speech data gathered from the MobiDic client. The background language model is adapted to law-related texts.

The use of different acoustic and language model adaptation methods will be studied in this work and the recognition performance will be compared to the unadapted system.

The outline of the thesis is the following. Chapter 2 deals with the construction and estimation of acoustic models. In Chapter 3, common acoustic model adaptation techniques are presented. Chapter 4 describes the estimation of language models. In Chapter 5, some of the most common language model adaptation techniques are presented. In Chapter 6, the experimental setup is described and the results are presented, analysed and discussed. Chapter 7 concludes the main findings of this work.

Chapter 2

Acoustic Modeling

2.1 Acoustic Models

Speech is in essence just a sequence of different sounds. Our brains are tuned to classify these sounds into basic phonetic units, or phonemes. From a sequence of phonemes we can distinguish words. From a pattern recognition point of view, this is quite an astonishing feat considering that the brain is also able to comprehend speech produced in different environments and by different speakers. Devising an algorithm for a computer to do the same is not a trivial matter.

Recording speech onto a computer and converting its' representation to numbers is however very easy and cheap with today's technology. In this light, it would seem more sensible to develop a data driven approach to classify and recognize speech sounds, rather than to program a knowledge-based expert system. This is the idea behind acoustic models that are used in most of today's speech recognition systems.

Acoustic models are statistical models that estimate the probability that a certain phoneme has been uttered in a recorded audio segment. The models are trained on several hours worth of pre-recorded speech. To give generality to the model the material includes speakers of different age and sex.

In the following sections we'll take a closer look at what characteristics of the speech signal are used for phoneme classification and how the acoustic models are constructed and trained.

2.2 Feature Extraction

2.2.1 Speech Signal

Digitally recorded speech is a discrete number representation of the air pressure changes that are produced when someone talks.

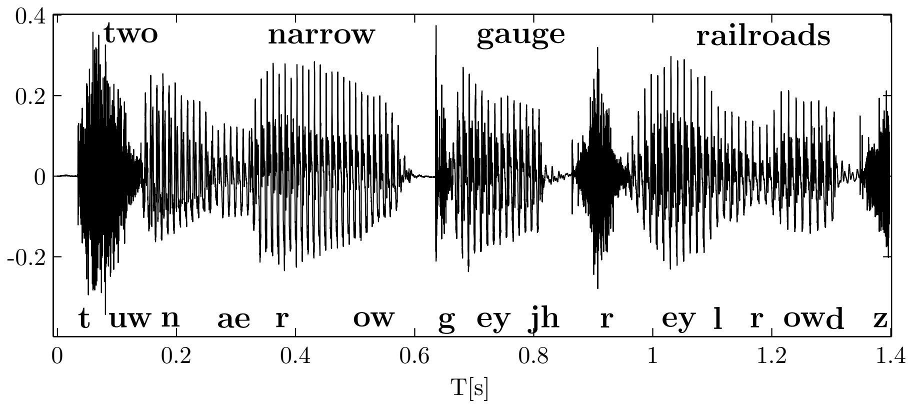

Figure 2.1: The sentence "Two narrow gauge railroads" is uttered. The individual phonemes are matched with the audio segments in which they occur.

The above figure shows the waveform of a digitally recorded speech signal. The waveform is made up of sequential samples. The number of samples depends on the length of the signal and the sample rate *F_S[Hz]*, how many times in a second the original signal is measured. The sample rate for this signal is 16 000 Hz. The length of the signal is 1.4 s, meaning a total number of 22 400 samples are used to represent the signal.

Another important factor is the resolution of the signal. That is the amount of memory (bits) used for the measurement of one sample. For the signal in Figure 2.1 the resolution is 16 bits, meaning a measurement can be assigned 65 536 different values. A higher resolution equals to a more accurate signal reconstruction.

In Figure 2.1 we can see that the phonemes are aligned with the audio segments in which they occur. Corpora used for training acoustic models usually come with phoneme level transcriptions of all the audio files. This means that the phonemes are aligned with the matching sample intervals.

Table 2.1: Phoneme level transcription

Start	**End**	**Phoneme**
0	2048	t
2048	4352	uw
4352	5376	n
5376	6144	ae
6144	6656	r
6656	8960	ow
8960	11008	g
11008	13056	ey
13056	15616	jh
15616	16128	r
16128	17152	ey
17152	19200	l
19200	19712	r
19712	20736	ow
20736	21760	d
21760	22400	z

Having access to the phoneme level transcriptions it's tempting to directly use the raw speech signal as an acoustic model. The raw speech signal carries all the vital information about the uttered phonemes, since the human brain can clearly interpret what words are pronounced. But the signal also carries a lot of excess information such as gender, mood, and other personal voice characteristics. An acoustic model should ideally be invariant to these features of the speech.

In addition, the waveforms take up a lot of memory. Using them directly as acoustic models consumes an unnecessary amount of computing resources.

Thus, further processing of the speech signal is still needed. Only the features which are essential for recognizing phonemes should be extracted and used for training.

2.2.2 Power Spectrum

When extracting vital information from any signal, it's usually more helpful to study the signal in the frequency domain rather than in the time domain. The discrete Fourier transform can tell us what frequency components S_k are dominant in a signal *s(n)*.

$$S_k = \sum_{n=0}^{N-1} s(n) e^{-j\frac{2\pi}{N}kn} \tag{2.1}$$

The value S_k gives the amplitude of the *k:th* frequency component. A good way to graphically illustrate the Fourier transform of a speech signal is a spectrogram. A spectrogram is a two dimensional colormap in the time-frequency plane, where the power of a frequency component is indicated by the color value of the plot.

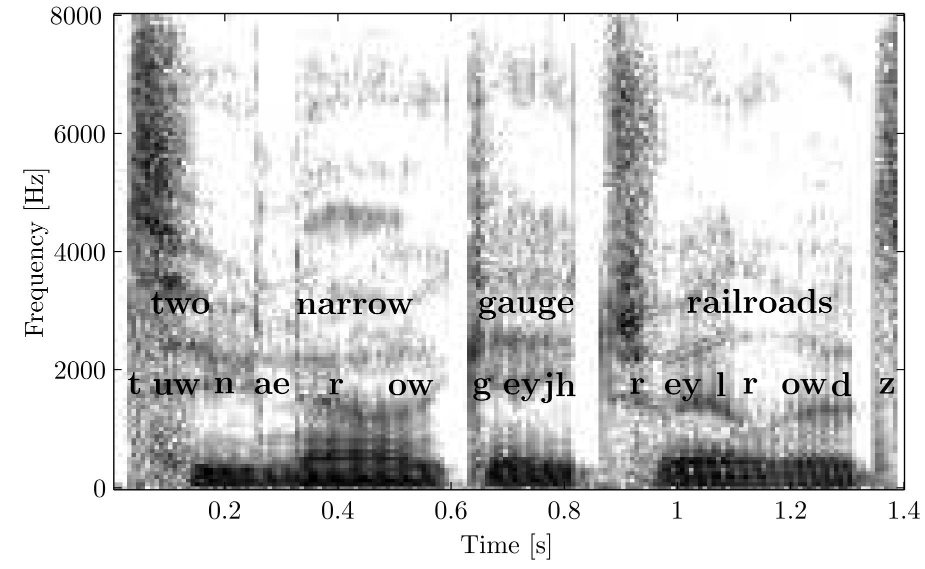

Figure 2.2: Spectrogram of the speech signal in Figure 2.1. Darker colors indicate higher power.

In Figure 2.2, the color value at a point (x,y) indicates the power of the frequency y at time x. As a general trend we notice that consonants have most of the power in the higher frequency bands.

The Fourier transform is a good starting point when attempting to classify speech signals. The relative magnitudes of the different frequency bands carry important information about what type of phonetic sounds are present in the signal.

To capture the frequency spectrum of individual phonemes the signal needs first to be split up into sample intervals, or frames. For a 16 000 kHz signal a frame length of 256 samples (16 ms) can be chosen. The frames are then set to overlap so that they start within 128 samples from each other.

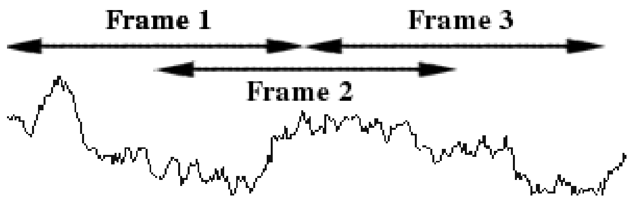

Figure 2.3: The waveform is segmented into frames. Each frame is 256 samples long and they start within 128 samples from each other.

A Fourier transform can now be applied to each individual frame. But first the frames have to be scaled with a Hamming window to reduce the distortions caused by the frame borders.

$$h(i) = 0.54 - 0.46cos(\frac{2\pi i}{N}) \tag{2.2}$$

A discrete Fourier transform is then applied to each frame t.

$$S_t(i) = \sum_{n=0}^{N-1} s(k+tN)h(k)e^{-j\frac{2\pi}{N}kn} \tag{2.3}$$

The Fourier transform $S_t(i)$ provides the frequency spectrum of an individual phoneme. Useful features can now be obtained from the frequency spectrum. The most elementary of features is the short-time power.

$$c_0 = \sum_{n=0}^{N/2} |S_t(i)|^2 \tag{2.4}$$

The short-time power c_0 can be thought of as the energy that the individual speech segment carries. It is a feature of the signal which is used for training acoustic models. Additional features are extracted from the different frequency bands of the signal.

2.2.3 Mel-frequency Cepstral Coefficients

Different frequency bands carry different significance when analysing speech. Because of the logarithmic nature of the human auditory system the acoustic features, which are vital for distinguishing phonemes, are not linearly separated on the frequency scale. The human ear is more sensitive to changes in pitch at low frequencies. It's logical to assume that frequency bands that determine phonetic differences are more narrow at lower frequencies and wider at higher frequencies. To map these frequency bands onto a linear scale the Mel-frequency scale is used [4].

$$f_{mel} = 2595\ log_{10}(1 + \frac{f}{700}) \tag{2.5}$$

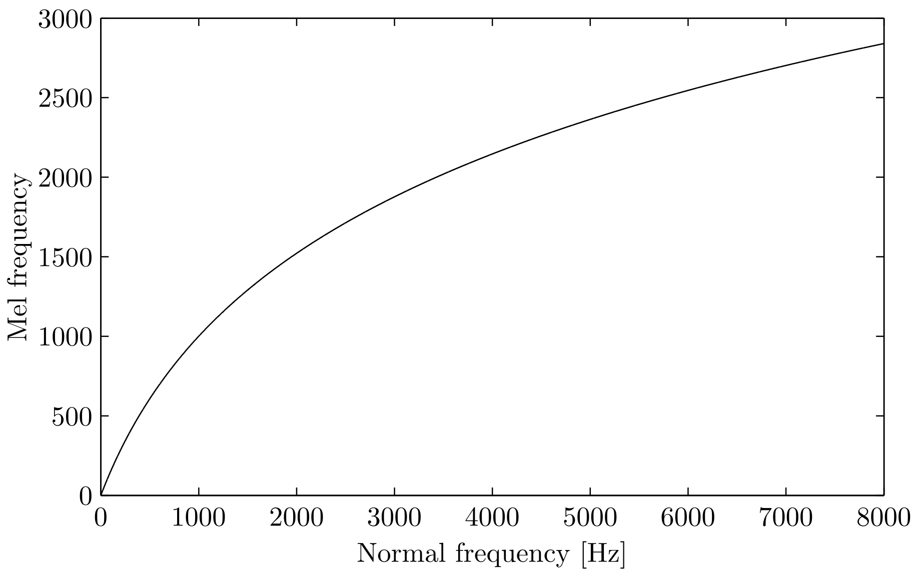

Figure 2.4: The relationship between the normal frequency scale and the Mel-frequency scale.

The Mel-frequency scale is divided into 21 equally long frequency bands and triangular band-pass filters are applied to each frequency band.

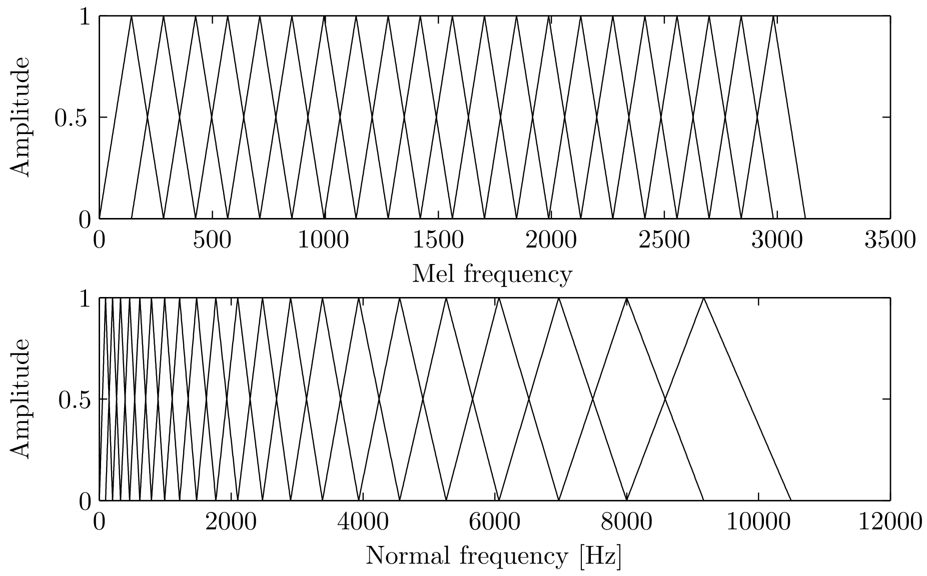

Figure 2.5: Triangular band-pass filters are applied to each frequency band. The frequency bands are located linearly on the Mel-frequency scale and non-linearly on the normal frequency scale.

The average energy *$S_{avg}(f_k)$* is calculated for each frequency band, where *f_k* is the center frequency of the *k:th* frequency band (k=0,1,..,20),

$$S_{avg}(k) = S_{avg}(f_k) = \frac{1}{N}\sum_{n=0}^{N} w_{FB}(n)S(f_k + \delta f(f_k, n)) \qquad (2.6)$$

and N is the number of samples taken in the band spectrum. The function *$\delta f(f_k,n)$* represents the neighbouring frequencies of *f_k* and *$w_{FB}(n)$* is a weighting function [4] .

As a final step in the signal processing chain, the log-values of *$S_{avg}(k)$* are calculated and a discrete cosine transform is applied to obtain the cepstrum ("spectrum of a spectrum") of the frequency band energies [4].

$$c_i = \frac{2}{N}\sum_{k=0}^{N-1} log|S_{avg}(k)|cos\left(\frac{2\pi}{N}ik\right) \qquad (2.7)$$

Typically twelve (i=1,..,12) cepstral coefficients are extracted from the cepstrum. These coefficients are called Mel-frequency cepstrum coefficients (MFCCs) because the computation is done in the Mel-frequency cepstrum. These twelve MFCCs ($c_1,\ c_2,\ ..\ ,\ c_{12}$) constitute together with the short-time power c_0 features of perceptual importance that are used for training acoustic models.

The MFCC parameters, introduced by Davis and Mermelstein in [5], tend to outperform other parametric representations of the speech signal, such as linear frequency cepstrum coefficients (LFCC), or linear prediction coefficients (LPC). It is thought that the MFCC parameters offer perceptually more relevant information about the short-time speech spectrum.

2.2.4 Delta Features

An additional set of features with more dynamic properties can be derived from the Mel-frequency cepstral coefficients. The time derivates Δc_i*(t)* of the basic features c_i *(i=0,1,..,12)* are also known as delta features and can be computed as follows [4]:

$$\Delta c_i(t) = \sum_{k=-M}^{M} k c_i(t+k) \tag{2.8}$$

where M is the number of neighbouring frames. The value of M is usually given a small value, such as 2.

Furthermore, the time derivates of the delta features themselves can also be computed, yielding so called delta-delta features. In the end, combining the basic features (13), delta features (13), and delta-delta features (13) gives a total number of 39 different features.

A 39-element observation vector $\mathbf{O}=[c_0\ \ c_1\ \ .\ \ .\ \ \Delta\Delta c_{12}]$ is thus computed for every frame in the speech signal, as illustrated in Figure 2.6.

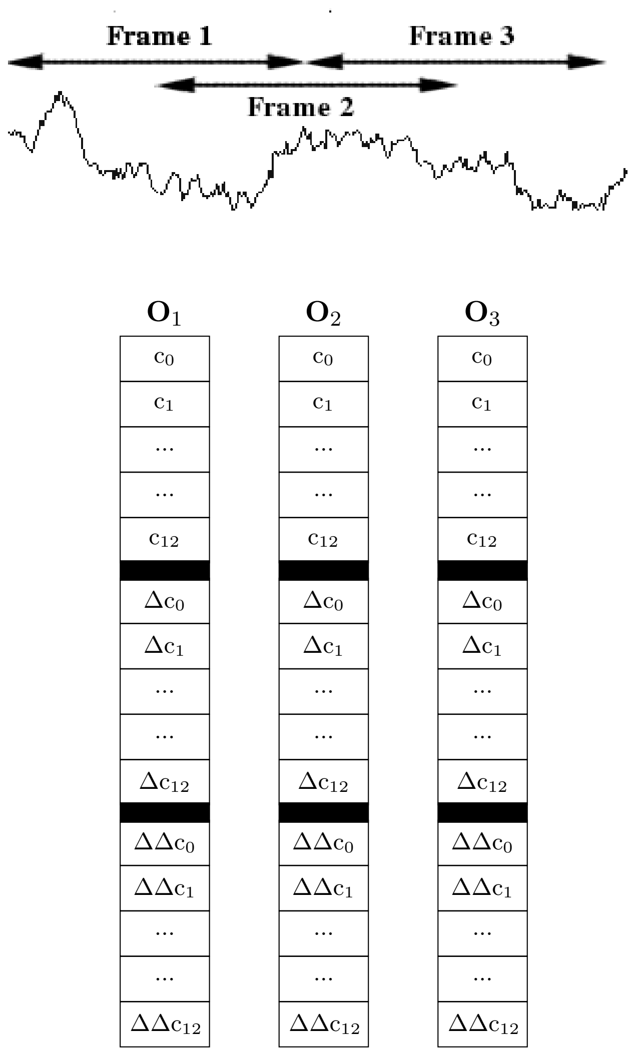

Figure 2.6: Each frame in the speech signal is represented by a 39-element MFCC observation vector.

Knowing the correct alignment between individual frames and phonemes, we can now start constructing statistical models for simple phoneme classification tasks.

2.3 Gaussian Mixture Models

An essential part in the construction of an acoustic model is choosing a method to classify phonemes based on feature observations. Given the observation vector **O**=[c_0 c_1 . . $\Delta\Delta c_{12}$] for a frame in a speech signal, estimates are needed for all phonemes, giving us the probability that this particular observation **O** corresponds to the occurrence of a certain phoneme */pho/*.

Having access to training data, which holds mappings between observation vectors and phonemes, the means and covariances of the feature vector elements can be calculated for each phoneme. The next step is to find a suitable probability distribution modeled around the mean vector.

Many naturally occurring phenomena have been successfully modeled by the Gaussian distribution. In the one-dimensional case, a random variable X, with a mean μ and variance σ^2, is said to have a Gaussian distribution if it has the following probability density function [6]:

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right] \tag{2.9}$$

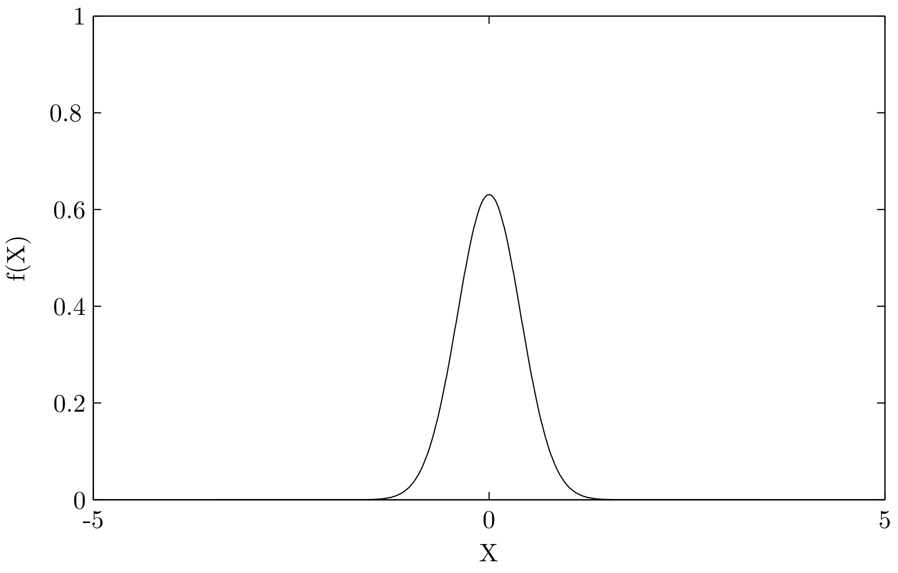

Figure 2.7: One-dimensional Gaussian distribution ($\mu = 0$, $\sigma^2 = 0.4$).

The one-dimensional Gaussian distribution in 2.9, can be generalized to any n-dimensional vector **X**:

$$f(\boldsymbol{X}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left[-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}-\boldsymbol{\mu})\right] \tag{2.10}$$

where $\boldsymbol{\mu}$ is the n-dimensional mean vector, $\boldsymbol{\Sigma}$ is the $n \times n$ covariance matrix, and $|\boldsymbol{\Sigma}|$ is the determinant of the covariance matrix $\boldsymbol{\Sigma}$ [6].

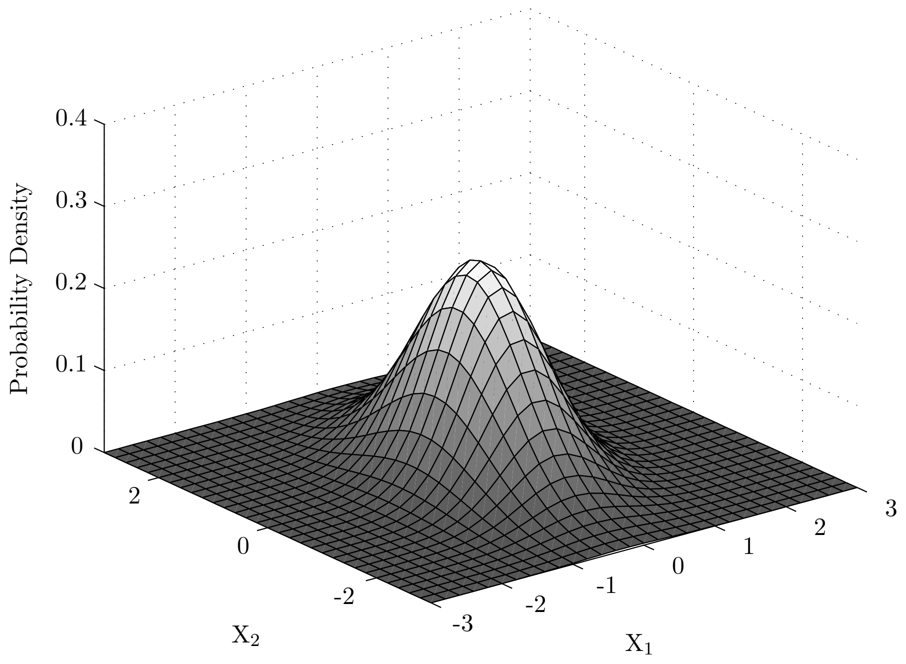

Figure 2.8: Multivariate two-dimensional Gaussian distribution.

In Figure 2.8 there is a graphical plot of a two-dimensional Gaussian distribution. Although it's harder to visualize a 39-dimensional Gaussian distribution the idea is the same when estimating the probability distributions for the MFCC observation vectors.

Lumping together phoneme statistics under a single Gaussian model has its' disadvantages. The observation vectors are usually not centered around a single point in vector space. They appear in clusters. This can be likened to the different classes of speaker variabilities encountered in the training data, such as gender, age, accent, or different speaking styles.

To estimate a more reliable acoustic model for a speaker independent ASR system, a mixture of different Gaussian density functions is preferable. Ideally, each distinct cluster in vector space should be assigned an individual Gaussian model.

The Expectation-Maximization (EM) algorithm [7] can be used to find the optimal cluster set and to estimate the Gaussian parameters for each cluster. The EM algorithm works by maximizing the log probability for the given training data and model parameters.

The final Gaussian mixture model (GMM) is estimated from the different cluster distributions as follows [6]:

$$\begin{aligned} f(\boldsymbol{X}) &= \sum_{m=1}^{M} w_m N_m(\boldsymbol{X}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m) \\ &= \frac{w_m}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_m|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu}_m)^T \boldsymbol{\Sigma}_m^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_m)\right] \qquad (2.11) \\ \textit{where} & \sum_{m=1}^{M} w_m = 1 \quad \textit{and} \quad w_m \geq 0 \end{aligned}$$

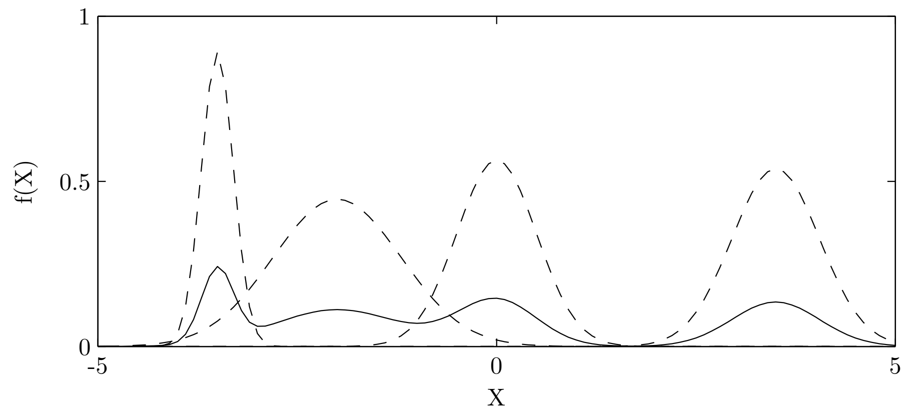

Figure 2.9: One-dimensional Gaussian mixture model. The dashed curves are Gaussian models of different clusters. The solid curve is the weighted sum of the different Gaussian distributions.

The mixture probability density function *f(**X**)* is a weighted sum of the individual Gaussian distributions $N_m(\boldsymbol{X};\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m)$. The graph of a one-dimensional Gaussian mixture model is shown in Figure 2.9.

By estimating a 39-dimensional Gaussian mixture model for each phoneme, we have a method for frame-wise phoneme classification based on the MFCC observation vector.

2.4 Hidden Markov Models

Having a means to classify phonemes frame by frame is already a good step towards a fully functioning ASR system. However, further processing is still needed.

When applying a GMM to classify the phonemes for the speech signal in Figure 2.1 (*"two narrow gauge railroads"*) we might expect to get an output similar to the following: */t/t/t/uw/uw/uw/n/n/n/ae/ae/ae/r/r/r/ow/ow/ow/g/g /g/ey/ey/ey/jh/jh/jh/r/r/r/ey/ey/ey/l/l/l/r/r/r/ow/ow/ow/d/d/d/z/z/z/*.

It's good to note that this is a raw simplification of the output since a 1.4s length signal like in Figure 2.1 would contain up to over 80 frames and the number of frames wouldn't be equally divided between the phonemes. Even recognizing this idealized case of frame output as the correct sentence is not a trivial matter but more on that later.

The GMM, like any statistical model, is not perfect. Classification errors are bound to happen considering that the pronunciation of many phonemes is very similar. Instead of the ideal clean cut frame output, we are more likely to get something like this: */t/d/t/uw/uw/ow/n/m/n/ae/aa/ae/r/r/r/ow/oh/ow/g/k /k/ey/ay/ey/jh/jh/jh/r/jh/r/ey/ae/ey/l/l/l/r/l/r/ow/ow/ow/d/t/t/z/s/s/*.

The frame output is distorted by acoustically similar, but erroneous, phonemes that pop up here and there. It's not totally impossible though to guess the correct phoneme sequence. For example, in the English language the phoneme */t/* is rarely followed by the phoneme */d/*. We can thus make an educated guess that the phoneme */t/* is uttered in the first time segment consisting of the output frames */t/d/t/*. Based on similar innate characteristics of the spoken language it's possible to construct a hidden Markov model (HMM) that estimates the likelihood for all phoneme sequences.

In the hidden Markov model the utterance of a phoneme is regarded as a hidden state which emits an observable representation of the signal, namely the MFCC observation vector. The phoneme specific GMMs provide a probability estimate that an observation is an emission of a certain state. State changes are modeled with transition probabilities a_{ij}, ie. the probability that a phoneme

i is followed by a phoneme j. Figure 2.10 illustrates the flowchart of an HMM with three states.

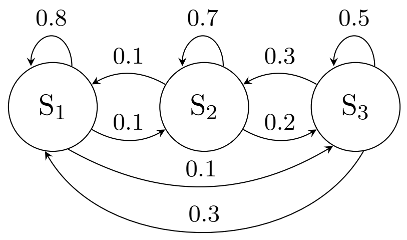

Figure 2.10: Flowchart of a 3-state HMM. The state transition probabilities are marked along with the flow lines.

The state transition probabilities a_{ij} form an $n \times n$ matrix $\boldsymbol{A}$, where n is the number of states in the HMM.

In speech recognition applications the number of HMM states is usually equivalent or correlated with the number of phonemes. A typical choice is to create a separate HMM for each phoneme and model each HMM with three states [8]. This is helpful in order to better capture the differences between the beginning, middle, and end of the phoneme.

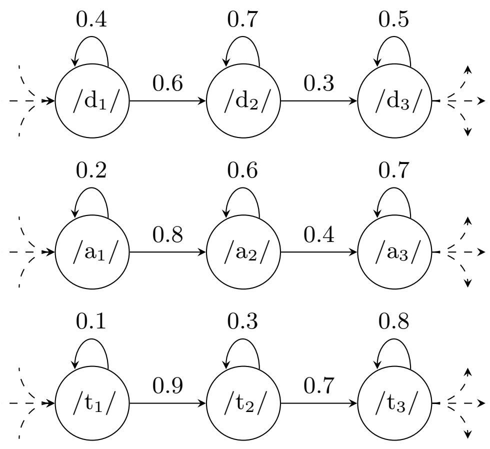

Figure 2.11: Each phoneme is modeled with its own 3-state HMM. The dashed lines show the inter-HMM transitions. The HMMs form a network through connections between each other's third and first states.

In this type of HMM framework, illustrated in Figure 2.11, the transitions are limited to self transitions and transitions to next states. The HMMs form a network through connections between each other's third and first states. This network of HMMs enables the creation of a phoneme-based speech recognizer [8].

An additional set of parameters in the HMM framework are the initial state probabilities π_i. They give an estimate for the probability that a state is the first in a sequence of states.

The state transition probability matrix $\boldsymbol{A}$ and the initial state probabilities π_i constitute the parameter set of an HMM. The values of $\boldsymbol{A}$ and π_i can be estimated from training data using the Baum-Welch algorithm [9]. Together with the emission probability density functions f_i they form a broader HMM parameter set commonly referred to as λ.

In continuous speech, phonemes are pronounced differently depending on the phonemes pronounced before and after. To model this variability, instead of modeling a phoneme with only one HMM, several HMMs are created for different contexts. A common choice is a triphone model, in which a phoneme is modeled in the context of the previous and following phoneme.

2.5 Viterbi Search

Given an observation sequence $\boldsymbol{O}=\boldsymbol{O}_1,\boldsymbol{O}_2,\ldots,\boldsymbol{O}_N$, the HMM $(\boldsymbol{A},\pi_i,f_i)$ can help us find the optimal matching state sequence $\boldsymbol{Q}=q_1,q_2,\ldots,q_N$. The object is to find a path $\hat{\boldsymbol{Q}}$ that maximizes the probability $P(\boldsymbol{Q}|\boldsymbol{O},\lambda)$ [9]:

$$\begin{aligned}\hat{\boldsymbol{Q}} &= \arg\max_{\boldsymbol{Q}} P(\boldsymbol{Q}|\boldsymbol{O},\boldsymbol{A},\pi_i,f_i)\\ &= \arg\max_{\boldsymbol{Q}} P(\boldsymbol{Q}|\boldsymbol{O},\lambda) \qquad (2.12)\end{aligned}$$

The Viterbi search algorithm, a technique based on dynamic programming methods, can be used to find the single best state sequence. The essence of the Viterbi search is storing the best path ending in each state i, for each time instant t. The quantity $\delta_t(i)$ is defined as follows [9]:

$$\delta_t(i) = \max_{q_1,q_2,\ldots,q_{t-1}} P(q_1q_2\ldots q_t = i, \boldsymbol{O}_1\boldsymbol{O}_2\ldots\boldsymbol{O}_t|\lambda) \qquad (2.13)$$

where $\delta_t(i)$ is the best score (highest probability) along a single path, at time t, ending in state S_i. For the time instant t+1, ending in state S_j, induction gives the following:

$$\delta_{t+1}(j) = \max_i [\delta_t(i) a_{ij}] f_j(\boldsymbol{O}_{t+1}) \tag{2.14}$$

To retrieve the best state sequence, we must keep track of the argument which maximizes 2.14, for each t and j. This is done via the array Ψ_t*(j)*.

The following procedure steps can be taken to find the best state sequence [9].

1. Initialization:

   At t=1, the best path ending in state S_i is simply.

   $$\delta_1(i) = \pi_i f_i(\boldsymbol{O}_1) \tag{2.15}$$

   No previous paths need to be stored in the array Ψ_t*(i)*.

   $$\Psi_1(i) = 0 \tag{2.16}$$

2. Recursion:

   At t+1, the best path ending in state S_j, can be determined from the best previous paths δ_t*(i)* combined with the transition probability a_{ij}. The path with the highest probability is chosen.

   $$\delta_{t+1}(j) = \max_i [\delta_t(i) a_{ij}] f_j(\boldsymbol{O}_{t+1}) \tag{2.17}$$

   The previous state which produced the best probability is added to the array.

   $$\Psi_{t+1}(j) = \arg\max_i [\delta_t(i) a_{ij}] \tag{2.18}$$

3. Termination:

   At the final time instant T, the state with the best ending path is chosen.

   $$q_T = \arg\max_i [\delta_T(i)] \tag{2.19}$$

4. Path (state sequence) backtracking:

   Starting from the best ending state q_T, we can retrieve the optimal path by backtracking along the pointers that are stored in the Ψ_t*(i)* array.

   $$q_t = \Psi_{t+1}(q_{t+1}), \quad t = T-1, T-2, ..., 1 \tag{2.20}$$

Figure 2.12 illustrates the Viterbi search method in a lattice structure, with circles representing the states and lines representing the optimal paths.

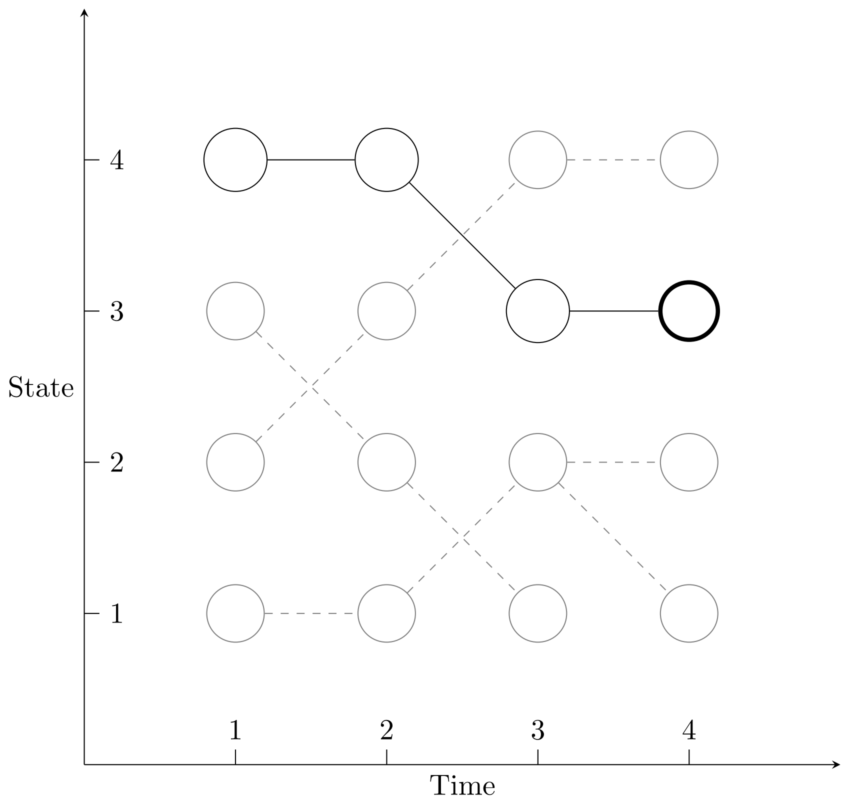

Figure 2.12: Viterbi search. The best path (line) to each state (circle) at each time instant is stored. The best path (solid line) is retrieved by backtracking from the best ending state (thick solid circle) at the last time instant.

2.6 Decoder

The Viterbi search applied alongside the GMM-HMM framework helped us find the optimal state sequence $\boldsymbol{Q}{=}q_1,q_2,...,q_N$ given the observation sequence $\boldsymbol{O}{=}\boldsymbol{O}_1,\boldsymbol{O}_2,...,\boldsymbol{O}_N$. The optimal state sequence can also be thought of as a representation of the optimal phoneme sequence.

The ultimate challenge is finding the optimal word sequence $\hat{\boldsymbol{W}}=W_1, W_2,..., W_m$, where the number of words m is unknown beforehand. This search problem, known in ASR applications as decoding, can be mathematically defined as maximizing the posterior probability $P(\boldsymbol{W}|\boldsymbol{O})$ [10].

$$\begin{aligned}\hat{\boldsymbol{W}} &= \arg\max_{\boldsymbol{W}} P(\boldsymbol{W}|\boldsymbol{O}) \\ &= \arg\max_{\boldsymbol{W}} \frac{P(\boldsymbol{O}|\boldsymbol{W})P(\boldsymbol{W})}{P(\boldsymbol{O})} \\ &= \arg\max_{\boldsymbol{W}} P(\boldsymbol{O}|\boldsymbol{W})P(\boldsymbol{W}) \qquad (2.21)\end{aligned}$$

Applying the Bayes rule on *P(**W**| **O**)* we end up with equation (2.21), which is a combination of two probability models. *P(**O**| **W**)* corresponds to the acoustic model and *P(**W**)* corresponds to the language model.

Before taking on the task of finding the most probable word sequence $\hat{\boldsymbol{W}}$, we need a model for the pronunciation of individual words. The information of how different words are pronounced is stored in a pronunciation dictionary, also known as a lexicon. It contains information of the exact phoneme sequence for every word. Tables 2.1 and 2.2 are example extracts from a monophone and triphone lexicon.

Table 2.1: Monophone pronunciation dictionary.

Word	**Pronunciation**
...	...
bannister	b ä n a s t ö
rudman	r a d m a n
gorman	g O r m a n
sustaining	s a s t E n i N
...	...

Table 2.2: Triphone pronunciation dictionary.

Word	**Pronunciation**
...	...
bannister	_-b+ä b-ä+n ä-n+a n-a+s a-s+t s-t+ö t-ö+_
rudman	_-r+a r-a+d a-d+m d-m+a m-a+n a-n+_
gorman	_-g+O g-O+r O-r+m r-m+a m-a+n a-n+_
sustaining	_-s+a s-a+s a-s+t s-t+E t-E+n E-n+i n-i+N i-N+_
...	...

In the case of an isolated-word recognizer, the vocabulary words with their specific phoneme sequences are mapped into an HMM network, as illustrated in Figure 2.13. The most probable word can be found by doing a Viterbi search over the HMM network.

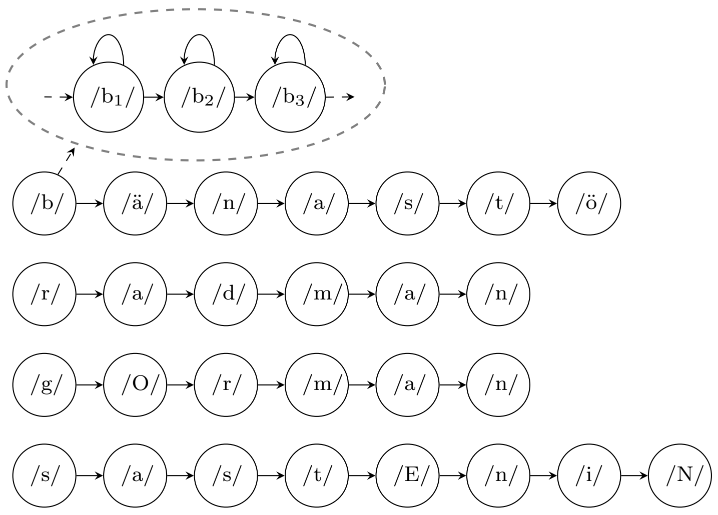

Figure 2.13: Search network for an isolated-word recognizer. The internal HMMs of the phonemes are concatenated. They form a separate HMM for each word.

For a continuous speech recognizer, the challenge of finding the optimal word sequence lies in the fact that there are no clear boundaries between the spoken words. The beginning of a new word can start at almost any time instant. Furthermore the language model $P(\boldsymbol{W})$ has to be incorporated into the search somehow.

A common strategy is to map the words and their concatenated phoneme HMMs into a bigger network, just like in Figure 2.13 but with transitions between the words [10]. The inter-word transition probabilities correspond to the language model $P(\boldsymbol{W})$. An example of a search network for a three-word vocabulary is found in Figure 2.14. In this case the language model is a bi-gram, so there is only one type of transition between the words.

In this type of search space, most methods try to find the optimal word sequence during a single run, simultaneously utilizing the knowledge sources of both the acoustic model and the language model. This is called a one-pass search strategy.

A one-pass search is usually performed so that the word hypotheses are formed in a time-synchronous fashion over the sequence of observation vectors [10]. Most one-pass search strategies are time-synchronous.

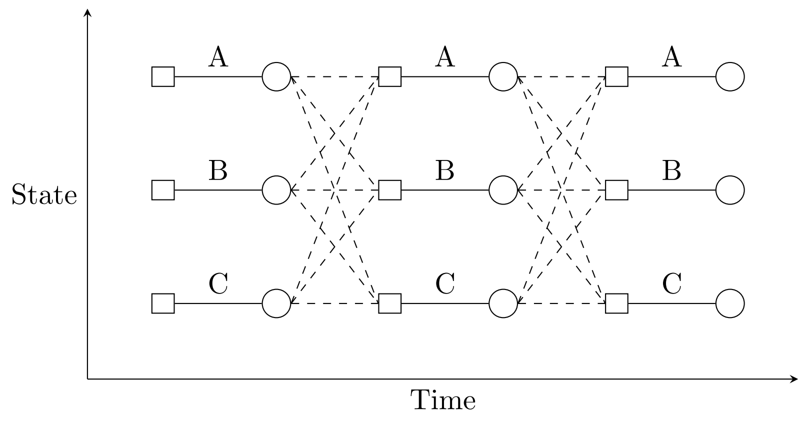

Figure 2.14: Search network for a three-word vocabulary (A,B,C) with a bigram language model. The acoustic model state transitions are represented as solid lines and the language model state transitions are represented as dashed lines.

A dynamic programming algorithm, such as the Viterbi search, is usually applied on the HMM network to find the most probable word sequence. Because the number of possible state sequences grows to astronomical proportions for a large vocabulary, a common method to reduce the search space is by removing the paths of unlikely word hypotheses whose scores fall under a fixed probability [10]. This type of search is known as a beam search, and it is implemented in many decoders.

Chapter 3

Acoustic Model Adaptation

3.1 Acoustic Model Adaptation

In the previous chapter we learnt that acoustic models are trained on audio corpora that contain several hours worth of pre-recorded speech from many different speakers. The estimated model parameters are an average representation of the entire group of speakers in the training set.

Deviations from the training conditions are inevitable when testing the ASR system in real life situations. The performance gets worse the less similar the testing conditions are compared to the training conditions [11]. This is illustrated in the graph in Figure 3.1.

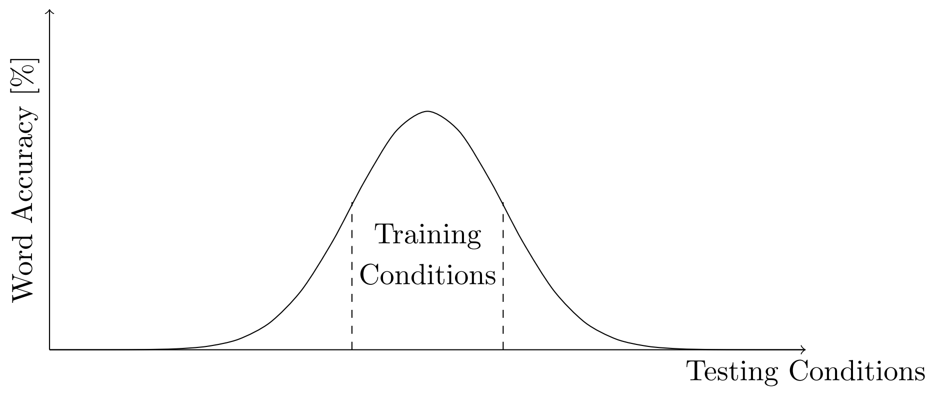

Figure 3.1: Acoustic variability degrades speech recognition performance.

The sources of these deviations can be grouped into environmental noise, different channels (telephone, near- or far-field microphone), and speaker variability [11]. The speaker variability factor is the toughest one to cancel out. It stems in part from the physiological differences in the vocal tract, but accent/dialect, cultural and emotional factors also account for the individual voice characteristics.

There are many different techniques which have been developed to compensate for speaker variability. The goal for all these adaptation methods is to minimize the difference between the acoustic model and the selected speaker. A common practice is to sample speech data from the new speaker and update the acoustic model according to the features which are extracted from the speech.

A speaker adaptation system can operate in two different modes. The adaptation can either be supervised or unsupervised [12]. In the supervised mode the transcription of the speech data is known beforehand. In the unsupervised mode the transcription is not known, and is taken from the output of a speech recognizer.

The distinction between static and dynamic adaptation can also be made [12]. In static mode all the adaptation data is made available at once and is used to adapt the final system during a single run. In dynamic mode the adaptation data is acquired in parts and the system is continuously adapted over time.

In the following sections two of the most used speaker adaptation techniques will be presented: maximum likelihood linear regression (MLLR) and vocal tract length normalization (VTLN).

3.2 Maximum Likelihood Linear Regression

The maximum likelihood linear regression (MLLR) method belongs to a family of adaptation techniques based on linear transformations. The general approach is to estimate a linear transformation of the model parameters, in order to construct a more appropriate acoustic model.

It is considered that the main differences between speakers are characterized by the Gaussian means. In MLLR adaptation a linear transformation is estimated to update the Gaussian mean parameters, as follows:

$$\hat{\boldsymbol{\mu}} = \boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b} \tag{3.1}$$

where $\boldsymbol{A}$ is an $n \times n$ matrix and $\boldsymbol{b}$ is an n dimensional vector [12]. The value of n is the dimensionality of the observations, 39 in the case of an MFCC observation vector.

Equation (3.1) can be simplified to

$$\hat{\boldsymbol{\mu}} = \boldsymbol{W}\boldsymbol{\xi} \tag{3.2}$$

where $\boldsymbol{W}$ is an $n \times (n+1)$ matrix and $\boldsymbol{\xi}$ is the extended mean vector

$$\boldsymbol{\xi}^T = [1 \quad \mu_1 \quad \ldots \quad \mu_n] \tag{3.3}$$

The matrix $\boldsymbol{W}$ is estimated such that the likelihood of the adaptation data is maximised [12]. The Expectation-Maximisation (E-M) algorithm can be used to estimate $\boldsymbol{W}$.

The power of this most basic form of MLLR lies in that only a small amount of adaptation data is needed to estimate a global transformation matrix $\boldsymbol{W}$. The global transformation matrix can then be used in (3.2) to transform Gaussian means of phonemes or triphones that haven't even been observed in the adaptation data.

As was noted earlier, it is generally thought that the most important speaker specific effect can be attributed to the Gaussian means. An extension to basic MLLR is to also transform the Gaussian variances. The transformation of the covariance matrix $\boldsymbol{\Sigma}$ takes the following form:

$$\hat{\boldsymbol{\Sigma}} = \boldsymbol{H}\boldsymbol{\Sigma}\boldsymbol{H}^T \tag{3.4}$$

The Gaussian means and variances can be transformed independently with equations (3.2) and (3.4). That is, separate transformation matrices $\boldsymbol{W}$ and $\boldsymbol{H}$ are estimated for each parameter set. In the constrained transform case, the means and variances are set to use the same transformation matrix $\boldsymbol{A}_c$.

$$\begin{aligned} \hat{\boldsymbol{\mu}} &= \boldsymbol{A}_c\boldsymbol{\mu} - \boldsymbol{b}_c && (3.5) \\ \hat{\boldsymbol{\Sigma}} &= \boldsymbol{A}_c^T\boldsymbol{\Sigma}\boldsymbol{A}_c && (3.6) \end{aligned}$$

This form of adaptation is known as constrained MLLR (cMLLR).

Instead of using the same global transformation matrix for all Gaussian models, different transformation matrices can be tied to Gaussians that are close to each other in acoustic space [12]. The transformation matrices are arranged into so called regression classes.

The organization of the regression classes can either be fixed or dynamic. The regression classes are fixed if their definitions are determined beforehand, based on adaptation data. The optimal number of regression classes is roughly proportional to the amount of adaptation data [13]. For example, a division into Gaussian models representing vowel and consonant sounds could be made. Hence, two transformation matrices are estimated for each group. In Figure 3.2 the adaptation framework for two fixed regression classes is shown.

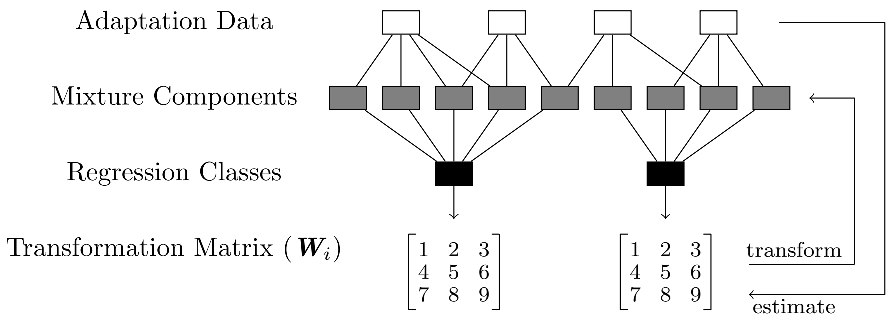
Adaptation Data
Mixture Components
Regression Classes
Transformation Matrix ($\boldsymbol{W}_i$)
$\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$
$\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$
transform
estimate

Figure 3.2: Adaptation framework for two fixed regression classes. The mixture components are divided into separate regression classes. The transformation matrices $\boldsymbol{W}_i$ are estimated to maximise the likelihood of the adaptation data.

First, the size of the adaptation data is determined. After that the mixture components are divided into an optimal number of different regression classes. The class definitions, what mixture components belong to which class, are determined beforehand. Finally, the transformation matrix $\boldsymbol{W}_i$ for each class is estimated to maximise the likelihood of the adaptation data.

Fixed regression classes work well in some cases when the amount of adaptation data is evenly distributed among the classes. However, if classes are assigned with insufficient data then the estimates will be poor. Therefore, it would be desirable to also determine the content distribution of the adaptation data and make the division into regression classes based on this. In other words, the regression classes are defined dynamically based on the the type of adaptation data that is available.

Dynamic regression classes and the mixture components that are tied to them are organized into a tree-like structure, as illustrated in Figure 3.3.

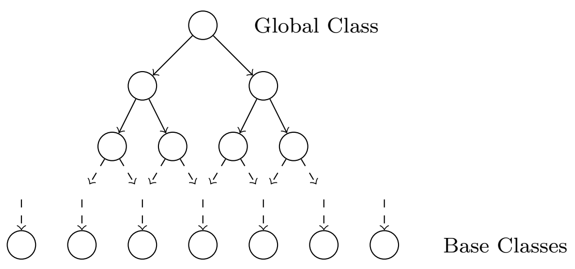

Figure 3.3: Regression class tree. The leaves of the tree represent the individual mixture components. At the top of the tree is the global class under which all mixture components can be grouped.

The leaves of the regression tree in Figure 3.3 represent individual mixture components and at higher levels the mixture components are merged into groups of similar components based on a distance measure between components [13]. The root node represents the global transformation matrix under which all mixture components can be grouped.

The regression tree is used to decide for what classes there is a sufficient amount of data that a transformation matrix can be reliably estimated. A search is made through the tree starting at the root. A transformation matrix is estimated at the lowest level in the tree for which regression class there is sufficient data. This allows adaptation data to be used in more than one regression class and it ensures the mixture components are updated with the most specific transformation matrix [13].

Typically a mean-only global MLLR adaptation gives a 15% reduction in WER compared to an unadapted model, using about a minute of adaptation data [12]. The WER tends to saturate very quickly for global MLLR, so adding more adaptation data usually does not improve performance. In experiments done by Leggetter and Woodland in [13], a WER reduction of 23% was achieved for a global MLLR using 40 sentences as adaptation data. For a similar MLLR adaptation that used regression classes the WER reduction was 55%.

Although MLLR has so far been presented as a speaker adaptation method, the technique can also be used in the same way for environmental adaptation.

3.3 Vocal Tract Length Normalization

It was stated earlier that the main source of speaker variability is the variation in length and shape of the vocal tract. The difference in vocal tract length tends to scale the frequency axis. For example, women tend to have shorter vocal tracts than men. As a result of this, female speakers exhibit formants (spectral peaks of sound) that are on average 20% higher than for male speakers [14].

Vocal tract length normalization (VTLN) is a method developed to reduce the effects of different lengths of the human vocal tract. In general, the method works by re-scaling the frequency axis with a warping function g_α, where α is the transformation parameter [14].

$$\begin{aligned} \tilde{\omega} &= g_\alpha(\omega) \\ \text{where} \quad 0 &\leq \omega \leq \pi \end{aligned} \tag{3.7}$$

The original frequency ω is transformed into the warped frequency $\tilde{\omega}$. The transformation is done in the frequency domain between 0 and π, where π corresponds to the Nyquist frequency.

In an MFCC framework, the warping function is applied on the Fourier power spectrum. The warped spectrum is then fed onwards to the MFCC filterbanks, as illustrated in Figure 3.4.

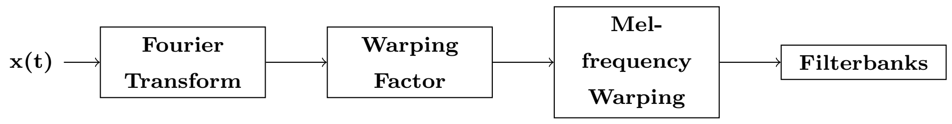

Figure 3.4: The warping factor is applied on the signal in between the Fourier transform and the Mel-frequency warping.

The warping function g_α can either be linear or non-linear. A linear function $g_\alpha(\omega)=\alpha\omega$ models the vocal tract as a uniform tube of length L, where a change in the length of the vocal tract from L to kL results in a scaling of the frequency axis by a factor $1/k$ $(=\alpha)$ [14].

Usually a warping function is defined as piecewise linear, to handle the bandwidth mismatching problem [15].

$$g_\alpha(\omega) = \begin{cases} \alpha\omega & \text{if } \omega < \omega_0 \\ b\omega + c & \text{if } \omega \geq \omega_0 \end{cases} \tag{3.8}$$

The linear warping function is not the best model of the vocal tract. A non-linear transformation is preferable and the bilinear warping function is often used [15].

$$g_\alpha(\omega) = \omega + 2tan^{-1}\frac{(1-\alpha)sin(\omega)}{1-(1-\alpha)cos(\omega)} \tag{3.9}$$

In Figures 3.5 and 3.6 curves of piecewise linear and bilinear warping functions are presented. It's noticeable for piecewise linear warping functions that $\alpha > 1.0$ corresponds to the compressing of the spectrum and $\alpha < 1.0$ corresponds to stretching of the spectrum. The opposite is true for bilinear warping functions, $\alpha > 1.0$ corresponds to the stretching of the spectrum and $\alpha < 1.0$ corresponds to compressing of the spectrum.

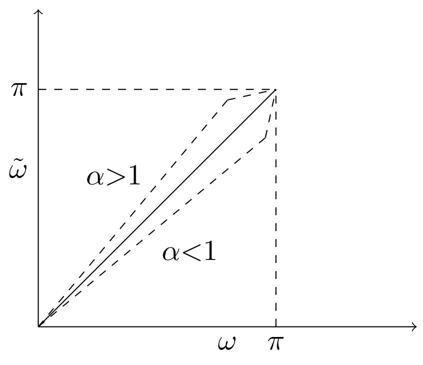

Figure 3.5: Piecewise linear warping function.

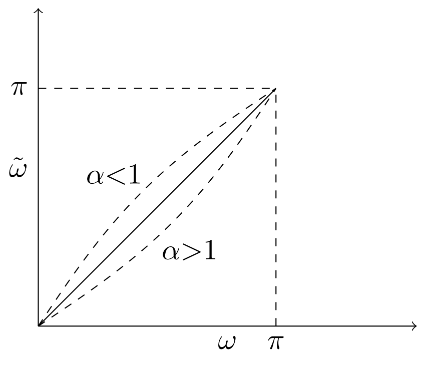

Figure 3.6: Bilinear warping function.

The optimal value of α is estimated on speaker-specific adaptation data. Similar to the estimation of the MLLR transformation matrix, the value of α is estimated to maximise the likelihood of the adaptation data.

The relationship between MLLR and VTLN was studied by Pitz and Ney in [16]. It showed that VTLN adaptation results in a linear transformation in the cepstral domain, and can therefore be considered as a special case of MLLR. This explains experiments showing that improvements obtained by VTLN and MLLR are not additive in some cases.

In experiments done by Uebel and Woodland in [17], the performance increases of VTLN and MLLR were compared. On average, the relative WER reduction for unconstrained MLLR was 9.7% and for VTLN (bilinear warping function) 6.6%. As a contradiction to [16], the joint improvement of using both VTLN and MLLR was additive, with an average WER reduction of 12.6%.

Chapter 4

Language Modeling

4.1 N-gram Models

The main goal of statistical language modeling is to improve the performance of various natural language applications. In speech recognition, this means finding the best possible estimate for *P(W_i)*, that is the probability for the word W_i to occur.

The most commonly used method in speech recognition for estimating *P(W_i)* is n-gram language modeling. N-grams are estimated from large text corpora containing millions of words covering a vast range of topics. N-gram models utilize the word history in a sentence. It is assumed that the probability for W_i only depends on the n - 1 previous words. The probability *P($W_i|H$)*, where H is the n - 1 preceding words, is estimated by counting the number of occurrences of the word contexts $W_{i-n+1},..,W_{i-2},W_{i-1},W_i$ and $W_{i-n+1},..,W_{i-2},W_{i-1}$ in the text corpus.

To illustrate a typical case, we can set n=3, which we call a trigram. We get the counts *C(W_{i-2},W_{i-1},W_i)* and *C(W_{i-2},W_{i-1})*. From these counts the probability is calculated using the maximum-likelihood estimate.

$$P(W_i|H) = \frac{C(W_{i-2},W_{i-1},W_i)}{C(W_{i-2},W_{i-1})} \tag{4.1}$$

Although seemingly a simple method for capturing regularities in written texts, n-grams offer a robust technique for modeling natural languages. The implementation is independent of language and domain. The diminished costs of mass storage space and the greater accessibility to large text corpora provided by the Internet and other mediums has also contributed to making n-grams a de facto standard for language modeling in speech recognition, and other natural language applications such as optical character recognition and machine translation.

4.2 Smoothing

4.2.1 Zero Probability Problem

As described earlier, n-grams are trained on text corpora which at best contain up to billions of words. This might seem like a sufficient amount of text to capture all possible n-grams that could ever occur in the language. The reality is however, that many n-grams are never seen in the training text, ie. for many word sequences $C(W_{i-2}, W_{i-1}, W_i)$=0. This causes also the probability $P(W_i|H)$ to become zero. This is clearly a problem. Let's consider the following example. We find 10 occurrences of the trigram *"holiday in Greece"* and 50 occurrences of the bi-gram *"holiday in"* in our training corpus. We can thus from (4.1) calculate the probability *P("Greece"| "holiday in")*:

$$P(\text{"}Greece\text{"}|\text{"}holiday\ in\text{"}) = \frac{C(\text{"}holiday\text{"}, \text{"}in\text{"}, \text{"}Greece\text{"})}{C(\text{"}holiday\text{"}, \text{"}in\text{"})} = \frac{10}{50} = 0.2$$

As we can see, this yields a non-zero probability and everything is fine so far. However, if we for example want to calculate the probability for the trigram "holiday in Bhutan", when *C("holiday","in","Bhutan")*=0 in the training text, we end up with:

$$P(\text{"}Bhutan\text{"}|\text{"}holiday\ in\text{"}) = \frac{C(\text{"}holiday\text{"}, \text{"}in\text{"}, \text{"}Bhutan\text{"})}{C(\text{"}holiday\text{"}, \text{"}in\text{"})} = \frac{0}{50} = 0$$

Although an uncommon vacation spot and a word sequence that is rarely seen in the English language, zero is clearly an underestimate for this probability.

When generating n-gram probabilities through the maximum likelihood estimate as in (4.1), we tend to get too high probability estimates for n-grams that are seen in the training text and too low estimates for the n-grams that are not seen. We would like to correct this unbalance by taking some probability mass from the seen events and redistribute it to all the unseen events. This is called language model smoothing.

4.2.2 Add-one Method

Many smoothing methods have been developed over the years. One common strategy to tackle the problem of uneven probability distributions is to manipulate the n-gram counts. Beginning with the simplest of methods, we simply add one to every n-gram count,

$$P(W_i|H) = \frac{1 + C(W_{i-2}, W_{i-1}, W_i)}{|V| + C(W_{i-2}, W_{i-1})} \tag{4.2}$$

where $|V|$ is the total number of words in the vocabulary. This takes away easily the zero probability problem. Instead of adding just 1 we can generalize this method to adding any constant k. However, these additive methods tend to perform rather poorly. The unseen n-grams are usually given too high probabilities.

4.2.3 Absolute Discounting

Instead of adding counts to every possible word context, a more successful smoothing scheme is to subtract counts from the seen n-grams, and distribute this left-over probability to the unseen n-grams. The probability redistribution to the unseen n-grams is usually done by backing off to lower orders of the word context. A good way to illustrate how this works in practice is to present the absolute discounting method [18], where both discounting and backing off are used together in the smoothing process.

$$P(W_i|H) = \begin{cases} \frac{C(W_{i-2},W_{i-1},W_i)-D}{C(W_{i-2},W_{i-1})} & \text{if } C(W_{i-2}, W_{i-1}, W_i) > 0 \\ P(W_i|W_{i-1}) * \beta_{W_{i-2},W_{i-1}} & \text{otherwise} \end{cases} \tag{4.3}$$

In absolute discounting, the n-gram count $C(W_{i-2}, W_{i-1}, W_i)$ is subtracted with a constant D and after that the maximum-likelihood estimate is calculated as normal. The constant D is typically optimized on held out data and the value is usually between 0 and 1. If $C(W_{i-2}, W_{i-1}, W_i)$ is zero, we back off to the word context W_{i-1}, W_i and multiply the corresponding probability $P(W_i|W_{i-1})$ with $\beta_{W_{i-2},W_{i-1}}$. The factor $\beta_{W_{i-2},W_{i-1}}$ is a back-off weight to ensure the probabilities sum to 1. In the case that the count $C(W_{i-1}, W_i)$ is also zero we simply back off to the unigram probability.

4.2.4 Kneser-Ney Smoothing

A further specialization of the absolute discounting method is Kneser-Ney smoothing [19]. This smoothing method differs mainly in how we estimate the lower order distribution. In absolute discounting we simply multiply the discounted maximum-likelihood estimate $P(W_i|\,W_{i-1})$ with the back-off weight $\beta_{W_{i-2},W_{i-1}}$ to get the lower order probability. Kneser-Ney smoothing introduces a modified probability function for the back-off n-grams. Instead of estimating the probability based on the number of occurrences Kneser-Ney smoothing derives the back-off probability from the number of distinct contexts the n-gram appears in.

$$P(W_i|H) = \begin{cases} \frac{C(W_{i-2},W_{i-1},W_i)-D}{C(W_{i-2},W_{i-1})} & \text{if } C(W_{i-2},W_{i-1},W_i) > 0 \\ \frac{N_{1+}(\bullet,W_{i-1},W_i)}{N_{1+}(\bullet,W_{i-1},\bullet)} * \beta_{W_{i-2},W_{i-1}} & \text{otherwise} \end{cases} \tag{4.4}$$

where $N_{1+}(\bullet, W_{i-1}, W_i)$ is the distinct number of trigram contexts the bigram W_{i-1}, W_i appears in and $N_{1+}(\bullet, W_{i-1},\bullet)$ is the total number of trigram contexts found in the training text where W_{i-1} is the middle word.

We can motivate the use of context counts over occurrence counts with an example. Let's consider the rare trigram *"beerfest at Tiffany's"*. There are no occurrences of this word sequence in the training text we are using. If we want to find a probability for this trigram using absolute discounting we must back off to the bi-gram probability *P("Tiffany's"|"at")*. This has quite a high probability, but mainly because of the large occurrence of the trigram *"breakfast at Tiffany's"*. To get an approximation from the real world, a Google search finds 1 340 000 hits for *"at Tiffany's"* and 1 110 000 hits for *"breakfast at Tiffany's*. We can thus deduce that a large part, about 80%, of the probability mass for *P("Tiffany's"|"at")* comes from a single trigram context. We will end up giving too much of this probability mass away if we use the maximum likelihood estimate *P("Tiffany's"|"at")* when backing off from unseen trigrams like *"beerfest at Tiffany's"*. To handle this problem the context counts $N_{1+}(\bullet, W_{i-1}, W_i)$ and $N_{1+}(\bullet, W_{i-1},\bullet)$ are used for estimating the back-off probability in Kneser-Ney smoothing. This will help to reduce the probabilities for rare n-grams that have a word sequence which occurs frequently but only in a few contexts.

4.2.5 Modified Kneser-Ney Smoothing

An improved variation of Kneser-Ney smoothing exits, called modified Kneser-Ney smoothing, presented by Chen and Goodman in [20]. The discount constant D for all non-zero counts is optimized. Instead of using the same discount constant for all n-grams, a discount function *D(c)* is introduced, where c is the number of occurrences the n-gram has.

$$D(c) = \begin{cases} 0 & \text{if } c = 0 \\ D_1 & \text{if } c = 1 \\ D_2 & \text{if } c = 2 \\ D_{3+} & \text{if } c \geq 3 \end{cases} \tag{4.5}$$

In [18], Ney. et al. suggest estimating D using a discounting method based on the Good-Turing estimate [21],

$$D = \frac{n_1}{n_1 + 2n_2} \tag{4.6}$$

where n_1 is the number of n-grams occurring once and n_2 the number of n-grams occurring twice. For the discount constants D_1, D_2, and D_{3+}, the corresponding estimates are derived from D as follows:

$$\begin{aligned} D_1 &= 1 - 2D\frac{n_2}{n_1} \\ D_2 &= 2 - 3D\frac{n_3}{n_2} \\ D_{3+} &= 3 - 4D\frac{n_4}{n_3} \end{aligned} \tag{4.7}$$

There are many other smoothing techniques which have been developed over the years. Most of these are based on different modifications of the discounting and backing off mechanisms. However, an extensive study about smoothing methods, done by Chen and Goodman in 1998 [20], showed that modified Kneser-Ney smoothing outperforms all other techniques.

4.3 Varigram Language Models

In the presented n-gram model examples so far, we've only considered cases where n=3. Longer range n-gram models would be more desirable since they could give an even better approximation for *P(W|H)*. We can see how this works by considering the example sentence *"Harry rides a big red bicycle"*.

Trying to estimate the probability for the last word *"bicycle"* in this sentence with only a trigram, *P("bicycle"|"big red")*, yields a much lower probability than if the whole sentence is taken into account. Trigrams such as *"big red firetruck"* or *"big red house"* are given about the same or higher probability than *"big red bicycle"*. On the other hand, if we were to use a longer range n-gram model that could capture the preceding word *"rides"* in its' context, the probability *P("bicycle"|H)* would be greatly elevated in comparison to other words.

We can see from the above example, that for an ideal approximation, an n-gram model that would reach all the way back to the beginning of the sentence is needed. There is a problem though when estimating the probability of longer n-gram contexts. Even for the biggest training text corpora the unreliability of n-gram probability estimates increases for longer contexts. This is because of the simple reason that the number of possible sentences grows for longer contexts but sparsity of training data causes longer n-grams to be overestimated. That's why it's common in most applications to use models of lower order ($n = 3 \ldots 5$). Deletion of rarely occurring n-grams from the model is one way of tackling the data sparsity and unreliability problem. This is called pruning and a common estimate is that as much as half of the n-grams can be removed without affecting the performance of the language model.

A method for pruning Kneser-Ney smoothed n-gram models is presented by Siivola et al. in [22]. In this method the language model is pruned by deleting n-grams whose probability falls under a threshold ϵ, while at the same time taking into consideration the effect of Kneser-Ney smoothing. Additionally, the computational complexity of computing n-gram counts for longer spans is reduced by the incremental growing of language models. This means, that instead of first computing all the n-gram counts up the highest order and then apply pruning, new n-grams $\boldsymbol{h}w$ are added from lower order n-grams $\boldsymbol{h}$ that have survived pruning and are already in the model. The main benefit of this is that counts $C(\boldsymbol{h}w)$ only need to be computed for histories $\boldsymbol{h}$ that are already in the model.

It can be concluded, that the method described above can calculate reliable long span n-gram models. It achieves this by removing rarely occurring n-grams.

4.4 Morph Based Language Models

We have so far considered words to be the most fundamental units in language modeling. In many languages, such as English, a lexicon containing 50 000 - 100 000 most common words is enough to capture nearly all of the unique words occurring in the billion word text corpora used for training the n-gram models. In other languages however, the situation is different. In Finnish for example, a lot of words are originally formed from their stem words by the use of different morphological processes such as inflection, derivation and compounding. Because of this the number of different word forms grows to be very large. One typical difference between a morphologically rich language, such as Finnish, and a morphologically poor language, such as English, is the use of suffixes at the end of words instead of prepositions.

Table 4.1: Prepositions vs. suffixes

English	**Finnish**	**English**	**Finnish**
car	auto	house	talo
in the car	autossa	in the house	talossa
out of the car	autosta	out of the house	talosta
to the car	autolle	to the house	talolle
in to the car	autoon	in to the house	taloon
from the car	autolta	from the house	talolta

From the example illustrated in the above table, we can see that for every new noun that is introduced, the Finnish lexicon grows with five possible new inflected words that are derived from the original stem word. In reality, the number of possible word forms in the Finnish language that can be derived from a noun is above 2000. This causes the lexicon to grow very rapidly. However, a bigger problem is that many word forms are never seen in the training corpora and are therefore not added to the lexicon. This in turn causes a massive increase of out-of-vocabulary (OOV) words. These are words that the speech recognition system has no means of recognizing because they are not in the lexicon.

Research has been done on finding alternatives to word-based language models, which are better optimized for morphologically rich languages. An intuitive approach would be to use subwords. In other words, splitting words into smaller fragments. Subwords could represent the shortest semantic building blocks in a word, for example *lefthanded* $\rightarrow$ *left+hand+ed*. In [23], Creutz et al. describe the Morfessor algorithm, which given a training corpus text, finds the most frequently occurring subwords, or morphemes. The algorithm can on statistical basis calculate the optimal morpheme set to represent the language. It tries to find a balance between the size of the morph lexicon and the compactness of the representation of the training text corpus. The smallest possible morph lexicon consists only of the individual letters in the alphabet. This will however radically increase the representation size of the training text because every word would be split up into as many morphs as the number of letters it contains. At the other extreme, if words aren't split up at all the lexicon grows very large because now every distinct word will have to be in the morph lexicon.

Table 4.2: The correlation between lexicon and representation size

|l|e|f|t|h|a|n|d|e|d|	→	small lexicon size, big representation size
|lefthanded|	→	big lexicon size, small representation size
|left|hand|ed|	→	optimal lexicon size and representation size

Since Morfessor is a statistical data-driven algorithm, the generated morphemes don't necessarily have to have a clear correspondence to any linguistic morpheme segmentations. This is also part of the strengths of the algorithm because it can be applied to any natural language without knowing the underlying grammatical structure.

In experiments done in [23], morph-based language models outperformed word-based language models in a number of speech recognition experiments for Finnish, Estonian, and Turkish, all morphologically rich languages. The performance boost was mostly attributed to the better modeling of out-of-vocabulary words.

4.5 Evaluation of Language Models

4.5.1 Word Error Rate

As was stated in the beginning of the chapter, the main goal for language models is to improve the performance of speech recognition. The most straightforward way to measure performance increase for different sets of language models is to run them in a speech recognition system and calculate the word error rate (WER), which is the most common performance measure for automatic speech recognition systems. The WER is calculated for an evaluation transcription (with the total number of words N_r) as the percentage of the number of substituted (S), deleted (D), and inserted (I) words in the output text generated by the speech recognizer.

$$WER = \frac{S + D + I}{N_r} \tag{4.8}$$

A letter error rate (LER) can also be calculated in a similar fashion. In some languages the LER is a more descriptive measure. For example in Finnish the errors made on the word level are usually quite trivial, such as a deleted or inserted letter at the end of the word. The WER is too harsh of a measure in these cases.

Running the speech recognizer separately with different language models and calculating the WER or LER on the evaluation transcription is a good measure when comparing performances between language models. However, it is very time consuming. One recognition run on a big evaluation data set can take several hours to complete. A faster way to measure language model performance is needed.

4.5.2 Perplexity

The perplexity measure (ppl) is one of the most common methods used for evaluating language models. It is a fast technique because the performance measure is calculated over a text. Perplexity is defined as

$$ppl = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(W_i|H)}} \tag{4.9}$$

where N is the number of words in the evaluation text and $P(W_i|H)$ is the probability yielded by the language model. Perplexity is a measure of how well the language model predicts the word sequence in the evaluation text. The perplexity value can also be described as the approximate number of equally probable words the language model has to choose from when predicting the next word in the sequence. This means that the lower the perplexity the better the language model. When evaluating language model performance the absolute perplexity value is usually not so important. The absolute value is dependent on the model but also on the evaluation text. More important is the relative perplexity reduction compared to a baseline model.

In Equation (4.9), the perplexity is assumed to be calculated over a word-based language model. For morph-based models the perplexity expression has to be rewritten as follows [24]:

$$ppl = \sqrt[N]{\prod_{i=1}^{L} \frac{1}{P(W_i|H)}} \tag{4.10}$$

where N is the number of words, and L is the number of tokens, morphs + word boundaries.

The perplexity measure is a much faster method for evaluating language models than what the word-error-rate is. There is however some dispute of how well perplexity reductions correlate with WER reductions. The main source of mismatch is considered to be that perplexity fails to account for the acoustic confusability of words. However in [25], Iyer et. al showed that perplexity reduction correlates well with WER reduction when the evaluation text is from the same domain as the LM training text.

In the article *"Two Decades of Statistical Language Modeling: Where Do We Go from Here?"* [26], R. Rosenfeld suggests as a rule of thumb, that a 5% perplexity reduction does not usually lead to a significant drop in WER, a 10-20% reduction usually leads to some reduction in WER, and 30% reduction is quite significant. For example, in the experiments done by Iyer et. al in [25], a relative perplexity reduction of 23% leads to a 4.5% relative WER drop. It is however good to remember that this correlation does not always apply and it depends highly on the model and evaluation data.

Chapter 5

Language Model Adaptation

5.1 Language Model Adaptation

Natural language is highly variable. The frequencies of different words and word sequences we encounter vary across topic, style and also time.

Illustrating the difference between topics through homophones, the word *"gorilla"* is more likely to occur in an article about tropical wildlife than the word *"guerilla"*. However if we switch the topic to world politics *"guerilla"* is probably more prevalent. Apart from single words, the frequency of specific n-grams can also be greatly elevated in some particular topics. For example, a commentary from a cricket match might have a significantly higher occurrence of the n-gram *"corridor of uncertainty"* compared to other topics.

Different styles of text and speech also influence heavily on n-gram statistics. Comparing a casual conversation with friends with a speech given at an official ceremony, we know that the latter is usually closer to the correct grammatical syntax. Texts also differ in style depending on their purpose. A technical report and an email written to a work colleague might both be just as close to grammatical perfection, but we could still expect less informal language in the email by the use of shorter sentences and more frequent use of first and second person pronouns.

Natural languages evolve all the time. Grammar rules can change, new styles emerge, old words become obsolete and new words and expressions are introduced. This is also seen in n-gram statistics of texts and speeches from different times in history. For example, the frequency of use of some technical terms usually rises and falls with the particular technology. Just consider how often we see the word *"VHS"* being used nowadays compared to 20 years ago.

In the previous chapter it was stated that n-gram language models are trained with large text corpora containing up to several millions of words. The best performing model is usually the one trained with the most amount of data. It may however come as no surprise that model performance suffers, when evaluating n-gram models on text or speech which differ from the training corpus in topic, style, or time of creation. To model phone conversations for example, an n-gram model trained with only two million words from similar conversations performs much better than a model trained with 140 million words from TV and radio news transcripts [27]. In experiments done by R. Rosenfeld in [28], it was also shown when evaluating language models on business related texts that the OOV rate is much lower for models trained with business language data as compared to models trained with texts from other topics. The timeline factor was also investigated, and as could be expected the models trained with more recent material had a lower OOV rate than the models trained with older texts.

Ideally, for each particular type of text or speech, we would like to have a corresponding language model trained with texts from the same domain. Data sparsity is a problem however. The total amount of text data found for some specific topic is usually only a small fraction of the big text corpora which are explicitly used for training general purpose language models. On the other hand, these smaller in-domain models can perform just as well or even better than larger out-of-domain models, as was noted earlier. For optimal performance however, it would be best to use all information sources and combine both the statistics of the bigger, more general background corpus and the smaller, more topic specific adaptation corpus.

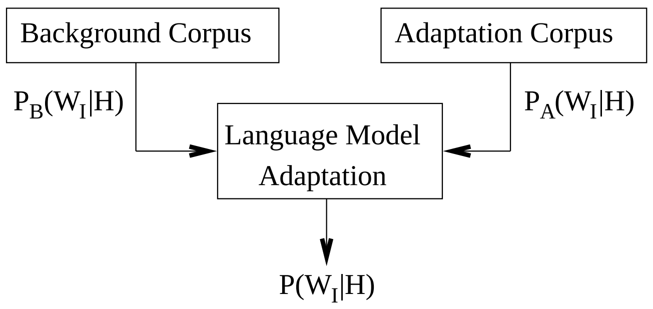

Figure 5.1: General framework for language model adaptation

Combining statistics from text corpora of different domains to yield one language model is called language model adaptation and the general frameworks for it are outlined in Figure 5.1. The method used for combining the different information sources is at the heart of the issue of language model adaptation. Some of the most common methods will be introduced in the next sections.

5.2 Linear Interpolation

Linear interpolation is one of the most straightforward language model adaptation techniques available. The n-gram models $P_A(W|H)$, computed from a small adaptation corpus, and $P_B(W|H)$, computed from a big background corpus, are merged by simply adding together the weighted probabilities of the two different models.

$$P(W|H) = (1-\lambda)P_A(W|H) + \lambda P_B(W|H) \tag{5.1}$$

The interpolation coefficient λ takes a value between 0 and 1. The value of λ is usually a constant and it's estimated beforehand. This is done by minimizing the perplexity when evaluating the model (as a function of λ) on some held-out data (usually a subset of the adaptation corpus). For some special implementations λ can also be a function of the word history *H*.

Linear interpolation can be further generalized from (5.1) to include any number of different language models P_i (i=*1...N*) to be combined into one single model.

$$\begin{aligned} P(W|H) &= \sum_i \lambda_i P_i(W|H) \\ where \quad & \sum_i \lambda_i = 1 \end{aligned} \tag{5.2}$$

In [27], Bellegarda motivates the use of linear interpolation with the assumption that the background model most of the time provides better estimates because of the larger corpus it has been trained on. The smaller adaptation model should give better estimates only for some rarely occurring topic specific words or word sequences. It might however be the other way around in some cases, that the frequency of these idiosyncratic word sequences is higher than the use of general language. Either way which model turns out to be more dominant, linear interpolation will boost high probability n-grams for the less significant model.

In theory, linear interpolation will yield an adapted language model which is at least just as good as the best of the two mixture models. This is of course based on the optimal value of λ and the perplexity minima estimated from the held-out data. In actual speech recognition, there are no guarantees on the performance.

5.3 Log-linear Interpolation

Another straightforward method of combining different sets of n-gram models $P_i(W|H)$ into a single model $P(W|H)$, is log-linear interpolation. Log-linear interpolation was introduced as a language model adaptation method by Klakow in [29]. It can in simple terms be described as linear interpolation in the logarithmic domain.

$$\begin{aligned} P(W|H) &= \frac{1}{Z_\lambda(H)} \prod_i P_i(W|H)^{\lambda_i} \qquad (5.3) \\ where \qquad & Z_\lambda(H) = \sum_W \prod_i P_i(W|H)^{\lambda_i} \end{aligned}$$

In log-linear interpolation the probabilities are merged by multiplication instead of addition. The interpolation weights are also powered to the probability models instead of multiplied. We can now see how this justifies the use of the term log-linear since multiplication and exponentiation correspond to addition and multiplication in the logarithmic domain.

Another difference from linear interpolation is that the product is divided by a normalization term $Z_\lambda(H)$. This is to ensure that the probabilities of all words sum to 1. It is also worth noting that there are no constraints set on the interpolation weights λ_i. They are not required to sum to 1 as in linear interpolation.

The use of log-linear interpolation for combining probability models has a well-founded mathematical basis. The theoretical cornerstone from which log-linear interpolation is derived is the Kullback-Leibler distance.

$$D(P(W)|P_i(W)) = \sum_W P(W) log \frac{P(W)}{P_i(W)} = d_i \qquad (5.4)$$

The Kullback-Leibler distance measures the difference between two probability distributions $P(W)$ and $P_i(W)$. If we have a set of language models $P_i(W)$ $(i=1...N)$ which we want to combine into a single target model $P(W)$, the optimal combination method can be found indirectly when minimizing the Kullback-Leibler distance *$D(P|P_0)$* for an additional model *$P_0(W)$*. Setting the distance measures *d_i (i=1...N)* as constraints and using Lagrangian multipliers a system function *$D_\Gamma(P(W))$* to be minimized can be expressed.

$$D_\Gamma(P(W)) = D(P(W)|P_0(W)) + \sum_i \gamma_i(D(P(W)|P_0(W)) - d_i) \tag{5.5}$$

If we choose *$P_0(W)$* to be a uniform distribution (all words have the same probability), minimize *$D_\Gamma(P(W))$* through partial derivatives, and change *$P(W)$* to a conditional probability *$P(W|H)$*, the target model will get the same expression as equation (5.3). A more detailed mathematical proof can be found in [30].

Log-linear interpolation outperforms linear interpolation in most situations. In adaptation experiments done by Klakow in [29], log-linear interpolation achieved a 12.8% perplexity reduction and a 3.1% WER reduction compared to linear interpolation. A drawback for log-linear interpolation is the normalization term *$Z_\lambda(H)$* that needs to be calculated for all words in the vocabulary. This will greatly increase the computational time it takes to calculate a log-linear interpolation model.

5.4 MDI Adaptation

Linear and log-linear interpolation provide a fairly simple way of adapting a general purpose language model to a more topic specific target model. A clear disadvantage though, is the static way in which the background and adaptation models are combined, relying on an averaged out interpolation weight.

When estimating language models from small adaptation corpora an issue that comes up is reliability. The small size of the adaptation corpus can lead to unreliable estimates especially for longer n-gram contexts. For example, in small adaptation corpora that contain relatively few documents there might be an disproportionate number of occurrences of some particular word contexts which only reflect the style of the writer or a certain subtopic and not give a good estimate of the topic as a whole. It's justified to question the use of the same interpolation weights for all n-grams when combining language models which have been trained on corpora that differ greatly in size.

Ideally, we would like to pick out the most reliable n-gram estimates from the adaptation corpus and get as close as possible to the background model. Using selected features *$\hat{P}_A(S_i)$* as constraints while at the same time minimizing the distance to a given probability model is called minimum discrimination information (MDI) adaptation and was first introduced by Della Pietra et. al. in [31]. The target model P_A is found by minimizing the Kullback-Leibler distance to the background model P_B.

$$P_A = \arg\min_Q \sum_{HW \epsilon V^n} Q(H,W) log \frac{Q(H,W)}{P_B(H,W)} \tag{5.6}$$

The solution depends on how the constraints *$\hat{P}_A(S_i)$* are set. In [32], Federico sets the unigram estimates from the adaptation corpus as constraints and reaches the following closed form solution using the Generalized Iterative Scaling (GIS) algorithm.

$$\begin{aligned} P_A(W|H) \quad &= \frac{1}{Z(H)} \left(\frac{P_A(W)}{P_B(W)} \right)^{\beta} P_B(W|H) \qquad (5.7) \\ where \qquad & Z(H) = \sum_{\hat{W} \epsilon V} P_B(\hat{W}|H) \left(\frac{P_A(\hat{W})}{P_B(\hat{W})} \right)^{\beta} \\ and \qquad & 0 < \beta \leq 1 \end{aligned}$$

This form of MDI adaptation is also known as fast marginal adaptation or unigram rescaling. The parameter β can be set freely between 0 and 1. Just as for log-linear interpolation a normalization term *Z(H)* needs to be calculated. A closer look at equation (5.7), and we notice that it works by simply scaling up the probability for words that are frequent in the adaptation corpus. The unigram estimate is reliable even for small corpora, therein lies the power of unigram rescaling as an adaptation method.

The parameter β is usually set by default to 0.5. In [33], Chen et. al. try to find the optimal β for different corpus sizes. The optimal value of β approaches 1 when the corpus size is very large and is closer to 0 when the adaptation corpus is smaller. This can be interpreted as a reliability factor. When the corpus is larger the unigram estimates are also more reliable and a bigger value can be assigned to β.

In most of the research that has been done on MDI adaptation, the performance is evaluated in comparison to the background model. In experiments done by Kneser et. al. in [34], MDI adaptation was evaluated on two different domains. For the first domain a relative perplexity reduction of 19.5% and a WER reduction of 9.8% was achieved compared to the background model.

For the other domain the perplexity was reduced by 18.3% compared to the background model, but the WER was increased by 4.5%. In similar experiments done by Federico in [32], an average perplexity reduction of 17.8% and a WER reduction of 5.2% was obtained.

A comparison between the performance of MDI adaptation and linear interpolation (and other adaptation techniques) was done by Chen et. al. in [35]. The adapted models were evaluated on two different test sets and compared to a background model. The average perplexity reduction for the MDI adapted model was 7.2% and for linear interpolation 13.2%. The average WER reduction was 1.8% for MDI adaptation and 1.2% for linear interpolation.

Chapter 6

Experiments

6.1 Experimental Setup

The following experiments are conducted in this work. Speech data supplied by each test speaker is divided into test data and adaptation data. The test data is run through the ASR system in a series of different experiments to determine the best performing acoustic and language models. The evaluation of speech recognition performance is done using the word error rate (WER). To get a hint of how the WER and the quality of the recognized text are correlated, reference sentences and speech recognition output with different WERs can be viewed in Appendix A. Example sentences in both English and Finnish are presented.

In the first line of experiments, unadapted baseline acoustic models are evaluated. After that, acoustic model adaptation is applied on the best baseline model with adaptation data from the specific test speaker. The acoustic model adaptation methods that will be evaluated are cMLLR, regression class cMLLR and VTLN. They are in turn evaluated for different amounts of adaptation data. A supervised mode is used for all acoustic model adaptation methods, meaning transcripts are available for all adaptation sentences.

Using the best performing acoustic model (unadapted or adapted) as part of the ASR system, the search for the best language model is started. Adaptation data (law-related texts) is used to adapt a baseline LM using different adaptation methods. The following LM adaptation methods are evaluated: linear interpolation, log-linear interpolation, and MDI adaptation.

6.2 Speech Recognition System

The TKK speech recognition system is used in this work. The system consists of units for feature extraction, acoustic modeling, pronunciation modeling, language modeling, and decoding.

The feature extraction unit divides the incoming speech signal into overlapping segments. Each segment is converted to a 39-dimensional MFCC vector.

The acoustic models are based on a set of continuous density GMM-HMM models. Triphones are used as the basic phonetic units. Each triphone is modeled with a 3-state HMM. The pronunciation of words is modeled in a pronunciation dictionary, or lexicon. A triphone lexicon is used with triphone-based HMMs. For the Finnish language a morph-based lexicon is used instead of a word-based.

N-grams are used for language modeling. The language models are trained with the VariKN language modeling toolkit [22], that is able to estimate variable-length n-grams. Language model adaptation is done with the SRI toolkit [36]. To model the Finnish language, morphs are used instead of words. The Morfessor toolkit [23] is used to generate the optimal morph set based on training data.

Decoding is implemented with a time-synchronous beam search decoder [37] [38].

The TKK speech recognition system also includes tools for training acoustic models and doing acoustic model adaptation (cMLLR and VTLN).

6.3 Significance Test

The same test data set is used to evaluate the different model setups of the ASR system. The measured performance differences need to be checked if they are statistically significant or not. In these experiments, a matched-pair sentence-segment word error test (MAPSSWE) is used to measure statistical significance [41]. The test is especially well suited for comparing ASR systems that recognize continuous speech.

The number of errors made per sentence is compared between two systems. Let N_1^i be the number of errors made in sentence i by system A_1, and N_2^i the number of errors made in sentence i by system A_2. Let Z_i=N_1^i-N_2^i, and i=1,2...,n, where n is the number of sentences. The average difference in the number of errors in a sentence made by the two systems is denoted as μ_z. A natural estimate can be defined as follows:

$$\hat{\mu}_z = \sum_{i=1}^{n} \frac{Z_i}{n} \tag{6.1}$$

The estimate of the variance of the Z_i's is:

$$\sigma_z^2 = \frac{1}{n-1} \sum_{i=1}^{n} (Z_i - u_z)^2 \tag{6.2}$$

The estimate of the variance of μ_z is then

$$\hat{\sigma}_\mu^2 = \frac{\sigma_z^2}{n} \tag{6.3}$$

The distribution W is defined as:

$$W = \frac{\hat{\mu}_z}{\hat{\sigma}_z / \sqrt{n}} \tag{6.4}$$

W is approximately a normal distribution with unit variance if n is large enough. The null hypothesis H_0 is that there is no significant difference between the two systems. In mathematical terms this is equal to μ_z=0. In these experiments H_0 is rejected for significance values over 95%.

6.4 Test Data

English test data consisted of 8 kHz audio recordings from a single test speaker. The recordings were made with the MobiDic client application. The speaker was a native male English speaker with a British accent. He will be referred to as E1. In total, 161 sentences (28 min. of speech) recorded by E1 were evaluated. The content of the dictations were mostly related to contract law.

Unfortunately very little Finnish audio data recorded with MobiDic was available at the time of this thesis project. To simulate the use of MobiDic, a mobile phone was used to make 16 kHz recordings with a dictaphone program. The recordings were transferred to a desktop computer for further evaluation and adaptation with the TKK ASR tools. In total 90 test sentences (23 min. of speech) were recorded by a native male Finnish speaker, referred to as F1. The test sentences were taken from the Finnish law text evaluation set. The only available Finnish audio data recorded with the MobiDic application was a set of six sentences (5 min. of speech) recorded in 8 kHz by a native male Finnish speaker (F2). The content of the these dictations was related to contract law.

Finnish test data also included 30 sentences spoken by F1, F2, and a third native male Finnish speaker, F3. Ten sentences were recorded by each speaker with a headset microphone using a 16 kHz sampling frequency. The sentences were taken from the Finnish law text evaluation set.

6.5 Adaptation Data

Adaptation data was divided into speaker adaptation data and language adaptation data.

Speaker adaptation data was available for speakers E1 and F1. For E1, 140 sentences (27 min. of speech) in total were reserved for adaptation. For F1, 50 sentences (11 min. of speech) were reserved for adaptation.

Language adaptation data was retrieved from different information sources containing law-related material, for example EURLEX (English) and FINLEX (Finnish).

For Finnish, 4.3 million words worth of text were available for LM adaptation. A text consisting of 300 000 words was reserved for a development set, used for fine-tuning adaptation parameters. A text set of 88 000 words was reserved for evaluation purposes.

For English, 1.9 million words were used for LM adaptation. The development set consisted of 30 000 words, and the evaluation set of 10 000 words.

6.6 Results

6.6.1 English

The English speech recognition experiments were conducted on evaluation data from speaker E1. The evaluation set consisted of 161 sentences recorded in 8 kHz digital waveform format.

Three different acoustic models were evaluated at starters to determine the best performing baseline model. The Fisher acoustic model is trained on 180 hours of conversational speech in American English recorded over the telephone [39]. The WSJCAM0 acoustic models are trained on 19 hours of planned speech in British English [40]. The WSJCAM0 D model is trained on speech recorded from a desktop microphone, and the WSJCAM0 H is trained on speech recorded from a headset microphone. The WSJCAM0 models use the BEEP lexicon, a British English pronunciation dictionary, while Fisher uses the corresponding American English CMU lexicon. Both lexicons are 60 000 words in size and they contain exactly the same words. The results of the baseline experiments are presented in Table 6.1.

Table 6.1: Baseline acoustic models

Acoustic model	Fisher	WSJCAM0 D	WSJCAM0 H
WER	79.7%	56.1%	63.3%

The WSJCAM0 D model achieved the best performance with a WER of 56.1%. It was used as the baseline acoustic model in the adaptation experiments. Three acoustic model adaptation methods were tested: global cMLLR, regression class cMLLR, and VTLN. The adaptation methods were tested for different sizes of adaptation data from speaker E1. The results can be seen in Figure 6.1.

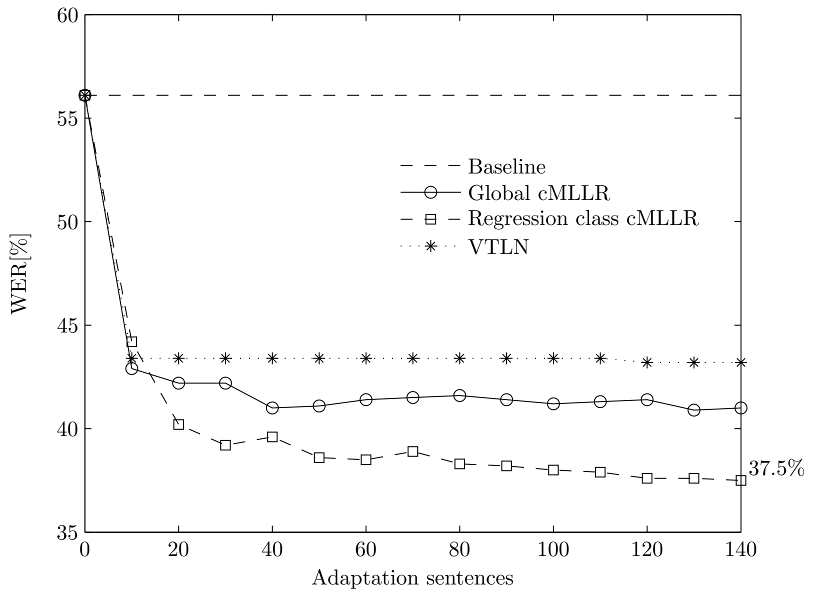

Figure 6.1: Regression class cMLLR achieves the highest performance increase. After 140 adaptation sentences WER has dropped with over 30%.

A noticeable trend for all three adaptation methods is that there is a sharp performance increase for the first ten adaptation sentences (2 min. of speech). After about 20 adaptation sentences (3 min. of speech) the decrease in WER starts to saturate for all adaptation methods. The regression class cMLLR method does give the best overall performance improvement and the WER drops to 37.5% (33.2% relative WER drop) after 140 adaptation sentences (27 min. of speech).

The regression class cMLLR adapted acoustic model trained on 140 adaptation sentences was chosen as the baseline acoustic model for the LM adaptation experiments. Five different n-gram language models were evaluated. The baseline LM is a modified Kneser-Ney smoothed varigram (n=6) model trained on the Gigaword text corpus, which comprises billions of words of text from different English newswire services. A similar law-domain varigram (n=6) model was trained on 1.9 million words of law-related texts. The 60 000 word Gigaword dictionary was used when training both language models. In the law-related training text, 2.6% of the words were not in the dictionary. This is also known as the OOV-rate. The OOV-rate for the Gigaword training text was 1.8%.

Three different adaptation methods were used to combine the Gigaword LM with the law-domain LM. The methods were linear interpolation, log-linear interpolation, and MDI adaptation. The interpolation weights $\lambda_{Gigaword}$ and λ_{Law} were optimized on the 30 000 word development set (2.0% OOV-rate). The optimal values were estimated to $\lambda_{Gigaword}$=0.18 and λ_{Law}=0.82. The same interpolation weights were used for log-linear interpolation. For MDI adaptation the law-domain unigram was used to adapt the Gigaword model. The default β value of 0.5 was used.

As a first point of evaluation the perplexity values were calculated for each model over the 10 000 word law-related evaluation text (2.7% OOV-rate). The relative perplexity changes compared to the Gigaword model were calculated for every model. The results are in Table 6.2.

Table 6.2: LM Perplexity over evaluation text

LM	**Lawtext perplexity**	**Perplexity change**
Gigaword LM	444	0%
Law LM	59	-87%
LIN	44	-90%
LOG-LIN	47	-89%
MDI	221	-50%

We can see that in theory all adapted language models perform well. The perplexity is reduced by 50-90% for all LM adaptation methods.

A similar set of perplexity experiments were also conducted on the 161 transcribed evaluation sentences spoken by speaker E1. The OOV-rate was 1.7%. The results can be seen in Table 6.3.

Table 6.3: LM Perplexity over E1 evaluation set

LM	**E1 evaluation set perplexity**	**Perplexity change**
Gigaword LM	357	0%
Law LM	762	113%
LIN	338	-5%
LOG-LIN	434	21%
MDI	253	-29%

There is perplexity reduction for the linear interpolation model (-5%) and the MDI adapted model (-29%). Perplexity increases quite significantly for the law LM (113%). There is also perplexity increase for the log-linear interpolation model (21%). Comparing the relative perplexity changes with the reductions achieved over the lawtext evaluation set, it can be concluded that the law-domain LM doesn't match that well with the 161 sentences spoken by E1.

Finally, a set of speech recognition experiments were performed on the evaluation sentences, using the TKK ASR decoder with the adapted acoustic model and the five different language models. The results are in Table 6.4.

Table 6.4: Speech recognition experiments on E1 evaluation set

LM	**WER**	**WER change**	**LER**	**LER change**
Gigaword LM	37.5%	0.0%	20.6%	0.0%
Law LM	42.7%	13.9%	22.9%	11.2%
LIN	37.0%	-1.3%	20.1%	-2.4%
LOG-LIN	38.4%	2.4%	20.8%	1.0%
MDI	36.7%	-2.1%	19.9%	-3.4%

The results are in line with the perplexity experiments on the E1 evaluation set. There is performance degradation for the law-domain LM and the log-linear interpolation method. The MDI adapted LM and the linear interpolation method both manage to improve speech recognition performance slightly.

An additional set of linear interpolation models are estimated by varying $\lambda_{Gigaword}$ between 0 and 1. This is to get a hint of where the optimal interpolation coefficient lies. The ASR performance and perplexity (E1 evaluation set) for these models is presented in Figure 6.2.

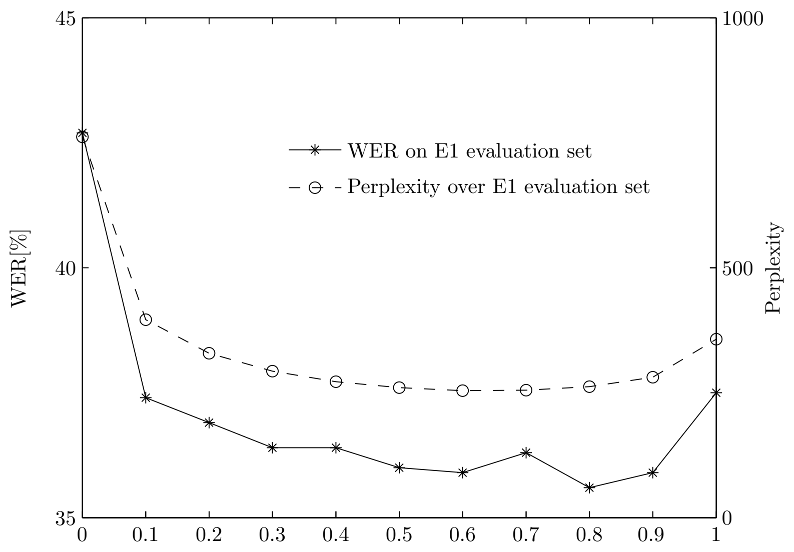

Figure 6.2: Linear interpolated models of Law LM ($\lambda_{Gigaword}$=0) and Gigaword LM ($\lambda_{Gigaword}$=1). The best performing LM has the interpolation coefficient 0.8 and achieves a relative WER reduction of 5.1%.

The best performance is achieved for $\lambda_{Gigaword}$=0.8, with a WER of 35.6% (5.1% relative reduction). This is quite far from $\lambda_{Gigaword}$=0.18, which is the value that was optimized on the development set.

The poor baseline performance on the E1 evaluation set can probably in part be attributed to the difficult language and the lack of proper adaptation data to match it with. A word error rate over 50% is still quite high though, suggesting there are also some significant differences between acoustic model and speech test data. To get an estimate of the level of acoustic mismatch between the E1 evaluation set and WSJCAM0 D, the performance of the acoustic model was evaluated on speech data recorded in identical conditions to that of the training data. The performance of WSJCAM0 D was tested on the WSJCAM0 si_et_2 (20k) evaluation set. The evaluation set consists of 436 sentences read by 14 different speakers. Just as in the previous experiments the Gigaword varigram (n=6) with the 60 000 word dictionary was used for language modeling. The OOV-rate for the si_et_2 (20k) set was 3.7%. The results of the experiments are in Table 6.6.

Table 6.6: WSJCAM0 D evaluation set experiments	
WER	25.5%
LER	11.7%

The WSJCAM0 D model achieves a WER of 25.5% on evaluation data recorded in an identical environment to that of the training data. Comparing this result with the WER of 56.1% on the E1 evaluation set, it's clear that there is big acoustic mismatch between the E1 evaluation set and the WSJCAM0 D model.

6.6.2 Finnish

The first set of Finnish speech recognition experiments were done on evaluation data from speaker F1. The evaluation data consisted of 90 sentences recorded in 16 kHz waveform format.

A baseline model setup was tested at first. The Speecon model was used as the baseline acoustic model. It is trained on 20 hours of both read and spontaneous Finnish speech recorded in 16 kHz format. A morph-based varigram (n=13) model trained on texts from the Kielipankki corpus (180 million words of Finnish text) was used as the baseline LM in the first experiments. The Morfessor program was used to extract an optimal set of morphemes from the training text. A morph lexicon of 19 000 subword units was used for pronunciation and language modeling. The OOV-rate when using a morph-based LM is 0% since the subword segmentations are able to build any word or word form not found in the training text.

The result of the baseline experiment on the F1 evaluation set is in Table 6.7.

Table 6.7: Baseline experiment on F1 evaluation set	
WER	21.2%
LER	4.6%

The Speecon model was then adapted to speaker F1 with different adaptation methods and varying adaptation data sizes. The adaptation methods used were global cMLLR, regression class cMLLR, and VTLN. The acoustic model adaptation results are presented in Figure 6.3.

Global cMLLR, regression class cMLLR and VTLN, all improve speech recognition performance. For both cMLLR-based methods, the highest drop in

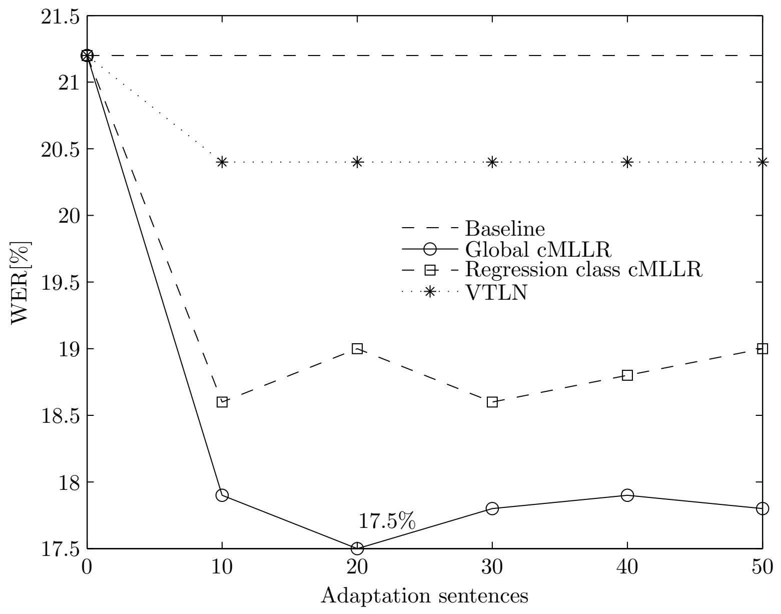

Figure 6.3: The Speecon model was adapted to speaker F1 with the global cMLLR, regression class cMLLR, and VTLN adaptation method. The global cMLLR achieves the highest performance increase.

WER happens during the first 10 adaptation sentences (2 min. of speech). After that, the performance starts to saturate and even degrade. The global cMLLR method achieves the highest performance increase with a WER of 17.5% (17.5% relative WER drop) after 20 adaptation sentences (4 min. of speech).

The best performing adapted acoustic model was chosen as the baseline acoustic model when doing the LM adaptation experiments. The best performing acoustic model was the global cMLLR model adapted with 20 sentences. Four different language models were evaluated alongside the Kielipankki baseline LM. A standalone law-domain varigram LM (n=13) was trained on 4.3 million words of Finnish law texts. A lexicon of 13 000 morphs was created alongside the law-domain LM.

Linear interpolation, log-linear interpolation, and MDI adaptation were used to adapt the baseline LM with the law-domain LM. The interpolation weights $\lambda_{Kielipankki}$ and λ_{Law} were optimized on the 300 000 word development set.

The optimal values were estimated to $\lambda_{Kielipankki}$=0.25 and λ_{Law}=0.75. The same interpolation weights were used for log-linear interpolation. For MDI adaptation the law-domain bi-gram (n=2) was used to adapt the Kielipankki model. The default β value of 0.5 was used.

The language models were first evaluated by calculating the perplexity for each model over a law-related evaluation text consisting of 88 000 words. The relative perplexity changes in comparison to the Kielipankki model were also calculated. The results are viewable in Table 6.8.

Table 6.8: LM Perplexity over law-domain evaluation text

LM	**Lawtext perplexity**	**Perplexity change**
Kielipankki LM	11220	0.0%
Law LM	7762	-30.8%
LIN	6918	-45.0%
LOG-LIN	8511	-24.1%
MDI	21878	95.0%

All adapted LMs give significant perplexity reductions over the law-domain evaluation text, except the MDI adapted model.

Perplexity calculations were also done over the 90 sentences recorded by speaker F1. The results are in Table 6.9.

Table 6.9: LM Perplexity over F1 evaluation set

LM	**F1 evaluation set perplexity**	**Perplexity change**
Kielipankki LM	12303	0.0%
Law LM	11220	-8.8%
LIN	6761	-45.0%
LOG-LIN	8318	-32.4%
MDI	14125	14.8%

The LM perplexities over the F1 evaluation sentences are relatively similar to the LM perplexities over the law-domain evaluation text in Table 6.2. All adapted LMs except the MDI adapted model give significant perplexity reductions.

The language models were finally evaluated in speech recognition experiments. The 90 recorded sentences by F1 were used for evaluation. The global cMLLR adapted acoustic model was used in all the experiments as part of the TKK speech recognizer. The results are in Table 6.10.

Table 6.10: Speech recognition experiments on F1 evaluation set

LM	WER	WER change	LER	LER change
Kielipankki LM	17.5%	0.0%	3.4%	0.0%
Law LM	20.7%	18.3%	4.3%	26.5%
LIN	19.2%	9.7%	3.7%	8.8%
LOG-LIN	20.0%	14.3%	4.2%	23.5%
MDI	18.7%	6.9%	3.6%	5.9%

Despite the perplexity reductions, none of the adapted LMs improved ASR performance on the F1 evaluation set.

In this case there was no clear relationship between perplexity reduction and a lower WER. Probably due to this, the linear interpolation coefficient $\lambda_{Kielipankki}$=0.25 was clearly not an optimal value either. An additional set of linear interpolation models were estimated by varying $\lambda_{Kielipankki}$ between 0 and 1. The WER and perplexity (F1 evaluation set) for these models is presented in Figure 6.4.

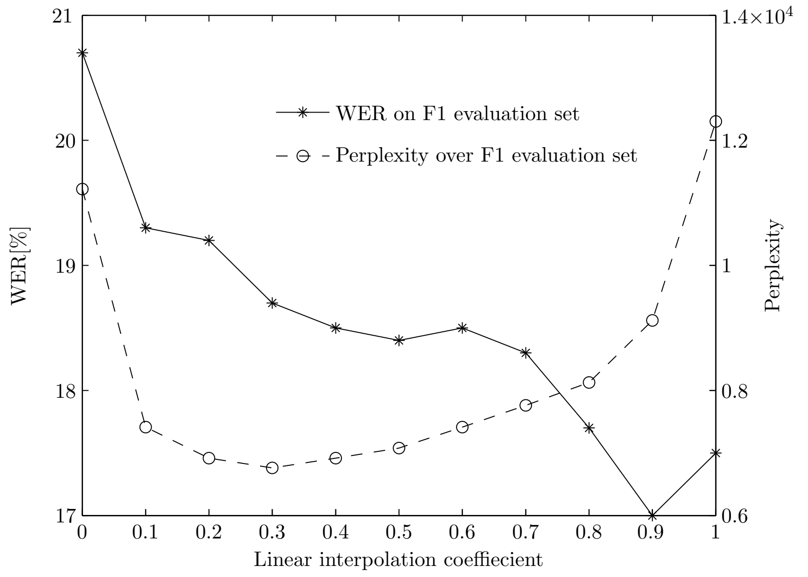

Figure 6.4: Linear interpolated models of Law LM ($\lambda_{Kielipankki}$=0) and Kielipankki LM ($\lambda_{Kielipankki}$=1). The best performing LM has the interpolation coefficient 0.9 and achieves a relative WER reduction of 2.9%.

From the curve in Figure 6.4, it's evident that for optimal performance, the interpolation coefficient should get more weight from the baseline LM instead of the law-domain LM. The best performing linear interpolation model is $\lambda_{Kielipankki}$=0.9 with a WER of 17.0% (2.9% relative reduction). It's also interesting to note the odd relationship between perplexity and WER after $\lambda_{Kielipankki}$=0.3. In contradiction with theory, ASR performance improves although perplexity increases.

The test sentences recorded on the MobiDic client application by F2 were also evaluated. The LM perplexities were calculated over the six test sentences. The results are in Table 6.11.

Table 6.11: LM Perplexity over F2 evaluation set

LM	**F2 evaluation set perplexity**	**Perplexity change**
Kielipankki LM	14125	0.0%
Law LM	15488	9.7%
LIN	9333	-33.9%
LOG-LIN	12023	-14.9%
MDI	17783	25.9%

The perplexities of the linear interpolation LM and log-linear interpolation LM are reduced over the F2 evaluation set. The law-domain LM and MDI adapted LM both get higher perplexity values than the baseline.

The language models were then evaluated in speech recognition performance. As acoustic model in all experiments on the F2 evaluation set, the 8 kHz Speechdat model was used. The model is trained on 8 kHz telephone speech from 4000 speakers. The results are in Table 6.12.

Table 6.12: Speech recognition experiments on F2 evaluation set

LM	**WER**	**WER change**	**LER**	**LER change**
Kielipankki LM	22.7%	0.0%	4.8%	0.0%
Law LM	30.4%	33.9%	7.1%	47.9%
LIN	26.7%	17.6%	5.6%	16.7%
LOG-LIN	24.7%	8.8%	5.5%	14.6%
MDI	22.9%	0.9%	5.0%	4.2%

From these results we can also conclude that there is no performance gain for any adapted LM, despite significant perplexity reductions for the linear interpolation LM and loglinear interpolation LM. The MDI adapted LM, which had the highest perplexity value, achieves the lowest WER of all the adapted models.

The last set of experiments were done on the 30 sentence evaluation set consisting of speech recorded by speakers F1, F2, and F3 (10 sentences/speaker) on a headset microphone in 16 kHz. As usual, perplexities were calculated for each LM over the evaluation sentences. The results are in Table 6.13.

Table 6.13: LM Perplexity over F1/F2/F3 evaluation set

LM	Evaluation set perplexity	Perplexity change
Kielipankki LM	13859	0.0%
Law LM	10594	-23.6%
LIN	6595	-52.4%
LOG-LIN	8252	-40.5%
MDI	16145	16.5%

Results in Table 6.13 follow the general trend. Perplexity drop occurs for all adapted LMs except the MDI model. The highest perplexity drop is achieved by linear interpolation with a reduction of 52.4%.

The speech recognition results are presented in Table 6.14. The unadapted Speecon model was used as acoustic model in all experiments.

Table 6.14: Speech recognition experiments on F1/F2/F3 evaluation set

LM	WER	WER change	LER	LER change
Kielipankki LM	18.3%	0.0%	4.0%	0.0%
Law LM	20.1%	9.8%	4.2%	5.0%
LIN	17.7%	-3.3%	3.6%	-10.0%
LOG-LIN	19.4%	6.0%	4.3%	7.5%
MDI	18.5%	1.2%	4.0%	0.0%

Unlike previous experiments, LM adaptation is successful at improving ASR performance on the F1/F2/F3 evaluation set. Linear interpolation reduces WER with 3.3%. The other methods do not manage to improve performance however. Comparing the absolute results with experiments on the other evaluation sets, there doesn't seem to be much difference for Finnish between speech recorded on mobile phones and headset microphones.

6.7 Analysis

6.7.1 English

In the baseline acoustic model experiments on the E1 evaluation set, WSJCAM0 D (WER 56.1%) outperformed WSJCAMO H (WER 63.3%) and Fisher (WER 79.7%). The poor performance of the Fisher model is not surprising considering it is trained on spontaneous speech in American English, a clear mismatch to the E1 evaluation set. The reason why a model trained on far-field data (WSJCAM0 D) performed better than a model trained on close-talk data (WSJCAM0 H) is likely an effect of the recording condition. The E1 evaluation set was recorded on a mobile phone by holding the device a few centimetres from the mouth.

Acoustic model adaptation improved ASR performance on the E1 evaluation set. After the 10 first adaptation sentences the performance boost over the baseline model was statistically significant (p=0.05) for all three adaptation methods: global cMLLR, regression class cMLLR, and VTLN. However, there was no significant difference between the methods after the 10 first adaptation sentences.

Averaged out over all adaptation sizes (10-140), the performance of the cMLLR-based methods was better than VTLN. Regression class cMLLR gave the best overall performance, gaining a significant performance increase over global cMLLR after 30 adaptation sentences (5 min).

It's also noteworthy that there is no significant drop in WER for global cMLLR after 30 adaptation sentences. For regression class cMLLR there is significant WER drop even after 70 adaptation sentences (13 min). These results adhere with the theory that regression class cMLLR is better at exploiting larger adaptation data sets, through the use of multiple adaptation matrices. Deciding what adaptation technique to use depends on how much adaptation data is available. Global cMLLR gives faster and better performance increase for adaptation data that is under 5 minutes. Regression class cMLLR is slower but can give better performance if larger amounts of adaptation data are available.

Language model adaptation did not give any further performance increase in this task. Log-linear interpolation and the law LM both performed worse than the Gigaword baseline model. Linear interpolation and MDI both achieved a WER that was lower than the baseline (1.3% and 2.1% relative WER drop), but the improvements were not statistically significant.

One of the reasons for the meagre results of LM adaptation is the apparent linguistic mismatch between the E1 evaluation sentences and the law-related adaptation texts. This is seen when comparing the LM perplexities over the lawtext evaluation set and the E1 evaluation text (Tables 6.8 and 6.9). For the lawtext evaluation set the relative perplexity reductions are significantly high (50-90%) for all adapted LMs. For the E1 evaluation text only linear interpolation and MDI reduce the perplexity, but the reductions are small (5% and 29%).

Finding LM adaptation texts that better match the intended dictation topic is the ideal solution. In this case, the dictations were mainly related to contract law. Language adaptation data on the other hand, was mainly related to case law. An LM trained only on contract law texts might have performed better. Finding large enough in-domain text sets for training a standalone contract law LM is an additional challenge. If data sparsity becomes an issue, MDI adaptation could provide an optimal solution since it has been known to offer performance increase even for smaller adaptation texts. In previous MDI experiments, adaptation data consisting of only a few thousand words has been enough to provide significant WER reductions [34] [42]. In [43], Vergyri et al. propose a method of LM adaptation where the adaptation data is taken directly from the recognition output. A feedback mechanism is also implemented where the user is given the chance to partially correct the output, mainly focusing on important topic-specific words. This method achieves a 13% relative WER reduction over the unadapted system. How to implement this feedback function on a mobile device in a user-friendly fashion is another problem though. For future work a considerable approach would be to collect written or manually transcribed work-related texts from the user and use these texts to directly adapt the background LM.

In conclusion, it can be said that language modeling was far from optimal in this task. Furthermore, the OOV-rate for the E1 evaluation set was 1.7% and it contributes to the WER with roughly double that percentage (3.4%). However, a far bigger error source seems to be the acoustic mismatch between the baseline acoustic model and the speech recordings of the E1 evaluation set. This fact is made evident when comparing the unadapted baseline performance between the E1 evaluation set (56.1%) and the WSJCAM0 D evaluation set (25.5%). It is not yet clear what is behind this huge gap in performance. There are a number of factors which may degrade ASR performance when using a mobile phone to record speech. Previous research has shown that there is significant performance degradation when using different types of microphones for training and testing [44]. The distance between the speaker's mouth and the microphone also affects performance. Usually close-talk data tested on a close-talk model gives better results than far-field data tested on a far-field model.

The much lower WERs of the Finnish experiments are partly explained by the fact that Finnish is phonetically an easier language to recognize than English. Based on ASR experiments on the F1/F2/F3 evaluation set (speech recorded on headset microphones), there didn't seem to be much performance gain coming from recording speech on a normal close-talk microphone instead of a mobile phone.

Future research should focus on finding out whether there really are any significant performance degrading factors related to recording speech on a mobile phone. More detailed and controlled set of experiments needs to be done. To rule out speaker, language, and recording environment variability, speech data should be recorded simultaneously on both mobile phone and normal headset microphone. Speech recognition performance is then evaluated separately for each set of recorded data.

Another way of spotting performance degrading factors is to modify some recording parameters. Changing the sampling frequency from 8 kHz to 16 kHz could by itself enhance performance. Trying out different recording methods, for example holding the phone at different distances while recording, could also be an interesting approach to pinpoint any factors which may boost or degrade performance. It would also be of interest to compare different mobile phone models to see if there are any big differences in microphone quality and suitability for ASR applications.

6.7.2 Finnish

Global cMLLR and regression class cMLLR gave statistically significant (p=0.05) WER reductions on the F1 evaluation set. The performance enhancement was achieved for both methods after only 10 adaptation sentences (2 min.). VTLN also improved performance but the WER reduction was not significant.

Performance of global cMLLR was significantly better than regression class cMLLR averaged out over all adaptation sizes (10-50). Global cMLLR achieved the lowest WER of 17.5% after 20 adaptation sentences (4 min.). On average, ASR performance saturated for all methods after 10 adaptation sentences. Regression class cMLLR was not able to exploit the increased size of adaptation data through multiple adaptation matrices, as it did for English speech data.

It is not clear whether this is related to the quality of adaptation data or to innate acoustic differences between Finnish and English.

Language model adaptation was unsuccessful at improving ASR performance on both the F1 and F2 evaluation set. Over the F1 evaluation set, the perplexities dropped significantly for the law-domain LM, linear interpolation LM and log-linear interpolation LM, but neither of these models got a lower WER than the baseline LM.

The perplexity values over the F2 evaluation set were also in stark contradiction with actual ASR performance. Linear interpolation LM and log-linear interpolation LM had significant drop in perplexity compared to the baseline LM, but ASR performance was degraded for both models with significantly higher WERs. Another contradiction was that the MDI adapted LM came closest to the baseline LM in performance with a WER of 22.9%, despite having a perplexity increase of 25.9%.

The failure of LM adaptation on the F1 evaluation set is somewhat unexpected given that the evaluation sentences are from the same source as the adaptation texts. The law-domain LM does attain perplexity reductions over both the E1 evaluation sentences (8.1% relative drop) and the law-domain evaluation text (30.8% relative drop), but in the ASR task the WER is 16.8% higher than the baseline LM. A major challenge for Finnish LM adaptation in this task seems to be perplexity, and it's poor ability to predict ASR performance for morph-based language models. This is exemplified in Figure 6.4, where ASR performance is evaluated for linear interpolation coefficients between $\lambda_{Kielipankki}$=0.0 and $\lambda_{Kielipankki}$=1.0. An optimal linear interpolation coefficient was estimated to be $\lambda_{Kielipankki}$=0.25, based on perplexity over a development set. However, optimal ASR performance is achieved for linear interpolation coefficient $\lambda_{Kielipankki}$=0.9. Examining both the WER and perplexity plots in Figure 6.4, it's also noticeable that the curves move in completely different directions after $\lambda_{Kielipankki}$=0.3. The performance improves even though perplexity increases. Comparing this with the corresponding experiment of English linear interpolated LMs (word-based) in Figure 6.2, there is much more correlation between the WER and perplexity curve.

The experiments in this work cast a doubt over whether perplexity is a suitable measure for predicting ASR performance of statistical morph-based language models. Most previous research done on morph-based LMs have usually concentrated on comparing the performance (perplexity, OOV rate, WER) to word-based LMs. Morph-based models are usually assigned higher perplexities than word-based models but achieve better ASR performance due to a lower OOV rate. Not much research has focused on comparing perplexity values between morphed-based LMs and relating this to actual ASR performance. In [24], Broman studied the combination of morph-based LMs and the models were evaluated by both perplexity and WER. The results are similar to this work in that there is very little correlation between perplexity and WER. It is possible that the true robustness of a morph-based LM is missed when only calculating the probability for the most likely word segmentation. Linear interpolation did however improve ASR performance on the F1/F2/F3 evaluation set. This shows that evaluation data can also have a significant effect on results.

The log-linear interpolation and MDI adaptation methods also gave disappointing results in this work. The less than satisfactory performance of their part can in some regard also be related to the properties of morph-based language models. Since the idea of MDI adaptation is to elevate the probability of words that occur frequently in the adaptation texts, a bi-gram(n=2) was used to rescale the baseline LM, to take in to account the morph-based structure of the LM. It could be justified to use even longer contexts in future research projects, because many words are segmented in to more than two morphemes. The main reason log-linear interpolation underperformed is probably because of the decision to use the same interpolation coefficient as was estimated for linear interpolation.

Unfortunately there were only six evaluation sentences available (F2 evaluation set) that were not taken from the same source as the adaptation texts. It's therefore difficult to judge how well the Finnish adaptation texts match with texts dictated by real MobiDic target users.

Chapter 7

Conclusions

7.1 Conclusions

In this Master's thesis work we have studied the integration of the mobile dictation service MobiDic with the TKK speech recognition system. Evaluation data consisted of speech recorded on mobile phones. English speech data was recorded in 8 kHz waveform format with the MobiDic client application. Finnish speech data was recorded in 16 kHz format with a third-party dictation program. The speech topics for both languages were law-related.

Because of the special characteristics of the evaluation data the main focus in this work was on model adaptation.

Acoustic model adaptation modifies the acoustic model of the ASR system to better fit the testing conditions, such as recording method, environmental noise, and speaker voice characteristics. Three acoustic model adaptation methods were evaluated in this work: global cMLLR, regression class cMLLR and VTLN.

Language model adaptation adapts a background language model to a specific topic or style. Linear interpolation, log-linear interpolation, and MDI adaptation were used in this work to adapt a background language model to law-related text.

Acoustic model adaptation improved ASR performance on English evaluation data. Global cMLLR, regression class cMLLR, and VTLN gave about the same performance increase (22% relative WER drop) after 10 adaptation sentences (2 min). After 30 adaptation sentences (5 min) regression class cM-

LLR gave significantly higher performance increase over global cMLLR and VTLN. Performance improvement for global cMLLR and VTLN saturated after 30 adaptation sentences while regression class cMLLR gave significant performance increase even after 70 adaptation sentences (13 min). The use of multiple adaptation matrices in regression class cMLLR makes it a better method for exploiting larger sets of adaptation data.

The poor baseline performance (WER 56.1%) on English evaluation data is troubling. Acoustic model adaptation successfully dropped WER with over 30% but it might still be too high for a commercial application. Future research should concentrate on finding the major factors that cause this big performance flop. Trying out and comparing different recording settings is a possible approach to see if there are any acoustic model mismatch issues related to recording speech on a mobile phone. Another approach would be to train a completely new acoustic model with speech data recorded on mobile phones. This would of course require the recording of several hours of speech data from different speakers.

Language model adaptation did not give any significant improvements in ASR performance on English evaluation data. English language adaptation data did not match that well with the language of the evaluation data. Gathering more appropriate adaptation texts to adapt the background language model is the natural solution. If data sparsity becomes an issue, selecting a smaller amount of texts from a narrower but more accurate domain could give better results with MDI adaptation than using larger text sets from a wider range of topics.

Global cMLLR and regression class cMLLR improved ASR performance on Finnish evaluation data. VTLN did not give any significant improvements. The performance increase after 10 adaptation sentences (2 min) was around 16% relative WER drop for global cMLLR and 12% relative WER drop for regression class cMLLR. Performance did not significantly improve for either adaptation method after the first 10 adaptation sentences. Regression class cMLLR was not able to improve performance for increased adaptation data size, as it did on English speech data.

Language model adaptation was unsuccessful at further improving ASR performance on Finnish evaluation data. Adaptation data did match evaluation data based on perplexity results, but reduced perplexity did not lead to improved ASR performance. MDI adaptation gave the worst performance of all the LM adaptation methods. It is likely longer contexts are needed for the rescaling operation, taking into account how single words are split into subwords in a morph-based language model.

In summary, acoustic model adaptation is the best guarantee for improving ASR performance on law-related speech recorded on a mobile phone. Global cMLLR gives the fastest improvements with relative WER reductions of 15-22% after only 2 minutes of speech. If adaptation data sets of over 10 minutes are available it's also worth trying out regression class cMLLR. Language model adaptation was a tricky affair in this project. The main challenge lies in finding adaptation data that will match the intended speech topics of the target users.

Appendix A

Speech Recognition Output

A.1 English

REF = Reference sentence.

OUT_U = Speech recognition output with the unadapted WSJCAM0 D model as acoustic model.

OUT_A = Speech recognition output with the regression class cMLLR adapted model as acoustic model.

REF: Hopefully Y would attempt to argue that the license is perpetual and it can therefore use the application even after the termination of the contract.
OUT_U: I have a clue why would it have to argue that the licenses perpetual would impress or use the operation even after termination to. (**WER 72.0%**)
OUT_A: Puzzling why would attempt to argue that the licenses perpetual look of therefore use the application even after termination of the. (**WER 36.0%**)

REF: At the very least there should be some limitation wording requiring Y to consult with X prior to such modification of license terms or removal of the software and the chance for X to modify the license terms or software itself.
OUT_U: For starters are released version drew some irritation wording requiring white to cancel the debts prior to such modification of brass instruments or removal of the software and each article X to modify the last trance also of worries. (**WER 53.7%**)
OUT_A: At the very least there should be some limitation wording requiring white to cancel the debts prior to such modification of license terms or removal of the software and the chance for decks to modify the license terms

all software (**WER 22.0%**)

REF: Perhaps the most important schedule for the client's purposes is schedule three C which relates to products for FDA approval and reimbursement authorization.
OUT_U: North platte emerge a important ritual were for the client's presence the show draws on Reid's shoe will which rose two products for FDA approval all federal reimbursement authorization. (**WER 78.3%**)
OUT_A: Perhaps the most important ritual for the clients purchased the shuttle's reads see her which related products for FDA approval and reimbursement authorization. (**WER 43.5%**)

REF: Achieving control of the concern requires control over its constituent assets if these assets are owned by a company either the assets must be acquired from that company or control must be acquired of the company itself.
OUT_U: R which is in control of the concern acquired control of the redskins featuring a search if these assets around by companies buy the assets must be acquired from Atlanta braves or control must be required over the companies. (**WER 64.4%**)
OUT_A: But she's in control of the concern requires control over its constituent assets if these assets around by companies buy the assets must be acquired from a company or control must be required over the companies. (**WER 44.4%**)

A.2 Finnish

REF = Reference sentence.

OUT_U = Speech recognition output with the unadapted Speecon model as acoustic model.

OUT_A = Speech recognition output with the global cMLLR adapted model as acoustic model.

REF: Artiklan toisessa kohdassa kielletään sijoittautumisoikeutta koskevat uudet rajoitukset jotka perustuvat jäsenvaltioiden kansalaisten syrjintään.
OUT_U: Artiklan j a toisessa kohdassa kielletään nyt sijoittautumisoikeutta koskevat uudet rajoitukset jotka perustuvat jäsenvaltioiden kansalaisten syrjintä. (**WER 30.8%**)
OUT_A: Artiklan ja toisessa kohdassa kielletään sijoittautumisoikeutta koskevat uudet rajoitukset jotka perustuvat jäsenvaltioiden kansalaisten syrjintää.

(**WER 15.4%**)

REF: Alioikeus katsoi että kantajia oli syrjitty viranhaussa sukupuolen perusteella ja esitti lisäksi ettei se voinut Saksan siviililain nojalla hyväksyä kanteita muutoin kuin matkakulujen korvausvaatimuksen osalta.
OUT_U: Alioikeus katsoi että kantajia on syrjitty viranhaussa sukupuolen perusteella tt esitti lisäksi ettei se voinut Saksan siviililain nojalla yöksi ja b kanteita muutoin kuin matkakulujen korvausvaatimuksen osalta. (**WER 20.0%**)
OUT_A: Alioikeus katsoi että kantajia on syrjitty viranhaussa sukupuolen perusteella ja esitti lisäksi ettei se voinut Saksan siviililain nojalla hyväksi ja kanteita muutoin kuin matkakulujen korvausvaatimuksen osalta. (**WER 12.0%**)

REF: Tapauksesta ei ilmene että satamatyö on yleisiin taloudellisiin tarkoituksiin liittyvää palvelua joka erottaa sen erityisesti muista talouselämän toiminnoista tai että etyn perustamissopimuksen noudattaminen estää yritystä täyttämästä velvollisuuksiaan.
OUT_U: Tapauksesta ei ilmene että satamatie on yleisiin taloudellisiin tarkoituksiin liittyvää palvelua joka erottaa sen erityisesti muista talouselämän toiminnoista p t tai että ehtii perustamissopimuksen noudattaminen kestää yritystä täyttämästä velvollisuuksia. (**WER 22.2%**)
OUT_A: Tapauksesta ei ilmene että satamatie on yleisiin taloudellisiin tarkoituksiin liittyvää palvelua joka erottaa sen erityisesti muista talouselämän toiminnoista tai että yhtiön perustamissopimuksen noudattaminen estää yritystä täyttämästä velvollisuuksia. (**WER 11.1%**)

REF: Ainoa seikka joka poikkeaa tästä puhtaasti kansallisesta taustasta on se että Werner asuu toisessa jäsenvaltiossa kuin se jossa hän on töissä.
OUT_U: Ainoa seikka joka poikkeaa tästä puhtaasti kansallisesta taustasta on se että Wärner asuun toisessa jäsenvaltiossa kun se jossa hän antoi pelissä. (**WER 23.8%**)
OUT_A: Ainoa seikka joka poikia tästä puhtaasti kansallisesta taustasta on se että Wärner asuun toisessa jäsenvaltiossa kuin se jossa hän antoi töissä. (**WER 19.0%**)

Bibliography

[1] B.H. Juang and L. R. Rabiner. Automatic Speech Recognition - A Brief History of the Technology Development. Encyclopedia of Language & Linguistics (Second Edition), pp. 806-819, Elsevier, 2006.

[2] M. Turunen, A. Melto, A. Kainulainen, and J. Hakulinen. Mobidic - A Mobile Dictation and Notetaking Application. Proceedings of Interspeech 2008, Brisbane, Australia, September 2008.

[3] Z. H. Tan and I. Varga. Network, Distributed and Embedded Speech Recogntion: An Overview. Automatic Speech Recognition on Mobile Devices and over Communication Networks, pp. 1-23, Springer, 2008.

[4] J. W. Picone. Signal Modeling Techniques in Speech Recognition. Proceedings of the IEEE, Vol. 81, No. 9, pp. 1215-1247, September 1993.

[5] S. B. Davis and P. Mermelstein. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuosly Spoken Sentences. IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-8, No. 4, August 1980.

[6] X. Huang, A. Acero, and H-W Hon. Spoken Language Processing: A Guide to Theory, Algorithm, and System Development, pp. 92-98 ,Prentice Hall, 2001.

[7] X. Huang, A. Acero, and H-W Hon. Spoken Language Processing: A Guide to Theory, Algorithm, and System Development, pp. 170-176 ,Prentice Hall, 2001.

[8] T. Hirsimäki. A Decoder for Large-vocabulary Continuous Speech Recognition. Master's Thesis. Helsinki University of Technology. Department of Computer Science and Engineering. 2002.

[9] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, Vol. 77, No. 2, February 1989.

[10] L. Rodgers. Dynamic Programming Search for Continuous Speech Recognition. IEEE Signal Processing Magazine, pp. 64-83, September 1999.

[11] D. Merino. Robustness in Language and Speech Technology, pp. 47-100, Kluwer Academic Publishers, 2002.

[12] P. C. Woodland. Speaker Adaptation for Continuous Density HMMs: A Review. ITRW on Adaptation Methods for Speech Automatic Recognition, pp. 11-18, 2001.

[13] C. J. Leggetter and P. C. Woodland. Flexible Speaker Adaptation Using Maximum Likelihood Linear Regression. Proceedings of ARPA Spoken Language Technology Workshop, pp. 104-109, 1995.

[14] E. Eide and H. Gish. A Parametric Approach to Vocal Tract Length Normalization. IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 346-349, May 1996.

[15] P. Zhan and A. Waibel. Vocal Tract Length Normalization for Large Vocabulary Continuous Speech Recognition. Technical Report, CMU-CS-97-148, School of Computer Science, Carnegie Mellon University, May 1997.

[16] M. Pitz and H. Ney. Vocal Tract Normalization Equals Linear Transformation in Cepstral Space. IEEE Transactions on Speech and Audio Processing, vol. 13, no. 5, September 2005.

[17] L. F. Uebel and P. C. Woodland. An Investigation into Vocal Tract Length Normalisation. Sixth European Conference on Speech Communication and Technology (EUROSPEECH '99), Budapest, Hungary, September 1999.

[18] H. Ney, U. Essen and R. Kneser. On Structuring Probabilistic Dependences in Stochastic Language Modeling. Computer, Speech and Language. Vol. 8. pp. 1-38, 1994.

[19] R. Kneser and H. Ney. Improved Backing-Off for M-gram Language Modeling. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing. Vol. 1, pp. 181-184, 1995.

[20] S. F. Chen and J. Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Computer Science Group, Harvard University, August 1998.

[21] I. J. Good. The Population Frequencies of Species and the Estimation of Population Parameters. Biometrika, Vol. 40, No. 3, pp. 237-264, December 1953.

[22] V. Siivola, T. Hirsimäki and S. Virpioja. On Growing and Pruning Kneser-Ney Smoothed N-Gram Models. IEEE Transactions on Audio, Speech, and Language Processing, Vol. 15, No.5, July 2007.

[23] M. Creutz, T. Hirsimäki, M. Kurimo, A. Puurula, J. Pylkkönen, V. Siivola and M. Varjokallio. Morph-Based Speech Recognition and Modeling of Out-of-Vocabulary Words Across Languages. ACM Transactions on Speech and Language Processing, Vol. 5, No. 1, Article 3, December 2007.

[24] S. Broman. Combination Methods for Language Models in Speech Recognition. Master's Thesis, Department of Computer and Information Science, Helsinki University of Technology, April 2005.

[25] R. Iyer, M. Ostendorf, and M.Meteer. Analyzing and Predicting Language Model Improvements. Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, 1997.

[26] R. Rosenfeld. Two Decades of Statistical Language Modeling: Where Do We Go from Here? Proceedings of the IEEE, Vol. 88, No. 8, August 2000.

[27] J. R. Bellegarda. Statistical Language Model Adaptation: Review and Perspectives. Speech Communication, Vol. 42, pp. 93-108, 2004.

[28] R. Rosenfeld. Optmizing Lexical and N-gram Coverage Via Judicious Use of Linguistic Data. Proceedings of 1995 European Conference on Speech Communication and Technology, Madrid, pp. 1763-1766, September 1995.

[29] D. Klakow. Log-linear Interpolation of Language Models. Proceedings of 5th International Conference on Spoken Language Processing, Sydney, Vol. 5, pp. 1695-1699, December 1998.

[30] A. Gutkin. Log-Linear Interpolation of Language Models. Master's Thesis, pp. 29-31. University of Cambridge, November 2000.

[31] S. Della Pietra, V. Della Pietra, R. L. Mercer, S. Roukos. Adaptive Language Modeling Using Minimum Discriminant Estimation. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, San Fransisco, pp. I-633-636, March 1992.

[32] M. Federico. Efficient Language Model Adaptation Through MDI Estimation. Proceedings of Eurospeech, Budapest, Vol. 4, pp. 1583-1586, 1999.

[33] L. Chen, J. Gauvain, L. Lamel, and G. Adda. Unsupervised Language Model Adaptation for Broadcast News. Proceedings of International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. 220-223, 2003.

[34] R. Kneser, J. Peters, and D. Klakow. Language Model Adaptation Using Dynamic Marginals. Proceedings of Eurospeech, Rhodes, Vol. 4, pp. 1971-1974, 1997.

[35] L. Chen, J. Gauvain, L. Lamel, and G. Adda. Dynamic Language Modeling for Broadcast News. Proceedings of International Conference on Spoken Language Processing, Jeju Island, Korea, pp. 1281-1284, October 2004.

[36] A. Stolcke. SRILM - An Extensible Language Modeling Toolkit. Proceedings of International Conference on Spoken Language Processing, Denver, Colorado, September 2002.

[37] J. Pylkkönen. An Efficient One-Pass Decoder for Finnish Large Vocabulary Continuous Speech Recognition. Proceedings of the 2nd Baltic Conference on Human Language Technologies, pp. 167-172, Tallinn, Estonia, April 2005.

[38] J. Pylkkönen. New Pruning Criteria for Efficient Decoding. Proceedings of the 9th European Conference on Speech Communication and Technology, pp. 581-584, Lisboa, Portugal, September 2005.

[39] C. Cieri, D. Miller, and K. Walker. The Fisher Corpus: a Resource for the Next Generations of Speech-to-Text. Proceedings of the Language Resources and Evaluation Conference, Lisbon, Portugal, June 2004.

[40] T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals. WSJCAM0: A British English Speech Corpus for Large Vocabulary Continuous Speech Recognition. International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 81-84, May 1995.

[41] L. Gillick and S. Cox. Some Statistical Issues in the Comparison of Speech Recognition Algorithms. Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing, pp.532-535, Glasgow, 1989.

[42] P. S. Rao, D. Monkowski, and S. Roukos. Language Model Adaptation via Minimum Discrimination Information. IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 161-164, Detroit, May 1995.

[43] D. Vergyri, A. Stolcke and G. Tur. Exploiting User Feedback for Language Model Adaptation in Meeting Recognition. Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 0, pp. 4737-4740, 2009.

[44] J-C. Junqua. Sources of Variability and Distortion in the Communication Process. Robust Speech Recognition in Embedded Systems and PC Applications, Chapter 1, pp. 1-31, Springer, 2000.