Aalto University
School of Electrical Engineering
Degree Programme in Communications Engineering

Katri Leino

Maximum A Posteriori for Acoustic Model Adaptation in Automatic Speech Recognition

Master's Thesis
Espoo, September 28, 2015

Supervisor: Professor Mikko Kurimo, Aalto University
Advisor: Seppo Enarvi Lic.Sc. (Tech.)

Aalto University
School of Electrical Engineering
Degree Programme in Communications Engineering

ABSTRACT OF
MASTER'S THESIS

Author:	Katri Leino		
Title:	Maximum A Posteriori for Acoustic Model Adaptation in Automatic Speech Recognition		
Date:	September 28, 2015	**Pages:**	65
Major:	Signal Processing	**Code:**	S3013
Supervisor:	Professor Mikko Kurimo		
Advisor:	Seppo Enarvi Lic.Sc. (Tech.)		

The purpose of the acoustic model in Automatic Speech Recognition system is to model the acoustic properties of the speech. Speech, however, has a lot of internal variation making development of a general acoustic model for all purposes an extremely difficult. Adaptation is used to tune the general acoustic models into a specific task, in order to improve the performance of the system.

Maximum A Posteriori (MAP) adaptation is one of the most common acoustic model adaptation techniques in the speech recognition. MAP adaptation scheme in AaltoASR, Automatic Speech Recognition system of Aalto University, was implemented for this thesis. Implementation was tested with speaker adaptation and compared with constrained Maximum Likelihood Linear Regression (MLLR) adaptation to confirm that implementation functions properly. Results were the same as in previous studies, thus it was concluded that implementation is function correctly. Constrained MLLR adaptation performs better when the adaptation set is less than 10 minutes, otherwise MAP adaptation is superior.

MAP implementation has other uses besides the adaptation. It successfully reduced the size of the acoustic model while improving the performance. MAP was also used to adapt colloquial language by giving more weight to the chosen corpus after Maximum Likelihood or discriminative training.

Keywords:	automatic speech recognition, adaptation, MAP, acoustic model
Language:	English

Aalto-yliopisto
Sähkötekniikan korkeakoulu
Tietoliikennetekniikan koulutusohjelma

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Katri Leino

Työn nimi:
Akustisen mallin MAP adaptointi Automaattisessa Puheentunnistuksessa

Päiväys:	28. syyskuuta 2015	**Sivumäärä:**	65
Pääaine:	Signaalinkäsittely	**Koodi:**	S3013
Valvoja:	Professori Mikko Kurimo		
Ohjaaja:	Tekniikan lisensiaatti Seppo Enarvi		

Puheentunnistimen akustisella mallilla mallinnetaan puheen akustisia ominaisuuksia. Puhetta on kuitenkin monentyylistä ja puhe vaihtelee jopa puhujittain suuresti. Akustisen mallin täytyykin mallintaa puhetta laaja-alaisesti toimiakseen tyydyttävästi arkisissa olosuhteissa. Kaikkiin tilanteisiin soveltuvan akustisen mallin opettaminen ei kuitenkaan ole käytännössä mahdollista. Tästä syystä akustisia malleja viritetään tiettyihin olosuhteisiin esimerkiksi adaptaatiolla.

Yksi yleisimmistä adaptaatiomenetelmistä on Maximum A Posteriori (MAP) adaptaatio. Tässä työssä esitellään MAP adaptaation implementoiti AaltoASR puheentunnistusjärjestelmään, ja tutkitaan mihin tarkoituksiin adaptaatiota voidaan soveltaa.

MAP adaptaatiota verrattiin Constrained Maximum Likelihood Linear Regression (CMLLR) -adaptaatioon puhuja-adaptaatiokokeessa implementaation toimivuuden varmistamiseksi. Todettiin, että CMLLR adaptaatio suoriutuu paremmin, jos adaptointiaineiston määrä on alle 10 minuuttia. Aineiston ollessa yli 10 minuuttia MAP adaptaatio on puolestaan soveltuvampi valinta, sillä MAP hyötyy adaptointiaineiston kasvusta enemmän kuin CMLLR. Tulokset vastaavat aikaisempia tutkimuksia, joissa MAP ja CMLLR adaptaatiota on verrattu keskenään.

Lisäksi huomattiin, että MAP implementointia voidaan käyttää myös akustisen mallin koon pienentämiseen sekä painottamaan tiettyä osaa opetusaineistosta tavallisen Maximum Likelihood tai diskriminatiivisen opetuksen jälkeen. Aineiston painottamismenetelmää testattin puhekielen adaptoimiseen.

Asiasanat:	automaattinen puheentunnistus, adaptointi, MAP, akustinen malli
Kieli:	Englanti

Acknowledgements

This thesis was done in the department Signal Processing and acoustics in Aalto University and lasted from June 2014 to June 2015. I wish to thank my supervisor Prof. Mikko Kurimo and advisor Seppo Enarvi for the patience and good advises during the making process. Thanks also to other colleagues for their help and new ideas and Tri for proofreading. Big thanks to Triton for providing their servers for the research. Finally, I would like to thank my family and friends for the support you have been giving me in the past year.

Espoo, September 28, 2015

Katri Leino

Abbreviations and Acronyms

ASR	Automatic Speech Recognition
CMFCC	Constrained Mel-frequency Cepstral Coefficient
DNN	Deep Neural Network
EM	Expectation Maximization
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
LER	Letter Error Rate
LVCSR	Large Vocabulary Continuous Speech Recognition
MAP	Maximum a Posteriori
MFCC	Mel-frequency Cepstral Coefficient
ML	Maximum Likelihood
MLLR	Maximum Likelihood Linear Regression
NN	Neural Network
OOV	Out-Of-Vocabulary
SD	Speaker Dependent
SI	Speaker Independent
WER	Word Error Rate

Contents

Chapter 1

Introduction

Speech is the most natural and easiest way for humans to communicate and relay information. However, recognizing and understanding speech is actually an extremely complex task even though it may be hard to believe from a human's point of view. Automatic Speech Recognition (ASR) has developed gradually during 60 years of research. In the past practical applications were used mainly in companies of specific fields such as in health care, military and call centers. Only during the last decade has speech recognition become more commonly available via smart phones and other Internet connected devices. Speech recognition requires a lot of computation capacity which was not available in the past and even today state-of-the-art systems are too heavy for user devices. Internet and cloud computing have made it possible to use computers and mobile devices as an interface for speech recognition systems, while the actual recognition is processed by servers of high capacity.

Speech recognition performs almost perfectly in well-controlled conditions, but in more challenging acoustic conditions performance is still far from perfect. Recognition is a difficult task because of linguistic and acoustic complexity of the speech. Humans recognize speech naturally and seemingly easily. Only when learning a new language do we realize the difficulties within and the level of proficiency needed to use it efficiently.

We all have been in situations where it is difficult to make sense what is being said because of, e.g., background noise or a strong accent. But we can get used to these problems to some extent. ASR systems have essentially these same issues. Either the conditions need to be taken into account during the training of acoustic model or model has to be adapted afterwards. Adaptation is a common technique for adjusting parameters of general acoustic model for a specific acoustic situation. Adaptation can significantly improve performance for speakers that are not well represented in the model's training data. The greater the mismatch between the adaptation and training

data, the more the adaptation improves the accuracy.

In this thesis acoustic model adaptation method Maximum a Posteriori (MAP) is implemented into AaltoASR system which has been developed in Speech Recognition group of Aalto University. MAP adaptation is commonly used in speaker adaptation, but like other adaptation methods, it can also be used to adapt any acoustical characteristics such as environmental noise or styles of speech. The most common adaptation scheme is speaker adaptation, where the acoustic model is adapted for a particular speaker. While speaker adaptation is simple adaptation task to perform, because adaptation data set is small and highly homogeneous, it usually increases accuracy remarkably. For this reason, speaker adaptation is used in this work to experiment functionality of MAP implementation.

The performance of MAP adaptation is compared to Constrained Maximum Linear Regression (CMLLR) adaptation. MAP and CMLLR are both commonly used adaptation methods and their theory is well-tested and widely known. Because it is known how adaptation methods should perform related to each other, the comparison is used to verify that MAP implementation is working properly.

The secondary objective of this thesis was to experiment whether MAP adaptation is suitable to improve an acoustic model for colloquial language. The acoustic model was trained with multiple corpora and not all of them represent colloquial language well. Therefore, it is investigated if MAP adaptation can be used to give more weight to certain corpus. Even though colloquial language has many interesting problems on its own, we will not cover this subject in detail. This being the case, we will focus on MAP adaptation, the topic of colloquial language acoustic model development being only touched upon.

The structure of this thesis can be divided into two parts. At first the theory needed to understand ASR systems and MAP adaptation is introduced in Chapters 2 – 4. Chapter 2 is about the basic framework of ASR systems. We begin by introducing the history of the technology related to ASR. Then, the basic framework of ASR system, i.e., feature extraction, acoustic model, language model, and decoder are introduced briefly. Chapter 3 introduces Maximum Likelihood training scheme of acoustic models. Adaptation and variations in speech signal in general are examined in Chapter 4, including theory behind MAP adaptation and MLLR adaptation methods.

The second part is dedicated in presenting the implementation and the results. MAP implementation is introduced in Chapter 5, and it explains important parameters and how MAP was implemented into AaltoASR. The testing on implementation is explained and analyzed in Chapter 6. Finally experiments and results are discussed further in Chapters 7 and 8.

Chapter 2

Automatic Speech Recognition

In this Chapter Automatic Speech Recognition (ASR) is introduced. Section 2.1 introduces the history behind the development of the most important technologies used today in ASR systems, dating back to the 1950's when the first digit recognizer was built. Key points of the following decades are explained including N-grams and Hidden Markov Models. Finally, Neural Networks that were successfully used for ASR in the 1990's and again in the beginning of 2000 are introduced.

The typical statistical framework of HMM-based ASR system is explained in Section 2.2. The goal of this thesis is to implement MAP adaptation scheme into AaltoASR, which is an ASR system developed in Aalto University. AaltoASR was designed for a Finnish large vocabulary continuous speech recognition (LVCSR); hence, some less popular methods, such as morphs, are also included in the system.

2.1 History

Speech is the most natural and easiest way for humans to communicate and transfer information. Ever since computers were invented, humans have been yearning to interact with them by solely speaking. However, recognizing and understanding speech is actually a highly complex task even though this may be hard to believe, because humans are specialized in hearing and understanding speech [33]. Speech recognition and synthesis has been an active research field since the 1930's and it has progressed remarkably, though some might think slowly, from a simple and limited single digit recognizer to a complex LVCSR system which can be used in many practical applications.

The technology of speech recognition system has progressed gradually in each decade. The first real breakthrough took place during the 1950's when

the famous Bell Laboratories built the first isolated digit recognizer [10]. However, system development was made possible by the efforts of speech pioneers such as Harvey Fletcher [15] and Homer Dudley [13] who understood the importance of frequency spectrum and the phonetic nature of speech sound. Modern speech related algorithms and methods process speech signals in frequency domain. Use of the frequency domain is beneficial due to several facts. One of them being that each phoneme contains characteristic frequencies, formants, which can be seen even by human eye in the spectrogram. In the frequency domain, components correlate with each other less than in the time domain, therefore, each component can be modeled and processed independently.

Early ASR systems used frequency pattern templates [32]. Each word had its own template, which input signal was compared to. Formant frequencies were measured from the input speech signal, processed into template and compared to the reference templates with pattern recognition. At that point only 10 to 100 isolated words could be recognized. Forming templates for each word, however, was quite inefficient because it limits the size of the vocabulary. Instead, templates were formed for each phoneme, which opened the door to continuous speech recognition [12, 47]. Template-based systems dominated the field until 1980's, when systems shifted into statistical approach.

By the 1970's, vocabulary of ASR systems had grown into medium sized (100 to 1000 words) and could recognize connected words and digits. Systems were still template-based at the time. However, ASR field started to evolve into new directions. Before, a pure speech signal was used as an input. Pure signal, however, is inefficient as an input signal since it contains much unnecessary information. Instead, essential features were extracted from the speech signal. Linear Predictive Coding (LPC) [1] was the first feature extraction method to be introduced to the ASR field. In the following decades extracting features from the speech signal before inputting them into the system became the state-of-the-art technique, and research of extracting these vectors became a hot topic. In the 1980's feature extraction methods Perceptual Linear Prediction (PLP) coefficients [22] and Mel-Frequency Cepstral Coefficients (MFCC) [11] were introduced and all of these methods are still commonly used in speech processing.

In 1975, IBM [28] introduced a statistical language model that was based on N-gram models. Language models before N-grams were simple word nets showing which sentences are allowed. It was not possible to recognize a sentence not found in the word net. Thus word nets are not suitable for LVCSR systems with an infinite number of sentences. N-grams can model language statistically and assign probabilities to word sequences in a way that

does not restrict the usage to predetermined sentences. N-grams are powerful statistical modeling method and have belonged to the ASR framework ever since.

ASR systems were commonly speaker dependent or speaker independent with small vocabulary in the 1970's [17]. Gradually, researchers began taking interest in developing larger speaker independent systems. Even with all the remarkable improvements with recognizers in the 1970's it was not possible to create SI LVCSR system with the prevailing technology. Template-based acoustic modeling became an issue, since templates cannot model a natural variety of the speech efficiently. Researchers started to see the wall with the intuitive methods and got interested in potential of statistical methods and finally in the 1980's HMMs made their big breakthrough.

Hidden Markov Model

Before the 1980's, ASR systems modeled speech with simple templates which could not handle the variety of speech effectively. It was noticed that stochastic approach is more suitable for speech related tasks. Instead of one template pattern, stochastic modeling method called Hidden Markov Model models a phoneme with a probability distribution, usually Gaussian Mixture Model. Thus acoustic variability can be modeled. SI systems were able to increase their vocabulary which enabled ASR systems to be used in proper applications.

Markov chain was discovered by A. A. Markov [40] already in the beginning of the 20th century. Before Baum-Welch algorithm was introduced in 1960's, there were no efficient method to estimate model parameters. Baum-Welch algorithm is a special case of a more general parameter estimation framework called Expectation Maximization (EM) algorithm [44]. With EM algorithm it is possible to iteratively train HMM model parameters. Different kinds of criteria can be used in estimation, but the first and still the most common one is Maximum Likelihood. Later on, other possible estimates have emerged, such as discriminative estimate Maximum Mutual Information (MMI). Discriminative training is utilized in the HMM based state-of-the-art systems [44], because it gives better estimates although it requires more computation power and tends to overfit more easily.

The basic theory of HMMs and some practical applications in speech processing was published by Baum et al. [3–7] in the late 1960's. However, it took nearly 20 years before HMMs become more common, mainly because the information about them did not reach or could not be understood by potential users [45]. The earlier papers were published in mathematical journals, which were not popular among engineers and researchers working with

speech processing. In addition, those papers were hard to understand and did not provide a sufficient tutorial for readers to implement HMMs into their own research [45]. When this was noticed, many tutorial reviews were written and published in journals in the fields of signal processing and acoustics [29, 36, 46]. In following decades, HMM become the main building block for speech processing systems such as ASR.

At first, log-concave and elliptically symmetric density functions were used to model each state in HMMs [32]. When more people attempted to use HMMs, it, however, become clear that these density functions did not model speech variety sufficiently in speaker independent tasks. In the early 1980's Bell Laboratories extended the theory of HMM to mixture densities [30, 31], thus Gaussian Mixture Models (GMMs) could be used with EM algorithm.

The basic framework of HMM-based ASR system has not changed much since. HMM-based systems still include feature extraction, language and acoustic model. Framework has retained its usability because of its flexibility and constantly improving methods and algorithms. With a HMM-based system, it is possible to train small and fast systems, but also large and complex systems without changing the training procedure. In addition, a huge amount of time and money has been invested in developing new algorithms and tuning systems, so it might be hard for some to give up on it. [44]

Neural Network

There had not been any challengers for GMM-HMM based ASR systems as they are well suited for speech recognition and have continuously been improved. In the 1990's, Bourlard and Morgan [9] tested if Neural Networks (NN) could be the next state-of-the-art technique in speech recognition. They built a recognizer where NN with a single layer of nonlinear hidden units was used to replace GMMs in HMM based system. Even though, this hybrid method was able to successfully predict the correct HMM, the performance was still lacking from GMM-HMM. The greatest obstacle at the time was the inadequate hardware and learning algorithms that were not capable of training NNs with many hidden layers and large output layer with large amount of training data. Large output layer is needed because the number of possible triphones, a combination of three phonemes, is thousands even if similar ones are tied together. Hence, monophones had to be used with NN. During last decade problems with hardware and algorithms have been overcome which enabled the introduction of Deep Neural Networks (DNNs) consisting of many hidden layers.

Combined with new training methods, DNN has become a promising

challenger for older methods. Nowadays, DNNs can outperform GMMs at acoustic modeling even with large data sets and vocabulary [23]. DNNs are able to approximate any function with desired accuracy given enough layers and nodes, similar to GMMs. Moreover, DNNs can also model properties that GMMs cannot. DNNs make more use of the training data as well. They need less training data than GMM-HMM based systems to achieve the same performance. However, there is no efficient way to parallelize the fine-tuning of DNNs. DNNs are more flexible than GMMs and they can be used in varying ways. In speech recognition field neural networks have been used also in feature extraction and to build the whole ASR system [19–21]. [23]

2.2 Overview

A typical framework of automatic speech recognition system, see Figure 2.1, includes two statistical models; acoustic and language model. *Acoustic model* gives the most likely acoustic unit based on acoustical properties of the input signal. However, before the speech signal is handed to the models, it is compressed by extracting relevant *features* from the signal. *Language model* has the knowledge about a language. It defines allowed words and how likely they occur together. *Lexicon* is a pronunciation dictionary which explains vocabulary in the phone-level. The final transcription for speech signal is decided in the *decoder* that utilizes the probabilities given by the models.

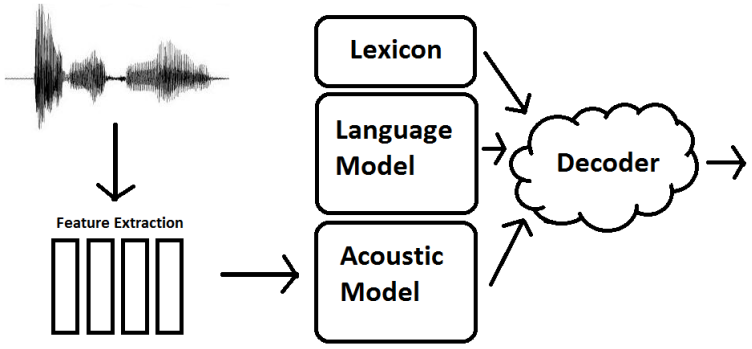

Figure 2.1: Structure of a typical ASR system.

The core problem in speech recognition is a statistical optimization problem of finding the estimate of the most probable sequence of words $\hat{W}$ when we have observations O. We want to maximize posterior probability $P(W|O)$ which according to Bayesian law can be written as follows [26]

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(W|O) \tag{2.1}$$

$$= \underset{W}{\operatorname{argmax}} \frac{P(W)P(O|W)}{P(O)} \tag{2.2}$$

$$= \underset{W}{\operatorname{argmax}} P(W)P(O|W). \tag{2.3}$$

The language model gives probability $P(W)$, and $P(O|W)$ is given by the acoustic model. In practice, observations O are feature vectors.

2.3 Feature Extraction

Feature extraction is a process that converts an input audio signal into more compact, fixed-sized acoustic vectors. Speech signal has a lot of unnecessary information from the recognition point of view, such as, background noise and the natural variety of human voice. The aim of feature extraction is to reduce both irrelevant information and noise, and therefore to make the recognition task easier and faster by reducing dimensions of the input while trying to minimize the loss of information that discriminates words from each other.

Speech and sound signals are usually processed in the frequency domain instead of the time domain, because frequency components correlate less with each other and each component can be modeled and processed independently [26]. Each sound has its own characteristic spectrum and it is possible to deduct what sound is being said based on the frequencies at each time interval. Differences are easy to detect by a human eye, as can be seen in the spectrogram, Figure 2.2, where Finnish word *kaksi* is represented in the time and the frequency domain. For example, vocals a and i can be seen to have different spectral peaks, formants.

In feature extraction, signal is divided into frames by taking overlapping 25ms Hamming window every 10ms. The window has to have a suitable length so that the phonetic information does not change, but not too short in order spectrum to be sufficiently stationary. Windows are then converted into the frequency domain with the fast Fourier Transform (FFT) and processed into feature vectors with a chosen method. One of the simplest and most widely used extraction methods is Mel-frequency Cepstral Coefficient (MFCC). MFCC utilizes the knowledge of human hearing and gives more emphasis to low frequencies that are important for humans to understand speech and suppresses frequencies with unimportant information [17]

MFCC features are extracted from speech signal as follows [17, 51]:

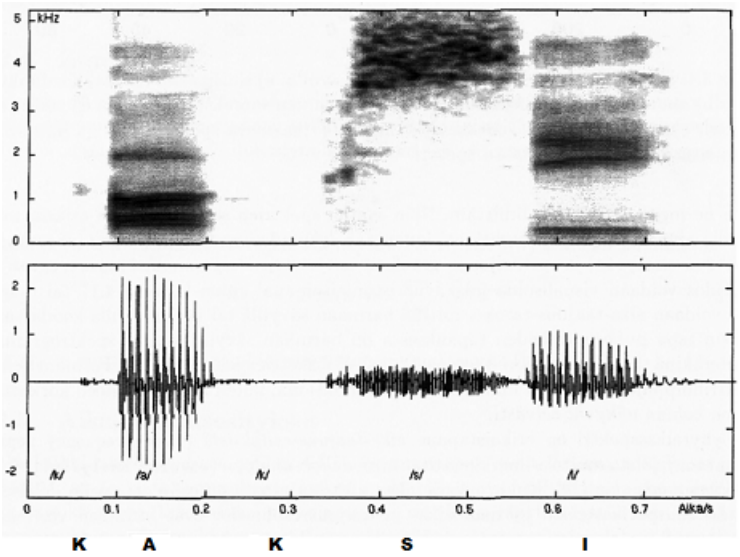

Figure 2.2: Upper diagram is the spectrogram of the word *kaksi*, where frequencies are in the y-axis and time in x-axis. Bottom is *kaksi* in the time domain, but instead of frequency, amplitude is given on the y-axis.[33]

1. The speech signal is divided into frames by taking overlapping 25ms Hamming window every 10ms.

2. Frames are converted into the frequency components with FFT.

3. Spectrum is mapped into a non-linear *mel-scale* by using triangular overlapping windows. Mel-scale approximates the non-linearity of the response of the human ear by giving more emphasis on low frequencies.

4. The components are compressed by a logarithmic energy function to mimic the loudness perception of the human auditory system.

5. A discrete cosine transformation (DCT) is used to reduce the dimensionality, without much impact on the modeling accuracy, and to obtain an uncorrelated feature vectors.

6. HMM-based models assume that individual HMMs are independent of each other, which is not, however, entirely true. Hence, first and second time derivatives are computed from surrounding components. These components are often called delta and delta-delta coefficients.

2.4 Acoustic Model

Acoustic properties of speech must be modeled with a statistical model because of variation in speech signals. Even if humans are requested to say aloud the exact same word, the acoustic signals are not identical, but can even be remarkably different. Because of this variation, there cannot be a single fixed model that is suitable for all sounds. Instead, we can model the sounds with a probability distribution, where the variation is taken into account. Statistical Hidden Markov Models are suitable for the task because they are flexible, reasonably fast, and well-performing. It is also easy to implement new methods to HMM-based system, e.g., adaptation methods to HMM models. Not only is it possible to train small and fast HMM-based systems but highly complex ones as well.

If we regard speech from a statistical point of view it can be seen as a time varying discrete Markov process, where each time instance can be described as some state that belongs to a defined state space $s_1, s_2, ..., s_N$, where N is the number of possible states. States can change to another state according to transition probabilities a_{ij} which describe how likely it is for a state to undergo a change from state i to j. It is also possible that state remains the same for multiple time instances.

One of the most important part of creating a model is choosing what speech unit each state represents. In small vocabulary systems, speech unit can be whole words, but usually in large vocabulary continuous speech recognition systems (LVCSR) speech unit is a subword called phoneme which is defined to be the smallest sound unit words are composed of and whose change will affect the meaning of the word itself [44]. Phonemes are different in each language, e.g., Finnish has 21 and English 44 phonemes.

Phoneme subunits are highly dependent on the context they appear in. They do not always sound the same but vary depending on the context they are in. The dependency becomes even more important in fast spontaneous speech, since many phonemes are not fully realized [26]. In ASR systems, context is taken into account by using triphones where context of the phoneme is also included. For example, triphone $\texttt{t} - \texttt{a} + \texttt{n}$ is a phoneme `a`, which is preceded by a phoneme `t` and followed by a phoneme `n`. Each triphone is modeled by a three-state HMM. More than three states can be used but it is not recommended to build a model larger than necessary because the greater the model complexity, the more free parameters has to be estimated.

For N phonemes, there are N^3 potential triphones. The number of triphones is decreased by clustering and tying together parameters of similar states with decision trees [50]. Triphones, that do not exist in the language,

are also removed. Triphone models are trained with examples extracted from the training data. Rare triphones do not occur often to have enough samples for training, which is why uncommon triphones are combined into a single "rubbish model".

When recognizing speech, we are basically trying to find out which phoneme is hidden in each feature vector. In other words, we do not know the state (phoneme) the process (speech) is in or moving into the next state. We can only observe the process from the outside in the form of feature vectors, which are extracted from the input speech signal we are trying to recognize. Because the actual states are hidden, Markov process cannot be directly modeled, which is why Hidden Markov Model (HMM) has to be used.

Each state in HMM has a conditional distribution of observations called an emission distribution b_i. Figure 2.3 shows an example of a simple three-state HMM, which is a typical model for triphone. Emission distributions are used to recognize correct triphones that enter the system. The number of triphones is thousands even after clustering similar states together, which is why a model that is able to express even highly complex distributions is needed.

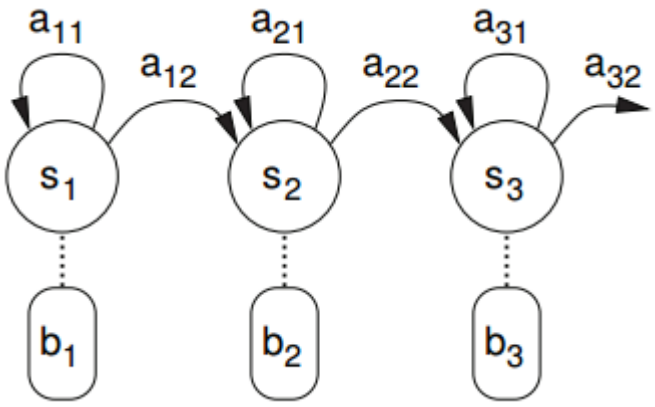

Figure 2.3: Three-state left-to-right HMM. The model is characterized by the set of states s_i, the transition probabilities a_{ij} and the emission distributions b_i. [44]

Emission distributions are typically Gaussian Mixture Models (GMMs) as they are flexible and have excellent properties. GMM is the probability density function represented by a weighted sum of Gaussian component densities. GMMs are superior to any other density functions, because they are able to represent any distribution with desired accuracy, presumed that

enough components are given [49]. A simple GMM is presented in Figure 2.4. Formally GMM is defined for each state j as

$$b_j(\mathbf{x}) = \sum_{k=1}^{M} c_{jk} N(\mathbf{x}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}), \tag{2.4}$$

where M is the number of components in the mixture, $\mathbf{x}$ is the feature vector and c_{jk} is the weight of distribution. Weights need to satisfy a condition $\sum_{k=1}^{M} c_{jk} = 1$. Gaussian distribution $N(\mathbf{x}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$ is given by a mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$ as follows

$$N(\mathbf{x}, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{\boldsymbol{\Sigma}_k \sqrt{2\pi}} \exp - \frac{(\mathbf{x} - \boldsymbol{\mu}_k)^2}{2\boldsymbol{\Sigma}_k^2}. \tag{2.5}$$

The number of mixture components depends on the training data. The more complex the data, the more complex the distribution has to be, in order to model the data adequately, thus the mixture requires more components. Gaussian distributions have the dimension of the feature vectors.

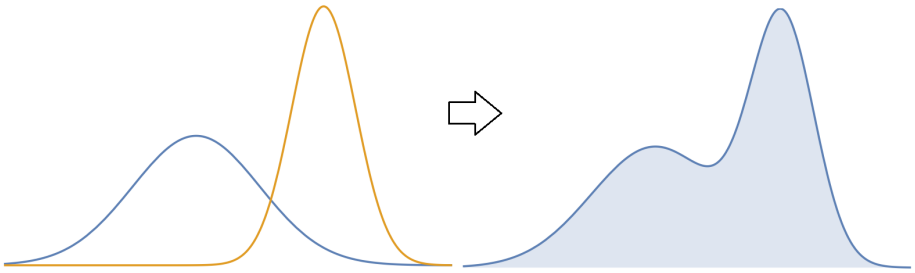

Figure 2.4: Gaussian Mixture Model is weighted sum of Gaussian distribution components. GMM in the figure has two components with equal weights. The left side shows the components separately and in the right is the modeled distribution.

2.5 Language Model

Acoustic information alone is not enough to recognize speech properly, thus, in addition, knowledge about a language is needed. Some information about the language is already modeled in the acoustic model, but the language model models language on a higher level in a form of vocabulary and grammar. As vocabulary increases, the number of acoustically similar words that

are easily confused with each other also increases. Language model utilizes a context and knowledge of the language to distinguish between words and phrases, and is capable of recognizing correctly regardless of acoustical similarities.

Language models have two purposes. Firstly, they define which words can be recognized by the system with lexicon. This helps to restrict the search space by ignoring phoneme sequences that are not allowed in the language which make the recognition process more efficient. Secondly, language models give a probability distribution $P(W)$ for word sequences in order to find the optimal sequence that maximizes Equation 2.3. [25]

Lexicon is a pronunciation dictionary that contains all allowed words and how they are pronounced in the form of triphones. However, in larger systems, the least common ones are discarded to diminish the search space. In Finnish, lexicon is typically generated based on what words occur in corpora used in training. However, in English, lexicon is typically created by linguists because the pronunciations cannot be easily generated automatically.

Modern statistical language models use N-gram models. An N-gram model is a powerful statistical representation of a grammar. It is able to model the grammar of a language by assigning a probability for an N word sequence, $w_1, ..., w_N$. N-gram is a word sequence that has N words. Basically, for each word there is a known history of N words. The history gives us a conditional probability distribution for the next word. Formally $P(W)$ can be written as [25]

$$P(W) = \prod_{i=1}^{N} P(w_i|w_1, ..., w_{i-1}). \tag{2.6}$$

For example [26], consider the sentence `Mr. Wright should write`. This is sentence acoustically highly challenging to recognize, but the language model has the knowledge of the context and can give a higher probability to name `Wright` than to the verb `write` after the word `Mr.` This is why it is important to choose the training corpora carefully so that it represents the recognized speech well enough.

Trigram models, which take into account the two previous words, used to be the most commonly used. Recently higher order N-grams have become more common. The accuracy can be significantly improved with higher order N-grams if suitable amount of training data is available. However, there is not much improvement after 5 to 7 grams [18].

N-gram models are trained with large text corpora. Probability $P(W)$ in Equation 2.3 gives the likelihood of the word sequence occurring. For example [17], in trigram model, let $C(w_{k-2}w_{k-1}w_k)$ represent a number of

occurrences of the three word sequence $w_{k-2}w_{k-1}w_k$ and $C(w_{k-2}w_{k-1})$ the occurrences of the sequence $w_{k-2}w_{k-1}$. Now, the probability of the trigram is given by

$$P(w_k|w_{k-1}, w_{k-2}) = \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})}. \tag{2.7}$$

The problem with this simple scheme is that it gives a probability of zero to word sequences that do not appear in the training corpus. These zero probabilities can be solved by smoothing methods, in which some of the probability mass is moved to models that do not have any mass. [50]

Lexicon is formed from the most common words in the corpora. In English, lexicon has typically 20 000 to 60 000 words [25]. All the words cannot be included in lexicon, because the search space would grow too large and make recognition unpractically slow. However, it is not possible to recognize out-of-vocabulary (OOV) words. OOV words are either missed or replaced by a wrong word which affects the nearby words, because n-grams aim to make sensible sentences.

Word based language models are suitable for languages such as English, where with a reasonable-sized vocabulary, it is possible to cover the most commonly occurring words. OOV words are not a big issue in English, but in a morphologically rich language it is a problem. In morphologically rich languages, such as Finnish, inflection and compounding causes the number of distinct word forms to become huge. Because of this, a Finnish corpus with 40 million words has over 1.9 million unique words, whereas an English corpus of the same size has only 190 000 [25].

For language models for morphologically rich languages, there exists a data sparsity problem, because there is not enough data to train reliable N-gram estimates for each word in each context. More time is required for recognition, and in case the vocabulary is large, much memory as well. For these reasons morphs are used instead.

Words are formed by joining morphemes, which are the smallest grammatical units in a language. In speech recognition, words are segmented into statistical morphs, which are similar but not identical to morphemes, as morphs are generated automatically because it does not matter if morphs have real meaning in the language or not. If morphs are used as an N-gram unit, by combining them, it is possible to recognize words that do not appear in the corpus. [25]

2.6 Decoder

The actual recognition task is performed by a decoder. Decoders combine probabilities of acoustic and language models to maximize the probability of Equation 2.3 to find the most probable word sequence. Decoders have two important properties [44]. Firstly, decoders have to be reliable to find the best of the most fitting word sequences. Secondly, the search needs to be efficient. The optimal word sequence can be found by going through systematically all possible sequences and choosing the most probable one, though this method is highly inefficient. This kind of method works with smaller systems, but as the number of possible words increases, the search space grows exponentially. In LVCSR, the search space grows so large that it would take too much time and computational capacity to compute each possible word sequence. Hence, in practice the search is limited to the most probable paths, which can be achieved by pruning. Pruning removes the unlikely hypotheses during the search. The search in decoders is usually performed with the Viterbi algorithm [16, 44].

The Viterbi algorithm can be used to find the most likely word sequence. In ASR decoder, the Viterbi algorithm goes, recursively, through possible word sequences, scoring them according to models. Unlikely word sequences are removed during the search. It is also possible for a decoder to return an N-best list of alternative word sequences, which increases the likelihood for correct sequence being among the list given to the user. Using N-best list might be suitable, e.g., for transcribing a dictation where an uncertain word might have some choices, where the user can choose from.

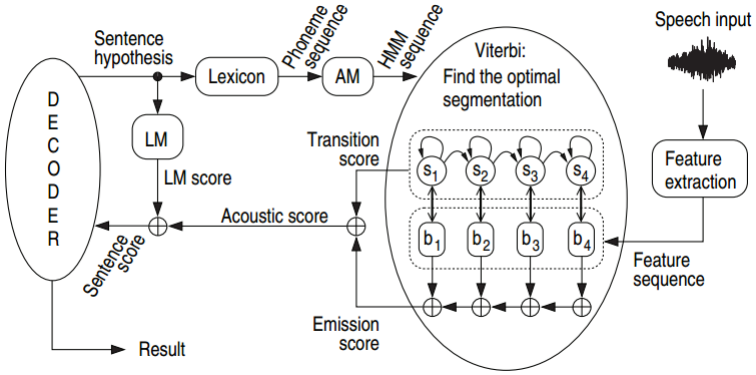

Figure 2.5: Structure of AaltoASR decoder decoder[44]

Figure 2.5 shows an information flow of an LVCSR system. The decoder

generates a sentence hypotheses, which are then ranked based on the scores given by acoustic and language models. The Viterbi algorithm operates in the decoder and also over the HMM sequences defined by the acoustic model. [44]

Language models give significantly smaller range of probabilities to decoder than acoustic models, hence they need to be scaled in order to affect decoding. This scale is called LM scale, which can also be used to determine the weight given to models. The suitability of an acoustic and language model can vary quite a lot depending on the task, thus more suitable model should be given higher weight. This is not to say that both of the models are not necessary. Even though usually we can predict which of the models should be weighted more, the optimal LM scale has to be determined empirically.

2.7 Evaluation

Evaluation of speech recognition systems may seem quite simple but analyzing error with a meaningful way can actually be very difficult. In error evaluation, the recognition output is compared to original transcriptions of sentences. The most widely used error measure is Word Error Rate (WER), where words are used as a base unit. Calculating WER is relatively simple [50]; the basic idea is to determine how many words needs to be corrected to turn the transcription into the recognition output. Incorrect words can be divided into three categories: inserted W_I, deleted W_D and substituted words W_S. WER is then defined by

$$WER(\%) = \frac{W_I + W_D + W_S}{W_N} \cdot 100\%, \tag{2.8}$$

where W_N is the number of words in the original transcription. The smaller the WER is, the better the system performs. It should be noted that WER can be more than 100% if the number of erroneous words is larger than the number of original words in the sentence.

There are, however, problems with the WER measure in languages that have long words and suffixes. For example, Finnish, which is used in this thesis, is one of these languages. In WER, the error of one letter is given with the same weight as a whole word being substituted or even deleted completely. However, the word can be understandable even if few letters have been replaced. For these kind of errors, a smaller error unit might be more descriptive. Hence, in this work Letter Error Rate (LER) is also used as an error measure. It is defined in a way similar to WER, but instead of

whole words, letters are compared to measure error. LER, of course, is not a perfect measure either since only one incorrect letter can already change the meaning of the word. WER and LER cannot be used as an absolute measure since there are many components that affect the recognition results, which is why a relative error reduction should be applied. However, using and comparing WER and LER can give us better understanding of the actual performance of the system.

According to Huang [26], for reliable results, the evaluation set should contain more than 500 sentences (6 to 10 words each) from 5 to 10 speakers. Huang also states that there should be more than 10% relative error reduction before actually including a new algorithm. However, this condition may be a bit strict depending on the starting point and how the system is going to be used. If the base model already produces good results, further improvement is challenging. In addition, a more general view should also be taken when deciding whether the algorithm should be implemented or not.

The evaluation set is used to measure the performance of the system after it has been optimized with the validation set. If the evaluation set is used to optimize the parameters, there is a risk of overfitting the model to the data, which reduces the general performance of the model. Dividing the data into a training, a validation and an evaluation set to be used in different phases in system development is a normal practice used in a machine learning.

The value of evaluations depends on how realistic the evaluation set is compared to how system is actually used. It is also possible that the method only works for very limited tasks, which is why it is recommended to test the method widely with different type of tasks to evaluate if the method is robust. [44]

Chapter 3

Training Acoustic Model

Acoustic models are statistical presentations of the acoustic properties of the speech as explained in Section 2.4. Model is trained utilizing machine learning techniques that are introduced more closely in Section 3.2. Learning algorithms require training data in order to find the most optimal solution for model parameters. In acoustic models, corpora are used as training data. Corpora include speech recordings with accurate transcriptions of what is being said in each recording. There is a lot of variation in the speech itself, but also in recording conditions, which need to be taken into account when selecting a suitable corpus for training. The more similar the training data is to the input of the final product, the better the accuracy of the system is. The performance of the model is tested with the evaluation data that shares similarities to the speech intended to recognize with the system.

The variety of a speech can cause a mismatch between the model and the evaluation data. The mismatch can be viewed either from the point of view of the feature domain or the model domain. Ideally all the variation and noise are filtered away during the feature extraction phase, but, unfortunately, extraction techniques are not perfect, and some unimportant information is always left in the feature vectors. That is why the mismatch compensation is performed also in the model domain. The simplest method of model compensation is to select the training data depending on the source of variation. Speaker Independent (SI) acoustic model, trained by multiple speakers, includes the variation of different speakers. Acoustic conditions are compensated similarly by selecting training data including expected acoustic conditions. If a more general model is wanted, which is able to perform adequately well in all acoustic conditions, *multi-style training* is to be used. Multi-style training is a training style where different acoustic conditions are included in the training data [37]. Section 3.1 introduces different ASR systems and type of training data they require shortly.

Even though the variety of speech has to be taken into account already in the training phase, the different types of variety are more closely investigated in the following Chapter 4. This chapter focuses on the different types of system and training schemes of acoustic models. After introducing Speaker Dependent (SD) and Speaker Independent systems, Maximum Likelihood (ML) training is described. At the heart of all training methods of an ASR system is the EM-algorithm which is introduced in Section 3.2.

3.1 System types

Because it is not possible to create general ASR that is suitable for all tasks, ASR systems are designed for different purposes and for different types of languages. To improve the performance of the system, typically a number of constrains can be imposed on the speech recognizers and the users depending on for what kind of use the system is made for. Constrains limit the speech system is able to recognize. For example, it is possible to train system that recognizes only certain words.

If the task does not require continuous speech recognition, it is recommended to design a simpler system that recognizes only the words that are needed. Limiting vocabulary is a simple way to make system more robust and increase the recognition accuracy [26]. The simplest ASR system is an isolated word recognizer that recognizes only single words. These kind of recognizers are usually used in command interfaces. The commands should be selected so that the vocabulary includes words that are acoustically as different as possible in order to improve accuracy of the system. Unfortunately, using an isolated word ASR system feels unnatural for the most users, because users have to pause between words. Humans rarely keep distinct pauses between words while speaking, hence isolated word systems are not suitable for applications that require long sentences. In addition, limiting vocabulary limits the versatility of the system. However, if a larger vocabulary is used, accuracy declines due to the exponential growth of possible word combinations and increase in acoustically similar words. If the system is well designed for the task, an isolated word system performs fast and is highly accurate. Uses are, however, limited and if insufficiently designed, usability can easily be lousy, hence the suitability of an isolated word system should be carefully considered.

The next stage is to allow connected words or continuous speech but limit vocabulary. These systems perform adequately well, but need to be designed specifically for the application, and hence do not usually function well for other purposes. Two decades ago, before mobile phones became more com-

mon, speech recognition was mainly used for dialing, call routing, dictation, and command and control applications. Applications had limited vocabulary and the recording conditions were quite constant. However, recently the growth of big data and computational power have enabled ASR technology to be applied in movable devices, such as laptops and smart phones, that have more complex acoustic environment and applications. It is possible to interact with a smart phone via speech, e.g., with Siri on iPhone and Google Now on Android. Voice commanding is becoming more common in home entertainment systems, such as Kinect on xBox and Amazon's Echo. ASR systems have to cope with more difficult acoustic environments than in the past, which causes pressure on developing more robust acoustic models.

Acoustic models can be classified into Speaker Dependent (SD) and Speaker Independent (SI) models. The SD models are designed to recognize a specific speaker and are trained with a few hours of this speaker's speech. The SD models were popular a few decades ago, because they do not require much training data and simple training methods were sufficient. The SD systems can also provide WER two to three times lower than the SI system with the same amount of training data [52]. The SD models are also less complex and faster than SI models. SD models are, however, quite impractical, because training data has to be collected from all of the possible users. [26]

It is not possible or reasonable to collect enough training data from all possible users and train SD model for each of them. Therefore, Speaker Independent models are applied instead. SI systems are trained with tens to thousands of hours of speech from hundreds of speakers. SI models are more robust and can handle larger variety of speakers better than SD models. Robustness also applies to recording conditions. If acoustical environment the system is applied to, is not known beforehand, acoustic model can be trained with speech from different acoustical environments to make the model robust of change in recording conditions. It is not, however, possible to make a good model for everything by merely using a highly diverse training data, because the heterogeneity of the data increases the spread of the model and reduces the general accuracy compared to the task-specific models.

3.2 Training

Training an acoustic model is basically finding the most optimal parameters for HMMs. Finding the optimal parameters is a challenging problem. There is no solution in a closed form [26], thus the model has to be trained iteratively with Expectation Maximization (EM) algorithm. The EM algorithm enables an iterative training of the model parameters with a training data

that includes transcribed speech recordings. Transcriptions are utilized to collecting the correct examples of triphones for each HMM state.

The EM algorithm has two steps. The first is the E-step where the statistics are accumulated from the training data. Then in the M-step, the model parameters are re-estimated based on the statistics accumulated in the E-step. After the re-estimation, the parameters model the training data better. The objective of the EM algorithm is to maximize the chosen objective function. Because EM algorithm is an iterative algorithm, the E- and M-steps are repeated to increase the log-likelihood of the objective function until the function converges close enough to a local maximum.

For clarity, let us go through an example [48] on using HMMs in an isolated word recognition. In this example, Figure 3.1 (a), a single HMM is trained for each word instead of triphones. The system is able to recognize three words: `one`, `two`, and `three`. For each of these words, HMM is trained with the examples of these words found on the training data. Based on the accumulated examples, parameters b_i of the HMM model are estimated with re-estimation equations. Equations are derived from the objective function used. After the training, words can be given to the system for recognition. In Figure 3.1 (b), unknown observations $\boldsymbol{O}$ are recognized by computing the likelihood probability $b_i(\boldsymbol{O})$ for each model i and the model with the highest likelihood is chosen as the recognition result.

In practice, the training is started with a simple initial model where each mixture b_i has a single Gaussian component. Each of these mixtures models a state in a triphone instead of words, which were used in the previous example. For each mixture component, occupancy is accumulated in the E-step of the EM algorithm. The occupancy describes the probability mass of the distribution. When the occupancy reaches a certain pre-determined limit, the Gaussian distribution is *split* into two Gaussian distributions, and the mixture gains another component. On the other hand, if the occupancy of Gaussian distribution does not achieve a minimum occupancy level, the Gaussian is *merged* into other distributions. Splitting and merging help determining how complex each mixture should be. The procedure also causes each of mixtures to have a different amount of components. The number of components depends on how many times the triphone state occurs in the training data, because each occurrence accumulates occupancy.

When the number of the mixture components is not determined before hand, the model complexity depends on the training data. More complex models can be trained with large data sets, because sets have higher occupancy for each HMM state. Large models have to be trained with more iterations than smaller ones, because the growth of the model is limited in each iteration. In addition, distributions are not usually split in every itera-

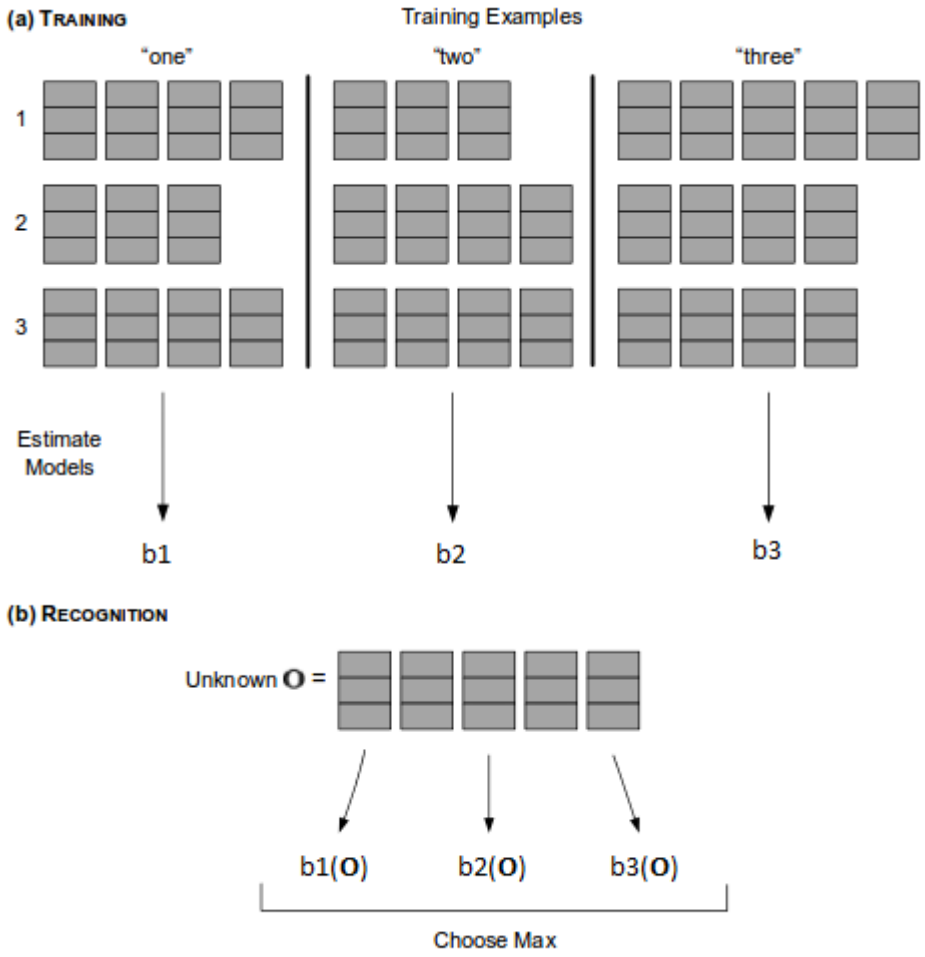

Figure 3.1: Training scheme example of an isolated word recognizer. [48]

tion in order to first estimate better model parameters for the current model. However, if the model is trained with too many iterations, it will overfit to the data and no longer generalize well to new data sets. Hence, the number of training iterations should be related to the chosen objective function and the amount of data. Some objective functions and training styles overfit more easily than others.

Different kind of criteria, objective functions, can be used in the M-step for the estimation. The most commonly used in ASR field is Maximum Likelihood (ML) estimation. Other objective functions can also be used, such as discriminative Mutual Maximum Information (MMI) [2] function, which

is nowadays commonly used, as well. In discriminative training, recognition accuracy is maximized instead of the accuracy of density function like in the ML training. Discriminative training requires more data and computational capacity and tends to overfit more easily, but otherwise they are superior to the ML estimates. ML training is, however, computationally lighter and produces adequate models, thus it is used in this work and introduced in following. [44]

Training consists of estimating parameters $\Lambda = (A, B)$ for each HMM. A is a set of discrete transition probabilities a_{ij} from state i to j and a set B includes emission probability distributions b_q over states $q \in Q$. In ML estimation we maximize the likelihood function $f(\Lambda|X)$ [51], given observation $X = x_1, x_2, ..., x_N$,

$$\Lambda_{ML} = \underset{\Lambda}{\operatorname{argmax}}\, f(\Lambda|X) = \underset{\Lambda}{\operatorname{argmax}} \sum_{Y} \prod_{t=1}^{N} a_{y(t),y(t+1)} b_{y(t)}(\boldsymbol{x}_t), \tag{3.1}$$

where $Y = y(1), y(2), ..., y(N)$ is a sequence of HMM states.

Estimating the emission probability distribution $b_q(\boldsymbol{x}_t)$ consist of estimating GMM parameters $\theta_q = (w, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ in the equation 2.4 . Re-estimation equations for each iteration of Baum-Welch algorithm have been conveyed in following way [51]

$$w_{q,i} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{q,i}^{r}(t)}{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{q}^{r}(t)}, \tag{3.2}$$

$$\hat{\boldsymbol{\mu}}_{q,i} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{q,i}^{r}(t) \boldsymbol{x}_t^r}{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{q,i}^{r}(t)}, \tag{3.3}$$

$$\hat{\boldsymbol{\Sigma}}_{q,i} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{q,i}^{r}(t) (\boldsymbol{x}_t^r - \hat{\boldsymbol{\mu}}_{q_i})(\boldsymbol{x}_t^r - \hat{\boldsymbol{\mu}}_{q,i})^T}{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{q,i}^{r}(t)}. \tag{3.4}$$

Training data includes multiple sequences which are denoted by R. For each sequence occupancy γ of state q in each feature vector is computed in order to estimate new parameters.

Chapter 4

Adaptation

In previous Chapter 3, we described the common training scheme and how the expected acoustic conditions can be taken into account already in training by choosing the speech with a similar background noise. Generality of the training set is decided depending on the amount of generality desired or whether the system is designed especially for the task. Specialization gives better accuracy for the task, but perform poorly generally. The most suitable system depends on the application. One would think that general systems that perform decently in every situation would be ideal, but unfortunately the more general system, the more poorly it performs in overall perspective. In difficult condition general system do not achieve adequate performance.

Some generalization is, however, advisable for practical reasons. In some situations it is not possible to predict all possible conditions beforehand, because in the end, it is up to the user how and where to utilize the system. In addition, sometimes we simply do not have enough data of the real conditions for training. In this case, the only possibility is to train a general model in order to the model to perform reasonable well. It is possible to utilize even small amount of data available to tune parameters of the general model closer to specialized model. The fine-tuning method is called *adaptation*.

Adaptation is used to reduce the *mismatch* between the model and the adaptation data. Both feature vectors and acoustic models can be adapted, but in this thesis the focus is on the model adaptation techniques; MAP and MLLR, which are discussed in Section 4.4. Model adaptation methods usually achieve higher accuracy than feature extraction based methods though model adaptation requires significantly larger computational cost as well [37].

Adaptation increases the performance more, larger the mismatch is. SI models are typically well-trained, hence the performance can be quite good even before the adaptation and the relative gain from adaption is small. There is, however, multiple styles and modes of adaptation and the accuracy

can be increased by choosing the most suitable one. Adaptation can also bring other benefits. In practice, computational resources are limited and users want to have fast real-time applications. Adaptation can also be used to reduce the size of acoustic models [52] by reducing number of parameters. Section 4.1 describes the adaptation modes and when they are usually applied.

Adaptation is most commonly used to make speaker independent acoustic models closer to speaker dependent models. This procedure is called *speaker adaptation*. Speaker adaptation adapts the model to one particular person to improve the recognition of his speech. However, speaker adaptation is not the only way to use adaptation. It is also possible to adapt non-native speakers and those not well represented in the SI training set. The acoustic conditions and the style of speech can be adapted as well, although speaking styles are not solely an acoustic problem. Adaptation of spoken language is experimented in Section 6.2. In Section 4.2 and 4.3 acoustic condition affecting to recognition results are examined.

4.1 Modes

Adaptation methods can be classified into supervised and unsupervised methods. The supervised adaptation requires speech data and transcriptions while the unsupervised adaptation manages only with the speech. In the unsupervised mode, two-pass decoding is typically used, in which the transcriptions are simply generated by ASR system itself. While in such a way estimated transcriptions may be full of errors, confidence on the recognition can be measured to ensure that the adaptation process applies only the most reliable material [52].

Adaptation methods can also be divided into static and dynamic methods. In the static adaptation, all adaptation data is available before the system is used. Dynamic systems, on the other hand, can be adapted during the recognition. A suitable mode of adaptation depends on the applications. In an offline system, it is possible to use multiple passes and even multiple different adaptation methods to ensure the quality. In real-time systems, however, adaptation has to be dynamic due to latency requirements.

Requirements for ASR systems differ depending on whether the system has to perform in real-time or not. The overall performance of the real-time system must operate below the real-time throughput and with small latency. Real-time adaptation has to be unsupervised due to latency requirements. Typically system cannot reserve much memory because in practice, there is limitations in memory and computational capabilities. The most of com-

monly used adaptation techniques fail to reach these requirements. Storing models for every speaker is not feasible in practice as it spends a lot of memory. Instead, speakers should be clustered. Adaptation techniques rely commonly on a static speaker clustering which causes latency because static clustering cannot be performed in real-time. Different solutions to support real-time speech recognition has been proposed, such as incremental version of EM algorithm, dynamic speaker detection, real-time tracking and clustering. [38]

4.2 Speech variability

Speech does not only convey messages but also information about the speaker himself. Humans are able to recognize gender, regional origin and emotions from the speech only. Most of this information, however, is unnecessary to speech recognition system, where the objective is to convert speech into text. Understanding the content is not necessary.

Speech varies greatly from person to another. The physiology of the speaker determines the voice. The shape and length of the vocal tract alters the pitch and range of voice and how much it can vary. Speech also varies even within the same person as it is actually impossible for humans to produce exactly identical speech signal each time, even if asked to repeat exactly same word [26].

The voice changes as human grows because the length of vocal tract also grows. Children have lot higher pitch compared to the adults, especially compared to men. Some alteration also appears when people grow old. Children's speech is challenging to recognize because it differs from adult's speech. Not only because of higher pitch, but also their speech is much simpler and contains made up words and incorrect grammar [8]. Children may change the loudness of their voice as well, even in middle of the sentence, making recording good quality sound quite difficult.

Then there are illnesses that can change the speech from the normal, for example, by affecting the lungs or vocal tract. People may also have speech disorders, such as dysarthria, dysphasia, and stuttering that even humans struggle to understand.

Speaker is the cause of many varieties and noise in the speech signal, and often completely unconsciously, as the variety does not bother the human listener. Speaker makes noises unconsciously by smacking lips or using non-communicational words which do not actually mean anything, such as `"mmm"`. In addition to variety caused by physiology, the environment affects directly the speaker and causes variation in the speech signal, even if the environ-

mental noise itself is filtered. Environment causes the speaker to change the voice louder, quieter, more tense or softer. These changes are often by reflex, such as speaking louder in noisy environment [39].

Speaking styles affect the acoustic properties of the signal. Speaking styles vary depending on regional origin of the speaker and situations speaker is in. Different situations require different speaking styles, e.g., people tend to speak more casually to close friends than superior at work. Non-native speakers have also speaking style of their own as they transfer some of phonetic properties of their native language into the accents. Speech of non-native speakers is difficult for ASR system to recognize because the speech includes phonemes that recognizer is not trained to recognize. Error rate of speakers with accents is 2 to 3 times higher than native speakers' [26].

Speaking styles affect especially in spontaneous speech used in casual conversations. In spontaneous speech, or under time pressure, duration of certain phonemes often reduces. This "slurring" is stronger in parts of sentences that are not important for understanding the message. On the other hand words that are important or easily misheard, tend to be articulated more carefully, or even hyper-articulated. Hence, speaking rate of spontaneous speech can vary highly even during the sentence. Slow speaking rate rarely affects the performance of ASR, but during hyper-articulation people make pauses among syllables, which degrades the performance.

Spontaneous spoken language has typically disinfluences, which makes language modeling difficult. Colloquial language includes false starts, repetitions, hesitations. Pauses can be misplaced as well, or they can be missing altogether. In practice, rereading, where conversation is dictated by someone with a clear voice, is sometimes used to increase recognition rate. With rereading, it is possible to remove most apparent disinfluences. Rereading can reduce WER to half of the original. [8]

People sound different based on the mood they have. The rate and loudness of the speech changes when people are, e.g., angry or nervous. Like people can observe the mood of the speaker from the speech, the mood can be detected from the spectrum of the signal as well. There has been some research to recognize emotions [14] which could be used to select an appropriate model for speaker based on current emotional state.

4.3 Environment

The speech signal can be distorted by many factors that are not related to the speaker or the spoken words. While humans can handle even strongly distorted signals, the ASR systems are highly sensitive to distortions.

Speech signal is a pressure wave that speaker produces from lungs. The pressure coming from the lungs is transformed and filtered into the speech by speaker's vocal tract. Different phonemes are produced by controlling vocal tract, and heard by listeners in the form of pressure wave. These vibrations can also be sensed by microphones that convert the wave into a digital signal that ASR system can process.

When an analog signal is converted into digital, the signal is quantized and modified which distorts the original signal. How much signal is distorted depends on the channel and the quality of microphone. High quality microphones have an ideal flat frequency response but users rarely have possibility to use such microphones, thus usually microphones with non-uniform frequency response are to be expected.

Other sound waves, in addition to speech signal, are also recorded by the microphone. The world is full of sounds. Sounds come from different sources with varying loudness. Even the room itself causes additional signals when the original signal is reverberated from the surrounding walls and objects, which is why the signal arrives to the microphone also through reflections. People can recognize places even eyes closed because places have sounds typical to them. In a workplace air condition, fans of computer, clattering coffee cups and coworkers' chatter is normal background noise. Car, park and beach have their own recognizable acoustic environment. Background sounds are also recorded into speech signal, of course, depending on a microphone used.

The type and placement of the microphone affects how much background noise is included into the signal. The farther the microphone, the more sensitive it has to be to sounds, which is why more additional noise is recorded as well. A headset records the clearest sound because the microphone is located close to mouth. Direction of microphone is important because some microphones are designed to receive signals only from certain directions. If microphone is directed in wrong angle, it will mostly record noise instead of speech. Recording from the distance should be done with a directed microphone or if the subject is moving, with microphone arrays. Microphone arrays utilize beamforming techniques which track the speaker and receive the signals only from the direction of the speaker.

Speech is often transferred via digital or analog channel which affects the quality of the signal. These channels have commonly restrictions and signal are encoded into more compact format and filtered to fit to a certain bandwidth. One of the most common channels is telephone. Older analog telephone lines transfer only frequencies from 300 Hz to 3400 Hz. Remaining frequencies are filtered, which means that speech signal has to fit to only 3000Hz bandwidth while human can hear from 20Hz to 20 000Hz. Cutting

the frequencies can cause clipping and loss of useful spectral information. Modern cellphone and VOIP channels fortunately can transmit bandwidth of 8000Hz, which is sufficient for speech recognition. [51]

Environmental variability remains to be unsolved challenge in the ASR field even though many methods have been developed to increase robustness of models. The noise is compensated in many parts of the ASR process. Many channels and microphones have some compensation and error control of their own. Feature extraction process and missing feature methods can lessen the effect of noise as well. The model compensation, including adaptation methods, are still necessary and work as the final frontier against the noise.

4.4 Methods

In this thesis two supervised adaptation methods are used in experiments: MAP and MLLR adaptation. These two are the most popular adaptation methods because of their simplicity and effectiveness. MAP and MLLR differ quite a much from each other. While MAP adaption adapts each parameter individually, MLLR updates them with a few transformations.

Both of these methods have their advantages and disadvantages. Updating all parameters individually requires more training data, hence in MAP adaptation larger training set is needed than in MLLR adaptation to achieve the same results. Transformation matrix of MLLR can be computed even from a single sentence while MAP requires at least a few minutes. However, as the adaptation data increases, MAP continues to improve more rapidly than MLLR. The basic MLLR does not benefit from the additional data. MLLR transforms typically only means and sometimes covariances, but with MAP it is possible to adapt all HMM parameters.

The following sections introduce theory behind MAP and MLLR adaptation for HMM-based ASR systems.

4.4.1 Maximum a Posteriori

ML estimation is still the most popular training method for HMM based ASR systems. ML estimates, however, are not suitable for adaptation. Typically in adaptation, only a small amount, some minutes, of suitable data is available. If a few hours of data were available, SD model training would be more suitable solution.

Maximum A Posteriori adaptation is a supervised model adaptation technique and is sometimes also referred to as Bayesian adaptation. MAP estimates are suitable in a situation when data availability is scarce, because

MAP estimates do not require large amount of data as they utilize priori information in addition to ML estimate.

In MAP adaptation, model parameters are individually re-estimated. New ML estimates for each HMM parameter are computed based on the adaptation data. MAP estimate is then formed by shifting the original prior parameter values towards the ML estimates. In other words, MAP estimate is a weighted average between ML estimate and prior estimate.

In ML training, the likelihood of the training data $p(x|\lambda)$ is maximized by estimating parameter values λ. MAP is otherwise similar to ML estimation, but in addition to the likelihood, the prior information is added, i.e., $p(x|\lambda)g(\lambda)$ is maximized, where $g(\lambda)$ is the prior information we have about the data. Choosing of the prior is important in MAP adaptation as it affects directly the estimates. In practice, the previously trained SI acoustic model is typically used as the prior, or if the model is iteratively adapted, the parameters of the previous iteration are used. The benefit of using prior is that less data is needed to make robust estimates, which is significant advantage when only a small amount of training data is available.

MAP requires more the adaptation data than MLLR to achieve the same performance because each parameter of GMM components are individually estimated. If the adaptation data consists of a single sentence, the sentence contains only a marginal of all the possible triphones. Only if there is an occurrence of the triphone in the adaptation data, is the parameter updated. The amount of occurrences is regarded as an occupancy. The more occupancy triphone has, the more weight newly computed ML estimate is given in MAP estimation. In other words, if the adaptation set is small, the model parameters do not change much after the MAP adaptation, because new ML estimates do not have much weight. MLLR performs better than MAP with small adaptation set because transformation matrices can be computed even from a single sentence. MAP adaption overtakes MLLR eventually when the amount of the adaptation data increases, because MAP estimates converge towards the ML estimates and SD model. If the training set is large, the weights of the new ML estimates are significantly larger than weights of the prior estimates. In this case MAP estimates are close to the ML estimates computed from the adaptation data, i.e., SD estimates.

MAP estimate θ_{MAP} is defined as

$$\theta_{MAP} = \arg\max_{\theta} f(x|\theta)g(\theta), \tag{4.1}$$

where $f(x|\theta)$ is the likelihood and $g(\theta)$ the prior distribution. The distribution parameters are denoted as $\theta = (w_1, ..., w_M, \boldsymbol{\mu}_1 ..., \boldsymbol{\mu}_M, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_M)$ for notational convenience. The parameters of the prior distribution are given

by the model chosen as the initial model, which is typically ML-trained SI model.

Multivariate Gaussian Mixture Model has joint probability density function

$$f(\mathbf{x}|\theta) = \prod_{t=1}^{T} \sum_{k=1}^{M} w_k N(\boldsymbol{x_t}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{4.2}$$

where $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}..., \boldsymbol{x_T})$ is sample of T i.i.d. observations drawn from a mixture of K p-dimensional multivariate normal densities. $\boldsymbol{\mu}_k$ notates mean, $\boldsymbol{\Sigma}_k$ covariance and w_k mixture weights. Weights w_k for k'th mixture component satisfy

$$\sum_{k=1}^{M} w_k = 1 \tag{4.3}$$

In MAP adaptation, a mean of a single mixture component is updated as following [53]

$$\hat{\boldsymbol{\mu}}_{map} = \frac{\gamma}{\gamma + \tau}\boldsymbol{\mu}_{ml} + \frac{\tau}{\gamma + \tau}\boldsymbol{\mu}_{prior}, \tag{4.4}$$

where $\boldsymbol{\mu}_{ml}$ is the new mean ML-estimate over the adaptation data according to Equation 3.3 and $\boldsymbol{\mu}_{prior}$ is the mean of the initial model. The weight of prior knowledge is adjusted empirically with τ. The occupancy of likelihood γ is defined as,

$$\gamma = \sum_{r=1}^{R} \sum_{t=1}^{T_r} L^r(t), \tag{4.5}$$

where L defines the likelihood probability in each sentence R in the data at time instant T.

As can be seen from the formulas, if the occupancy of the components is small, the MAP estimate will remain close to the mean of the initial model. On the other hand, if the triphone is well presented in the data, thus occupancy is large, MAP estimate is shifted more towards the ML estimate. Shifting can be constrained with weight parameter τ. The optimal τ depends on the initial model and data, and there is no closed form solution of finding the optimal value. Hence τ has to be determined empirically for each adaptation instance.

4.4.2 Maximum Likelihood Linear Regression

Instead of updating model parameters separately, an alternative approach is to use a linear transformations to adapt *acoustic classes*.

The advantage of the linear transform based adaptation methods is that all Gaussian parameters can be adapted with only a few transformations, or even with a single one. Each acoustic class has its own transformation. Acoustic class can be chosen to be an individual speaker, acoustic environment or even subset of phonemes. Regression trees are typically used to find suitable classes for phonemes [52]. As only a few transformations are needed, even a small amount of adaptation data is sufficient. The most popular linear transformation adaptation method is Maximum Likelihood Linear Regression (MLLR). MLLR is able to rapidly adapt even large models and is more suitable for real-time adaptation than MAP adaptation.

Usually only Gaussian means are adapted in MLLR adaptation, because the main differences between speakers can be assumed to be characterized by the mean [35]. However, it is possible also update covariances. Gaussian means μ are updated according to [52]

$$\hat{\boldsymbol{\mu}} = \boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b}, \tag{4.6}$$

where $\boldsymbol{A}$ is a $D \times D$ regression matrix and $\boldsymbol{b}$ is a D-dimensional vector. D is the dimension of the feature vector $\mathbf{o}$. Usually Equation 4.6 is expressed as

$$\hat{\boldsymbol{\mu}} = \boldsymbol{W}\boldsymbol{\xi}, \tag{4.7}$$

where $\boldsymbol{\xi}^T = [1 \quad \boldsymbol{\mu}_1 \quad \boldsymbol{\mu}_2 \quad ... \quad \boldsymbol{\mu}_n]$. Equation 4.7 is used because then the problem reduces of finding the optimal $D \times D$ regression matrix $\boldsymbol{W}$ that maximizes the likelihood of the adaption data with the transformed model. $\boldsymbol{W}$ has a closed form solution [35]. Speaker adaptation in MLLR can be seen as a transformation to a new acoustic space, where the space is a new speaker.

The adaptation of the means has the greatest impact on the accuracy, but it is possible also to adapt covariances with MLLR. The covariance adaptation does not affect the performance as much as means, but might slightly decrease WER [51]. Diagonal Gaussian covariances $\boldsymbol{\Sigma}$ can be updated by using the adapted means to compute a transformation matrix $\boldsymbol{H}$ with following equation [52]

$$\hat{\boldsymbol{\Sigma}} = \boldsymbol{L}\boldsymbol{H}\boldsymbol{L}^T, \tag{4.8}$$

where $\boldsymbol{L}$ is the Choleski factor of the original covariance $\boldsymbol{\Sigma}$.

In the normal MLLR adaptation, the tranformation matrix $\boldsymbol{W}$ for Gaussian means and $\boldsymbol{H}$ for covariances are estimeted separately. In Costrained

Maximum Likelihood Linear Regression (CMLLR), the same $D \times D$ matrix $\boldsymbol{A}$ and the addition vector $\boldsymbol{b}$ is used for the both means and covariances.

$$\hat{\boldsymbol{\mu}} = \boldsymbol{A}\boldsymbol{\mu} - \boldsymbol{b} \tag{4.9}$$

$$\hat{\boldsymbol{\Sigma}} = \boldsymbol{A}^T\boldsymbol{\Sigma}\boldsymbol{A} \tag{4.10}$$

The same transformation matrix $\boldsymbol{A}$ can be used to transform the feature vectors $\mathbf{x}$ instead of changing the acoustic model. In practice, adapting the acoustic features is often preferred. The feature vectors are adapted by the following equation

$$\hat{\mathbf{x}} = \boldsymbol{A}^{-1}\mathbf{x} + \boldsymbol{A}^{-1}\boldsymbol{b}. \tag{4.11}$$

Chapter 5

Implementation

The LVCSR system used in this thesis is called AaltoASR [24]. It has been developed in Speech recognition group of Aalto university, and source codes were recently published as an open source[1]. AaltoASR uses the typical framework of HMM-based ASR system described in Chapter 2. The original source codes of the system are written in C++, Perl and Python. The goal of this thesis was to implement MAP adaptation scheme into AaltoASR. Implementation was designed so that MAP estimates can be estimated in two different ways. The traditional way is using the occupancy for weighting the MAP estimate, but there is also an alternative to use weights of the mixture components instead.

In this chapter, the implementation of MAP and the usage is shortly introduced. Important functions and parameters, and how they affect the MAP adaptation are also presented. Parameter optimization is more closely examined in the next Chapter 6.

As explained in Section 4.4.1, MAP adaptation procedure shares many similarities to ML training. Both use the EM-algorithm as the basis. The E-step is the same in both procedures but the M-step, where estimates are computed, is different. Important source files for MAP implementation are `estimate.cc` and `Distribution.cc`. In each iteration of ML training, new model is estimated in `estimate.cc`, which performs the EM-algorithm. The model parameters are re-estimated by calling estimating functions in `Distribution.cc`, where all distribution related operations are computed. Training script `training.pl`, written in Perl, performs the whole training scheme by calling and combining C++ functions.

In ML training, the *initial model* is computed from the training data. In MAP adaptation, there usually already is a trained acoustic model needing

[1] https://github.com/aalto-speech/AaltoASR

adaptation which is used as the initial model. Both ML and discriminatively trained models are suitable as will be confirmed in the following chapter.

MAP adaptation utilizes training data for estimating MAP estimates. The data used for the adaptation is called *adaptation data*. The amount of the adaptation data can be less than in training, but the data has to be more homogeneous. The adaptation set is given to the system in a form of mono 16kHz wav-files, which are converted into feature vectors in a system. The combined length of training data should be at least few minutes long [48] for robust estimates, as is also confirmed in Chapter 6. Transcriptions are also needed because MAP is a supervised method.

In this MAP implementation, only model means are adapted. Other model parameters will simply stay the same. It is possible to deliver estimation formulas for other HMM parameters: covariances, mixture weights and transition probabilities. However, adaptation of these parameters were not implemented. It is still possible to compute new ML estimates from adaptation data for the other model parameters during MAP adaptation, but it is not recommend unless the size of the adaptation data is considerably large.

The training script has important parameters that will be introduced next. Following switches are important when calling `estimate.cc` in ESTIMATE_MODEL function of training script.

`-t`
Determines if the transition probabilities are updated or not. The transition probabilities can be updated only by ML estimates, hence the switch should be left off.

`--no-mixture-update`
If the switch is on, the size of the mixture does not change, i.e., splitting, merging and mixture weight update are not performed. If the model size reduction is desired this switch should be off, otherwise switch should be on, because ML estimates will break the model if the size of adaptation set is not sufficient.

`--maptype $VALUE`
There are two different styles of MAP adaptation. If the implementation that utilizes mixture weights is to be used, `--maptype` should be set to 1, and if the occupancy implementation is wanted, switch should be set to 2.

`--tau $VALUE`
Determines the value of prior weight parameter τ. Default value is set to 5.

New means are estimated for each Gaussian distribution in MAP adaptation. Estimates of mean and covariance are computed in function GAUSSIAN::ESTIMATE_PARAMETERS, which is located in `Distribution.cc`. Mean estimation of MAP was also implemented there as presented in Algorithm 1 and 2.

Algorithm 1: MAP algorithm using occupancy

input : τ, prior means $\boldsymbol{\mu}_{prior}$, covariances $\boldsymbol{\Sigma}_{prior}$, occupancy γ
output: MAP adapted means μ_{map}

foreach *mixture* m **do**
  foreach *gaussian* i **do**
    $\gamma =$ `GetGaussianOccupancy(`i`)` ;
    $\mu_{ml} =$ `GetMLestimate(`i`)` ;
    if $\gamma >= 0$ **then**
      $\mu_{map} = (\tau \times \boldsymbol{\mu}_{prior}[m,i] + \gamma \times \boldsymbol{\mu}_{ml}[m,i])/(\tau + \gamma)$;
    else
      $\mu_{map} = \mu_{prior}$;
    $\boldsymbol{\Sigma}_{map} = \boldsymbol{\Sigma}_{prior}$;

Gaussian mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ are stated as a vector and matrix because features are presented as a vector. In case of AaltoASR, mean is a vector and covariance is a diagonal matrix with 39 dimensions. Larger dimensions require more computational capacity, hence linear algebra operations are computed utilizing Lapack++ library that is designed for computing large linear algebra operations.

The more data is used in training, the higher the value of the occupancy. Each state is modeled with GMM and the occupancy of the state is divided for each component. When the occupancy for a certain component reaches the predetermined limit, the distribution will split into two new Gaussian distributions. In AaltoASR, the limit is set to 200 by default, thus occupancy values vary from 0.001 to 200. However, there is a limit how many components each mixture can have, which is why the occupancy for popular states will increase without upper limit causing problems in occupancy based MAP adaptation. These states are fortunately rare. In the occupancy based MAP adaptation procedure, large occupancy values cause problems because τ has to be increased as well for adaptation to function properly. Hence, the optimal value of τ is dependent of the amount of adaptation data used. This problem is more closely experimented in Section 6.2.

Book *Techniques for noise robustness in automatic speech recognition* by Virtanen et. al. [51] presents that instead of the occupancy, mixture compo-

nent weights could be used to weight MAP estimates. In this MAP implementation, only the components whose newly computed ML weights reach a certain limit are updated. The updated mean is an average mean between new ML estimate and prior weighted by mixture weights. The implementation is presented in Algorithm 2.

$$\hat{\boldsymbol{\mu}}_{map} = \frac{w}{w+\tau}\boldsymbol{\mu}_{ml} + \frac{\tau}{w+\tau}\boldsymbol{\mu}_{prior}. \tag{5.1}$$

Mixture weights, c_k, do not change during the adaptation, hence they are only dependent on the initial model instead of the amount of adaptation data. In addition, weights do not have as large interval as occupancy, because they have restriction $\sum_{k=1}^{M} c_k = 1$, hence the highest possible weight is 1. Therefore, in theory it is easier to find balance between new and prior estimates.

There is, however, drawback of using mixture component weights instead of occupancy. Mixture weights only inform which of the components were common in the training data of the original model. Hence, less information of adaptation data is used. Even if some component was common in training data set, it can be rare in the adaptation data set, which causes that poor estimate may be given more weight in adaptation than deserved.

In AaltoASR, the maximum number of components in the mixture can be chosen freely, but usually 80 is used. For the large SI model, the mixture size is typically from 20 to 50, which is why the size of the weights is usually under 0.1. Therefore, the value for τ should also be below 1.

Algorithm 2: MAP algorithm using mixture component weights

input : τ, prior means $\boldsymbol{\mu}_{prior}$, covariances $\boldsymbol{\Sigma}_{prior}$, mixture component weight w
output: MAP adapted means μ_{map}

foreach *mixture* m **do**
  foreach *gaussian* i **do**
    $w =$ `GetMixtureCoefficient(`i`)` ;
    $\mu_{ml} =$ `GetMLestimate(`i`)` ;
    if $w > \tau/10$ **then**
      $\mu_{map} = (\tau \times \boldsymbol{\mu}_{prior}[m,i] + w \times \boldsymbol{\mu}_{ml}[m,i])/(\tau + w)$;
    else
      $\mu_{map} = \mu_{prior}$;
    $\boldsymbol{\Sigma}_{map} = \boldsymbol{\Sigma}_{prior}$;

Chapter 6

Evaluation

In this chapter MAP adaptation implementations are tested and evaluated. Two MAP versions were implemented into the system which were introduced in the previous chapter. In the experiments, the implementation that uses the occupancy, γ-MAP, is mostly used. The implementation that utilizes mixture weights, w-MAP, is only examined briefly in Section 6.2.

Speaker adaptation is used to evaluate the performance of MAP in Section 6.1. The amount of the adaptation data and the prior weight τ are both important parameters and affect directly adaptation performance. The relationship of these two parameters are examined as well.

The results of speaker adaptation is presented in Section 6.1. We examine how the size of adaptation data set, the value of the prior weight parameter τ and the number of iterations affect the adaptation. In addition, we investigate if it is possible to reduce the size of the model with MAP adaptation while achieving adequate performance.

In Section 6.2, MAP is applied into more general task. We examine if MAP adaptation can be used to give more weight to a certain corpus if multiple corpora are used in the training. In addition, we investigate if it is possible to train acoustic model better suited for spoken language by utilizing MAP adaption to adapt spoken language to general SI model.

6.1 Speaker Adaptation

It was necessary to confirm that implementation of MAP adaptation works correctly. Speaker adaptation was selected for this purpose. Speaker adaptation is relatively simple compared to, e.g., speech style adaptation. The used adaptation data is typically homogeneous, therefore adaptation more certainly improves results. MAP adaptation is an old method. Its theory is

well known and many times tested, hence we had a strong hypothesis how MAP adaptation should perform in speaker adaptation compared to SD and MLLR adapted models.

All the following results were produced with traditional MAP implementation, γ-MAP, which uses occupancy values to determine the weight between new and old estimate. The second implementation version, w-MAP, is not included due to poor results. Further analysis of w-MAP is included in the Section 7.

Experiment requires a large amount of one speaker's speech. Audio book *Syntymättömien sukupolvien Eurooppa* by Eero Paloheimo is read by a single person, hence it was chosen for the experiment. 7 hours of the material was used for the experiments. The data was divided into two; training set of 5 hours and evaluation set of 1.5 hours. Remaining 30 minutes were left for developing purposes. The training set was used to train SD models and to adapt SI models. The results were evaluated with the evaluation set.

Following models were used in speaker adaptation experiments. All acoustic models are ML trained if not stated otherwise.

- Language model: `morph40000`.
  Model was trained with news texts. The vocabulary size is 8429.

- SI Acoustic model: `speecon_all_multicondition_ml`.
  Trained with Speecon corpus [27] including 60-90 hours of speech. Speecon includes mainly standard language with varying acoustic conditions, hence model is a quite robust for environment changes. Model has 40 000 Gaussian components. This model is used as an SI initial model in these experiments if not stated otherwise.

- SI Acoustic model: `speecon_all_multicondition_mmi`.
  Model is discriminatively trained with MMI estimates. The training set and model size is the same as in ML trained SI acoustic model above. [41–43]

- SD model: Different-sized SD models were trained with audio book *Syntymättömien sukupolvien Eurooppa*. The model sizes vary from 4000 to 6000 Gaussians depending on the size of the training set which was from 90 minutes to 180 minutes.

The size of the adaptation set

The size of the adaptation set has a great impact on how much the model improves. The size also affects which adaptation method is the most reasonable choice. Figure 6.1 shows how larger set improves WER for MAP, MLLR

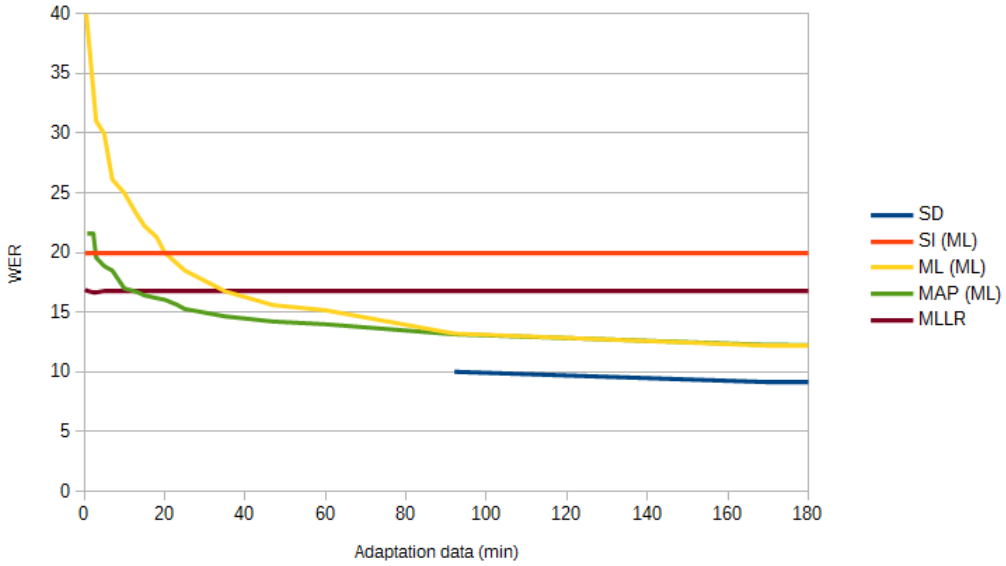

Figure 6.1: The size of the adaptation data affects differently depending on the adaptation method. X-axis contains the size of the adaptation set, and y-axis Word Error Rate. MAP adaptation is drawn in green, MLLR adaptation in purple and adaptation updating means with ML estimates is in yellow. Blue line represents the size of the training set of Speaker Dependent model (SD). Red line of SI model is only for reference so that it is possible to compare adaptation to the original model.

and ML adaptation. ML adaptation has the same procedure as MAP adaptation, but instead of MAP estimates, new ML estimates are simply used. SD model (blue) is also shown in the figure, but as SD training requires at least 90 minutes of data, results are not reported before the 90 minute time instance. As the training data of SD model increases, the performance of the model improves steadily. For the reference, the performance of the SI model before adaptation is marked with a red line. In this experiment τ was set to 10.

MAP adaptation improves as the adaptation set increases, which was expected. If the length of the adaptation set is less than 2 minutes, MAP adaptation produces worse mean estimates than original SI model has, due to the bad ML estimates. MAP estimates are computed as a weighted mean between ML mean estimates and SI model means. If there is not enough data for the training, ML estimates are not able to model the data well.

Figure 6.1 shows that when ML estimates were trained with only 30 seconds of the speech, WER of MAP adaptation was over 40%. This is way worse than with MAP estimates (WER=22%). The difference of these results show the importance of using prior information in adaptation. However, as the adaptation data increases, ML estimates eventually reach the MAP estimates. Interestingly, the point when ML estimates perform as well as MAP estimates is the same as when SD model training is possible.

MAP adaptation converges towards the SD model, but never reaches it. It may be because only mean parameters are updated. If all HMM parameters were updated, the adaptation should improve even further [48].

Comparing MAP and CMLLR

In these experiments, linear transformation adaptation method CMLLR with a single transformation matrix was used. CMLLR is fundamentally different from MAP as it transforms feature vectors instead of model parameters. CMLLR was able to improve WER even with a small amount of data, as can be seen from Figure 6.1. However, it does not improve remarkable after the adaptation set grows. WER stays around 16.70% even if amount of data is increased from one minute to ten minutes or more. At first CMLLR adaptation improves the model more, but after ten minutes the performance of MAP is better. Hence, it can be concluded that if we are able to get over 10 minutes of the adaptation data, MAP adaptation should be used instead of CMLLR.

If more than one transformation matrix were used, CMLLR would be able to better utilize the increase in the data. CMLLR with multiple transformation matrices is typically regression tree based [52]. This type of CMLLR is also available in AaltoASR, but due to the lack of time and proper documentation, CMLLR of single transformation was only tested.

It is possible to adapt discriminative models as well. MAP adaptation decreases WER from 18.85% to 16.73%, when τ is 5 and the size of the adaptation data 7 minutes. However, when CMLLR was tested, WER increased instead. The problem might be either in the used discriminative model or with the AaltoASR.

Optimal τ

The prior weight parameter τ (Equation 4.4) has to be determined empirically, since there is no common method or closed form equation to compute the optimal value. The optimal value of τ depends on the initial model and the size of the adaptation set. If the τ is set poorly, the adaptation can

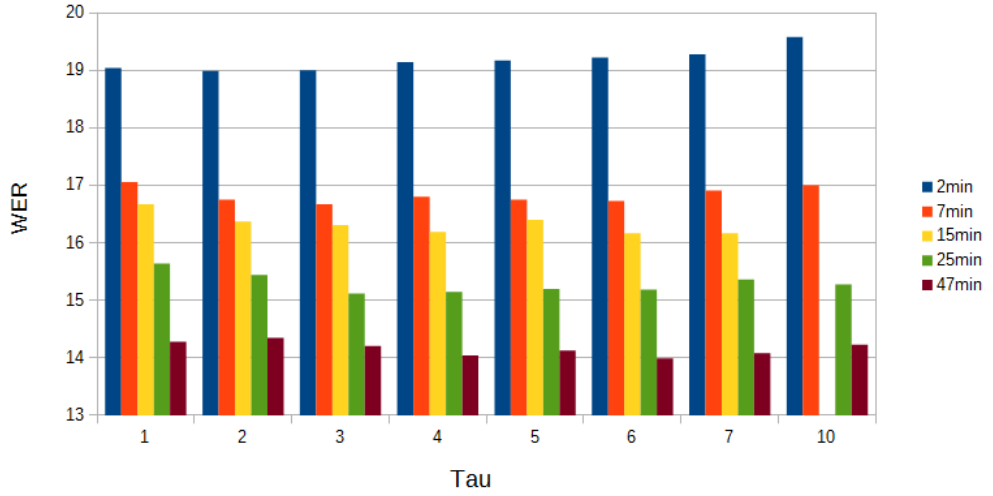

Figure 6.2: The value of τ and the size of the adaptation data (colors) affect WER.

make the model perform worse than before adaptation. For this reason, we experimented if there was some optimal value for τ in general level, and how it depends on the size of the adaptation data set. In Figure 6.2 the results are presented. The figure shows how WER correlates with the change of τ (x-axis) and the size of adaptation data set (colors). It can be seen that there is no single optimal value, but rather an interval. In addition, the optimal interval of τ seems to shift to larger values as the amount of data increases.

If the adaptation data set is large, the occupancy values tend to be large as well, meaning that the new estimates have more weight in MAP estimate if the prior weight is not increased as well with change of τ. The interval shift might imply that there is some kind of ideal balance between new and prior estimates. Less occupancy is accumulated from smaller data sets. Therefore the increase in τ has more influence in MAP estimates and causes WER to converge faster to SI model when the adaptation set is small.

Iterations

As MAP estimates can be used in a training as well, it is interesting to see if the number of the iterations have any effect in adapting process. As can be seen from Figure 6.3, iteration was not such a good idea. The performance of adapted model worsen with each iteration. The experiments were done

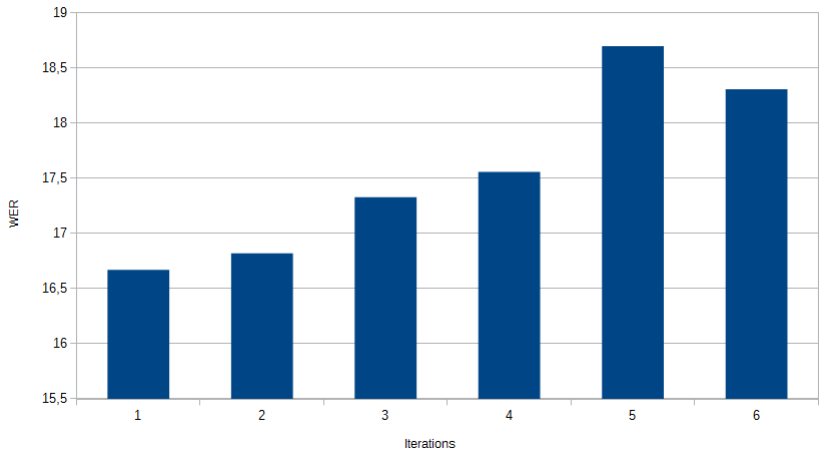

Figure 6.3: Multiple MAP adaptation iterations affecting the WER of the model

with adaptation set of 10 minutes and with $\tau = 30$. Maybe the value of τ should have been even higher, in order to the model of previous iteration to have more weight as now model might have been overlearning the small data set.

Reducing model size

MAP adaptation can be used to reduce the model size. If the adaptation data set is small compared to the training set, the occupancy will not accumulate for every Gaussian component in the model. In AaltoASR, if Gaussian does not have occupancy it is merged into other Gaussian distributions. If merging is not switched off, it reduces the model size to half of the original size in each iteration, until there is enough occupancy for all Gaussians.

Iterative MAP adaptation does not improve the performance, as was seen in Figure 6.3 At least if the model size stays the same. However, if components with poor MAP estimates are removed during the adaptation, the MAP adaptation should perform better. The hypothesis was that if the model size is reduced, WER after adaptation will stay at least the same.

Figure 6.4 shows that hypothesis was correct. However, surprisingly the difference between reduced model and normally adapted model was not large. The figure shows WER for normally MAP adapted model of 40 000 components (blue) and for model of which size was reduced to 20 000 components

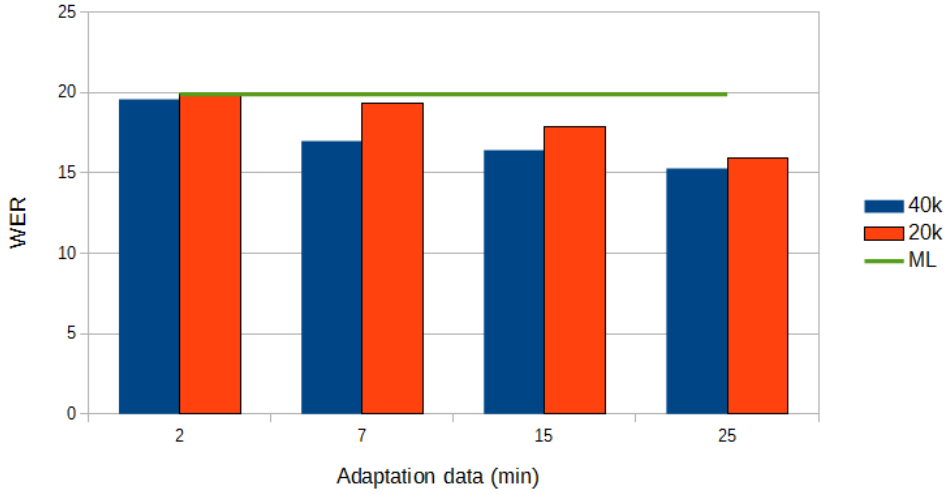

Figure 6.4: When the model size can change

with MAP adaptation (red). Green line shows WER of the SI model before adaptation.

MAP adaptation improves the SI model, even if the model size is reduced. Surprising was that the difference between reduced and unreduced adapted models was smaller than expected. Especially when the adaptation set was 2 minutes, the difference between models was only 0.4%. The unreduced model is able to utilize the smaller amount of data better than reduced model. The difference is again reduced, when the size reached 25 minutes. The improvement of reduced model as the adaptation set increases was expected, because MAP estimates converge closer to SD model estimates.

6.2 Colloquial Language Adaptation

The reason why MAP adaptation was chosen as the topic of this thesis was because we wanted to develop better acoustic models for spoken language recognition. Colloquial language has a lot of internal variation which makes developing general models difficult task. In addition, Finnish spoken language corpora are scarce.

Acoustic model for spoken language has been trained previously with four corpora. Some of these corpora represent the natural spoken languages better than others. Because each of these corpora are small, it is not possible to

train a model with only one of them. But we would like to give more weight to the more suitable data to make better model for natural spoken language. The hypothesis was that maybe it is possible to give more weight to certain corpora utilizing MAP adaptation after the training.

Following Finnish spoken language corpora were used in the experiments. All the files are in Microsoft WAVE format with sample format of 16 kHz 16-bit PCM.

- **DSP** corpora includes conversations of 117 students. In total there are 2532 utterances in 5 hours of audio. DSP material is the most natural one of these corpora. [1]

- **FinDia** includes ten spontaneous dialogues between friends, with the duration ca 45 minutes each, and 7-8 hours in total. [2]

- **RadioCon** corpus includes four radio recordings:

  Aamushow: Two episodes of Bassoradio Aamushow, containing 202 utterances, 11 minutes of audio, from three different speakers in total.
  PuhujainKulma: Two episodes of PuhujainKulma podcast, containing 112 utterances, 8 minutes of audio, from three different speakers in total.
  Kultabassokerho: One episode of Bassoradio Kultabassokerho, containing 100 utterances, 10 minutes of audio, from three different speakers in total.
  FYM-Podcast: One conversation from Radio Free Your Mind between two speakers, containing 26 utterances, 1.5 minutes of audio.

- **Speecon** is large corpus from 60 to 90 hours of speech. It mostly includes standard language, but ten hours (5499 utterances) of it resembles spoken language. [27]

Colloquial language corpora were divided into training and evaluation set. Evaluation set includes following data sets: Aamushow, Puhujainkulma and 25 minutes of DSP corpora. Colloquial language acoustic model `puhekieli2014` was trained with the rest of the data. The model `puhekieli2014` was ML trained resulting in 40 000 individual mixture components. In addition Speecon model `speecon_all_multicondition_ml` was used in the experiments. This model was described more closely in previous section.

Colloquial language recognition is not only acoustic problem but linguistic as well, hence language model has important role in recognition. The

[1] META-SHARE: `http://urn.fi/urn:nbn:fi:lb-20150123` (2013-2014)
[2] META-SHARE: `http://urn.fi/urn:nbn:fi:lb-20140730194`

language model used in experiments designed for recognizing Finnish spoken language [34].

Experiment started by training acoustic model with all of the corpora. After training, the model was MAP adapted with the corpus we wanted to give more weight to. DSP corpus resembles the most natural spoken language, hence it was chosen as the adaptation data.

Table 6.1 shows that weighting the model with DSP corpora improves WER, though the improvement is small. The overall WER of all evaluation sets does not change, but set containing part of DSP corpora shows improvement. The WER of DSP set can be improved even further to 45%, but the more weight we give to the DSP corpus, the more WER of PuhujainKulma set increases, which causes increase in total WER.

Table 6.1: Adapting models with DSP corpus. The evaluation set consists of three sets Aamushow (aam), PuhujainKulma (puh) and part of DSP (dsp) corpus. Acoustic model Puhukieli2014 was adapted with two different MAP implementation. w-MAP uses mixture component weights to determine weight between estimates and γ-MAP utilizes distribution occupancy.

Model	WER			
	aam	puh	dsp	total
Puhekieli2014	53.4	59.4	49.5	52.6
DSP	54.0	61.1	51.1	53.9
Puhekieli2014+w-MAP ($\tau = 0.5$)	53.3	59.1	48.8	52.1
Puhekieli2014+γ-MAP ($\tau = 90$)	53.7	59.5	48.6	52.1

The reason why adapting DSP set increases WER of PuhujainKulma set might be because PuhujainKulma has clearly less noise than the other two. Hence, instead of speaking style, we may be actually adapting recording conditions.

Because the results did not improve as much as expected with γ-MAP implementation, w-MAP implementation was also tested. Surprisingly both implementations performed evenly. However, one important thing to notice is that in γ-MAP case, optimal τ is 90, while in speaker adaptation the optimal value was around 5. w-MAP, on the other hand, τ was set to value of 0.5 in both experiments. The other reason for testing w-MAP was due to the optimal value of τ is easier to find. However, further testing should be made between differences of the two implementation.

We also adapted Speecon SI model `speecon_all_multicondition_ml` with DSP data set. The reason behind this experiment was the hypothesis

that using a robust model as a base, the resulting model is better. However, this was not a case. We could not improve the overall WER of SI model, which was 64.1%.

In speaker adaptation experiments duration models were updated. Duration model models the duration of phonemes. However, in SI model adaptation, this setting had to be turned off because otherwise WER after adaptation increased to 70%. In speaker adaptation, updating duration model did not affect the results. This was interesting result as it implies that duration model has greater impact in colloquial language recognition than in speaker adaptation.

Chapter 7

Discussion

At first, MAP adaptation seemed a simple task to implement and test. However, as I begun working, the complex relationships between the different parts of the training scheme surprised me. The further I tested more and more interesting things to investigate appeared. It was, unfortunately, impossible to experiment them all. In the following sections, I explain the limits of the experiments introduced in the previous chapter that should be taken into consideration when interpreting the results. Lastly, I present what should be researched in the further in Section 7.3.

7.1 Speaker Adaptation

Speaker adaptation was used to confirm that implementation was working properly. The used corpus was large, consisting of 7 hours of speech from a single person. The corpus was an audio book, hence the speech was clearly pronounced, however, the vocabulary was at times difficult and varied from chapter to chapter. We assumed that the corpus was homogeneous. This was actually not the case. Each chapter had a different topic, hence the difficulty of vocabulary varied. Some chapters had a lot of technical words and foreign names. The error rates varied from 10% to 40% depending on the chapter. The variation was taken into account by using a large evaluation set consisting of few different chapters and by keeping the adaptation set always the same.

Speaker adaptation was tested only with one speaker. To confirm that adaption works in general sense, more speakers actually should be tested. But as the corpus was large, it was concluded that MAP adaptation will work for other speakers as well. There are not many corpora available in Finnish that have large amount of speech from a single person. However, if

there had been more time, speaker adaptation could have been tested with multiple speakers using smaller adaptation sets.

Only one SI acoustic model was used as an initial model for adaptation. The model was relatively large and robust SI model. Smaller models should also have been tested in order to moderate if MAP adaptation would have behaved differently. The model size of the initial model is crucial as it affects directly to τ, because the occupancy of adaptation set is allocated to all model components. If the number of the components is large, the occupancy is spread more widely and has in average smaller value for each component.

Most of the model and training parameters were not optimized, because finding optimal parameters is time-consuming. LM-scale affects WER a lot and it was kept constant (35) in all experiments. Probably smaller LM-scale would have been better, since the language model used in speaker adaptation had been trained with news text corpora instead of similar audio books.

Corpus included also foreign names. Foreign words always increase WER because the Finnish phonemes and phonemes in foreign languages are different. In the audio book, the reader reads foreign words as close to the original pronunciation as she could. Because of this, she used phonemes that are absent from the Finnish language, hence which are also absent from the models.

Systematic and random errors caused by language model do not affect to analysis of the results, because the relative gains are compared to each other in the experiments. Corpora-based errors on the other hand is minimized by using sufficiently large training and evaluation sets.

MAP adaptation is well-suitable for reducing model size. In my opinion, it is the best application for MAP implementation, based on the experiments. It is a great advantage to be able to reduce model size by half and keep the same performance, or even improve it. In practice, the model size is crucial, because it affects directly how fast ASR system performs. The less user needs to wait for recognition results, the better. However, limits of splitting and merging operations could have been tested more carefully. For example, how much the model size should be reduced and when training SD model is more efficient.

7.2 Colloquial Language Adaptation

Not many spoken language corpora in Finnish are available. Acoustic model of spoken language used in the experiments was trained with four corpora. The corpora have distinct recording and acoustic conditions which causes variation, but there is also a problem that spoken language itself has a lot of

variability. Hence, training data is far from homogeneous thus model does not perform adequately.

Colloquial language adaptation was included in the thesis because the original idea behind implementing MAP adaptation was to improve acoustic models for spoken language. Because all corpora were not equal quality, we wanted to give more weight to certain corpora utilizing MAP adaptation.

DSP corpus was chosen as the adaptation set because DSP is the closest to natural spoken language. DSP may not have been the most appropriate choice, because the corpus is quite difficult. Corpus includes files from multiple sessions which have been recorded with different types of microphones. In addition, there is background noise and the vocabulary is not restricted in any way. Hence it is all in all a difficult corpus. For testing purposes it may have been too difficult and smaller and more homogeneous set should have been used. As for now, it was not clear if the speaking style or acoustic environment was adapted. In the experiments, after model was adapted with DSP set, the WER of clean corpus increased, while WER of the set with more background noise stayed the same. This implies that acoustic environment might have been adapted. But with these experiments alone, we cannot be certain what part was actually adapted. It is highly possible that unknown factors affect adaptation. Further testing, hence, is needed.

Experiments showed that MAP is usable for weighting the corpora after the model training. Adapting model again with the most important training material seems make model more specialized in that data set. The specialization causes performance to decrease in data that do not resemble the adaptation set, which was seen in the experiments.

Difficult acoustic properties are not the only difficult aspect in the spoken language. Language modeling is difficult as well and causes high error rates if not modeled properly. In Finnish, written and spoken languages differ greatly from each other. Not only duration and pronunciation, but the grammar and vocabulary is also different. By developing language model, WER could be improved.

7.3 Future work

It was noticed during the spoken language adaptation experiments that duration models affect the results greatly. Duration of the phonemes is important to model in Finnish with duration modeling, because meaning of the word can be changed to another by changing the duration of a single phoneme.

In speaker adaptation, the speaker spoke standard language. Duration models were updated as well during the adaptation and they did not affect

performance of the system. It was not until the spoken language adaptation experiments, when the problems arose. The adapted model performed much worse than originally when the duration model was updated. This suggests that the duration models are more important in colloquial language and should be trained properly with large training set. However, the effects of the duration model were not the main point in this thesis and was just tested by an accident because the duration model updating was accidentally left on in MAP implementation. Hence, the reasons behind why duration modeling affects the colloquial language more than the standard is not analyzed in this thesis.

Only mean update was implemented in MAP adaptation. While mean is the most important HMM parameter to update, updating others as well would improve adaptation. I tested updating ML covariance estimate during the MAP adaptation. By updating covariance with ML estimate, the results were improved if the adaptation set was at least few hours. In speaker adaptation updating the covariances only made the performance worse, because there was not enough data to compute robust ML estimates. However, when the model was adapted with DSP set, updating covariances as well improved the WER of DSP evaluation data to 46%. This suggests that MAP implementation could be improved further by adding at least covariance update.

Chapter 8

Conclusions

MAP adaptation is one of the most common adaptation schemes in the speech recognition. The theory behind MAP adaptation is relatively simple and easy to apply, but it had not yet been implemented into AaltoASR, thus it was a great topic for this thesis.

Two versions of MAP adaptation were added into the system. The first one is traditional and utilizes occupancy to compute the MAP estimate. Occupancy describes the probability mass of the distributions which were estimated from the adaptation data. The second one uses mixture weights instead. It was concluded that occupancy-based implementation improved WER more, however, the optimal parameter for τ was more difficult to find. More testing for the mixture weight-based implementation is needed.

Because of the long history, MAP has been compared to MLLR in many experiments and the differences of these methods are commonly known. Hence, to proof that the MAP implementation performs correctly, the comparison with CMLLR was chosen for the experiment. CMLLR adaptation used utilizes single transformation to the all features.

From the adaptation tasks, speaker adaptation is the easiest. In speaker adaptation, the adaptation data is highly homogeneous and the model can be improved even with a small amount of the data. In addition, the performance of adaptation can be compared to SD acoustic model trained with the same adaptation data. Speaker adaptation is the most common adaptation style, hence, it is known how the performance of MAP and MLLR changes with the size the adaptation data. For these reasons the speaker adaptation was chosen as the main experiment for this theses.

Speaker adaptation experiments confirmed that implementation was working correctly, i.e., MAP adaptation improved SI model and the error rate decreased as the adaptation data set grew. When MAP was compared to MLLR adaptation, it was noticed that MLLR was more suitable in situa-

tions where the adaptation set was less than 10 minutes. However, when the set was over 10 minutes, MAP adaptation benefited the additional data more than MLLR. Hence, MAP should be used if there is over 10 minutes of data available.

There is also other ways to apply MAP adaptation. It was noticed that MAP could be used to reduce the model size. Model size reduction improves the efficiency of the model, and thus the efficiency of the whole system making the recognition faster. Colloquial language adaptation was investigated as well, but we could not deduct if the speaking style or acoustic environment was adapted based on the experiments. However, it seemed that MAP implementation can be used to give more weight to some data set after the acoustic model training.

Although MAP adaptation is an old method, surprisingly, I was unable to find much information about how the prior weight parameter τ affects adaptation and how the optimal value can be found. Hence it needed to be investigated. The optimal value seems to depend on the size of the model and the adaptation data and it is quite hard to have a good suggestion for the optimal value based on the experiments. For speaker adaptation, however, I would suggest using $\tau = 5$. For the spoken language adaptation, the optimal was $\tau = 90$. Higher value was needed because the adaptation data set is large. If the length of adaptation set is more than an hour, I recommend to try different values of τ to find the optimal. More experiments are needed to see if there are any patterns between model size and the size of adaptation set in order to make any recommendations for more complicated adaptations.

Bibliography

[1] Atal, B. S., and Hanauer, S. L. Speech analysis and synthesis by linear prediction of the speech wave. *The Journal of the Acoustical Society of America 50*, 2B (1971), 637–655.

[2] Bahl, L., Brown, P., de Souza, P. V., and Mercer, R. Maximum mutual information estimation of hidden markov model parameters for speech recognition. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'86.* (1986), vol. 11, IEEE, pp. 49–52.

[3] Baum, L. E. An equality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities 3* (1972), 1–8.

[4] Baum, L. E., Eagon, J. A., et al. An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology. *Bull. Amer. Math. Soc 73*, 3 (1967), 360–363.

[5] Baum, L. E., and Petrie, T. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics* (1966), 1554–1563.

[6] Baum, L. E., Petrie, T., Soules, G., and Weiss, N. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics* (1970), 164–171.

[7] Baum, L. E., and Sell, G. Growth functions for transformations on manifolds. *Pac. Math 27*, 2 (1968), 211–227.

[8] Benzeghiba, M., De Mori, R., Deroo, O., Dupont, S., Erbes, T., Jouvet, D., Fissore, L., Laface, P., Mertins, A., Ris, C., et al. Automatic speech recognition and speech variability: A review. *Speech Communication 49*, 10 (2007), 763–786.

[9] Bourlard, H. A., and Morgan, N. *Connectionist speech recognition: a hybrid approach*, vol. 247. Springer Science & Business Media, 1994.

[10] Davis, K. H., Biddulph, R., and Balashek, S. Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America 24*, 6 (1952), 637–642.

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

[12] Denes, P. The design and operation of the mechanical speech recognizer at university college london. *Journal of the British Institution of Radio Engineers 19*, 4 (1959), 219–229.

[13] Dudley, H., Riesz, R., and Watkins, S. A synthetic speaker. *Journal of the Franklin Institute 227*, 6 (1939), 739–764.

[14] El Ayadi, M., Kamel, M. S., and Karray, F. Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition 44*, 3 (2011), 572–587.

[15] Fletcher, H. The nature of speech and its interpretation1. *Bell System Technical Journal 1*, 1 (1922), 129–144.

[16] Forney Jr, G. D. The viterbi algorithm. *Proceedings of the IEEE 61*, 3 (1973), 268–278.

[17] Gales, M., and Young, S. The application of hidden markov models in speech recognition. *Foundations and Trends in Signal Processing 1*, 3 (2008), 195–304.

[18] Goodman, J. T. A bit of progress in language modeling. *Computer Speech & Language 15*, 4 (2001), 403–434.

[19] Graves, A. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711* (2012).

[20] Graves, A. *Supervised sequence labelling with recurrent neural networks*, vol. 385. Springer, 2012.

[21] Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning* (2006), ACM, pp. 369–376.

[22] Hermansky, H. Perceptual linear predictive (plp) analysis of speech. *the Journal of the Acoustical Society of America 87*, 4 (1990), 1738–1752.

[23] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE 29*, 6 (2012), 82–97.

[24] Hirsimaki, T., Pylkkonen, J., and Kurimo, M. Importance of high-order n-gram models in morph-based speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on 17*, 4 (2009), 724–732.

[25] Hirsimäki, T. *Advances in unlimited-vocabulary speech recognition for morphologically rich language*. Teknillinen korkeakoulu, 2009.

[26] Huang, X., Acero, A., Hon, H.-W., and Foreword By-Reddy, R. *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice Hall PTR, 2001.

[27] Iskra, D. J., Grosskopf, B., Marasek, K., van den Heuvel, H., Diehl, F., and Kiessling, A. Speecon-speech databases for consumer devices: Database specification and validation. In *LREC* (2002).

[28] Jelinek, F., Bahl, L., and Mercer, R. Design of a linguistic statistical decoder for the recognition of continuous speech. *Information Theory, IEEE Transactions on 21*, 3 (1975), 250–256.

[29] Juang, B.-H. On the hidden markov model and dynamic time warping for speech recognition—a unified view. *AT&T Bell Laboratories Technical Journal 63*, 7 (1984), 1213–1243.

[30] Juang, B.-H. Maximum-likelihood estimation for mixture multivariate stochastic observations of markov chains. *AT&T technical journal 64*, 6 (1985), 1235–1249.

[31] Juang, B.-H., Levinson, S. E., and Sondhi, M. M. Maximum likelihood estimation for multivariate mixture observations of markov

chains (corresp.). *Information Theory, IEEE Transactions on 32*, 2 (1986), 307–309.

[32] Juang, B.-H., and Rabiner, L. R. Automatic speech recognition–a brief history of the technology development. *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara 1* (2005).

[33] Karjalainen, M. *Kommunikaatioakustiikka*. Helsinki University of Technology, 2008.

[34] Kurimo, M., Enarvi, S., Tilk, O., Varjokallio, M., Mansikkaniemi, A., and Alumäe, T. Modeling under-resourced languages for speech recognition. *Submitted to Language Resources and Evaluation (LRE) Special Issue on Under-resourced Languages* (2015).

[35] Leggetter, C. J., and Woodland, P. C. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech & Language 9*, 2 (1995), 171–185.

[36] Levinson, S. E., Rabiner, L. R., and Sondhi, M. M. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *Bell System Technical Journal, The 62*, 4 (1983), 1035–1074.

[37] Li, J., Deng, L., Gong, Y., and Haeb-Umbach, R. An overview of noise-robust automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 22*, 4 (2014), 745–777.

[38] Liu, D., Kiecza, D., Srivastava, A., and Kubala, F. Online speaker adaptation and tracking for real-time speech recognition. In *INTERSPEECH* (2005), pp. 281–284.

[39] Lombard, E. Le signe de l'elevation de la voix. *Ann. Maladies Oreille, Larynx, Nez, Pharynx 37*, 101-119 (1911), 25.

[40] Markov, A. A. Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga. *Izvestiya Fiziko-matematicheskogo obschestva pri Kazanskom universitete 15*, 135-156 (1906), 18.

[41] Pylkkonen, J., and Kurimo, M. Analysis of extended baum–welch and constrained optimization for discriminative training of hmms. *Audio, Speech, and Language Processing, IEEE Transactions on 20*, 9 (2012), 2409–2419.

[42] Pylkkönen, J., and Kurimo, M. Improving discriminative training for robust acoustic models in large vocabulary continuous speech recognition. In *INTERSPEECH* (2012).

[43] Pylkkönen, J., and Kurimo, M. Optimization-based control for the extended baum-welch algorithm. In *INTERSPEECH* (2012).

[44] Pylkkönen, J. *Towards efficient and robust automatic speech recognition: decoding techniques and discriminative training.* Aalto University, 2013.

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

[46] Rabiner, L., and Juang, B.-H. An introduction to hidden markov models. *ASSP Magazine, IEEE 3*, 1 (1986), 4–16.

[47] Sakai, T., and Doshita, S. The phonetic typewriter. In *IFIP Congress* (1962), pp. 445–450.

[48] Sharma, H. V., and Hasegawa-Johnson, M. State-transition interpolation and map adaptation for hmm-based dysarthric speech recognition. In *Proceedings of the NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies* (2010), Association for Computational Linguistics, pp. 72–79.

[49] Sorenson, H. W., and Alspach, D. L. Recursive bayesian estimation using gaussian sums. *Automatica 7*, 4 (1971), 465–479.

[50] Turunen, V. T., et al. *Morph-based speech retrieval: Indexing methods and evaluations of unsupervised morphological analysis.* Aalto University, 2012.

[51] Virtanen, T., Singh, R., and Raj, B. *Techniques for noise robustness in automatic speech recognition.* John Wiley & Sons, 2012.

[52] Woodland, P. C. Speaker adaptation for continuous density hmms: A review. In *ISCA Tutorial and Research Workshop (ITRW) on Adaptation Methods for Speech Recognition* (2001).

[53] Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., et al. *The HTK book*, vol. 2. Entropic Cambridge Research Laboratory Cambridge, 1997.