Muhammad Hashim Abbbas

Confidence Scoring and Speaker Adaptation in Mobile Automatic Speech Recognition Applications

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 23.01.2017

Thesis supervisor:

Prof. Mikko Kurimo

Thesis advisors:

M.Sc. Matti Varjokallio

M.Sc. Leo Hämälainen

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Muhammad Hashim Abbbas

Title: Confidence Scoring and Speaker Adaptation in Mobile Automatic Speech Recognition Applications

Date: 23.01.2017 Language: English Number of pages: 8+64

Master's in Communications Engineering

Professorship: Speech Recognition Code: S3013

Supervisor: Prof. Mikko Kurimo

Advisors: M.Sc. Matti Varjokallio, M.Sc. Leo Hämälainen

Generally, the user group of a language is remarkably diverse in terms of speaker-specific characteristics such as dialect and speaking style. Hence, quality of spoken content varies notably from one individual to another. This diversity causes problems for Automatic Speech Recognition systems. An Automatic Speech Recognition system should be able to assess the hypothesised results. This can be done by evaluating a confidence measure on the recognition results and comparing the resulting measure to a specified threshold. This threshold value, referred to as confidence score, informs how reliable a particular recognition result is for the given speech.

A system should perform optimally irrespective of input speaker characteristics. However, most systems are inflexible and non-adaptive and thus, speaker adaptability can be improved. For achieving these purposes, a solid criterion is required to evaluate the quality of spoken content and the system should be made robust and adaptive towards new speakers as well.

This thesis implements a confidence score using posterior probabilities to examine the quality of the output, based on the speech data and corpora provided by Devoca Oy. Furthermore, speaker adaptation algorithms: Maximum Likelihood Linear Regression and Maximum a Posteriori are applied on a GMM-HMM system and their results are compared. Experiments show that Maximum a Posteriori adaptation brings 2% to 25% improvement in word error rates of semi-continuous model and is recommended for use in the commercial product. The results of other methods are also reported. In addition, word graph is suggested as the method for obtaining posterior probabilities. Since it guarantees no such improvement in the results, the confidence score is proposed as an optional feature for the system.

Keywords: Acoustic model, ASR, Confidence score, MAP, MLLR, Speaker adaptation

Preface

First of all, I owe my gratitude to Almighty ALLAH for granting me perseverance and determination to complete this thesis.

This thesis has been done at Department of Signal Processing and Acoustics (SPA), Aalto University School of Electrical Engineering. I thank my supervisor Prof. Mikko Kurimo for his continuous support throughout this thesis. I am grateful to him for providing me the opportunity to work in his group. My advisors Matti Varjokallio and Leo Hämälainen have been providing me with help and valuable input, and I am thankful to them. The assistance of my friend Salman Ishaq has also been precious for this work. Some parts of this thesis are the outcome of our collaboration. Special thanks goes to my life-long friend Bilal Ahmad and Tahir bhai for motivating me and helping me in various stages.

Finally, I dedicate this effort to my parents, Prof. M. Zafar Abbas and Tahira Parveen for their countless blessings and support. Everything in my life is due to their hard-work and faith they put in me. My brother Dr. M. Waqar Abbas is an inspiration for me and his encouragement has been the most significant factor in my work and life overall.

Otaniemi, 12.12.2016

Muhammad H. Abbas

Contents

Abbreviations

ASR	Automatic speech recognition
CDHMM	Context-dependent Hidden Markov model
CER	Confidence error rate
CIHMM	Context-independent Hidden Markov model
GMM	Gaussian mixture model
HMM	Hidden Markov model
LM	Language model
LVCSR	Large vocabulary continuous speech recognition
LVSR	Large vocabulary speech recognition
MAP	Maximum a Posteriori
ML	Maximum likelihood
MLE	Maximum likelihood estimation
MLLR	Maximum likelihood linear regression
OOV	Out of vocabulary
PDF	Probability density function
SD	Speaker-dependent
SI	Speaker-independent
WER	Word error rate

List of Figures

List of Tables

1 Introduction

1.1 Overview

Humans interact with each other through some modes and methods, and speech is arguably the most important of those. Human speech is delivered in a natural language. Currently, many natural languages exist and all of those have different structures. In recent years, significant advancements have been observed in the field of constructing systems that are able to recognise the speech. They are generally referred to as Automatic Speech Recognition (ASR) systems. The last decade, in particular, has seen some major improvements in development of such systems. Their use in everyday life has increased tremendously, which was previously in specific tasks of some ASR-oriented application only. The demand of ASR system applications in industries is ever increasing. Until a few years ago, it was a big question if primary user devices can provide the computational power needed to run this system. Recently, the vast use of internet and cloud storage facilities have made possible the use of speech recognition in common mobile devices. The application areas of ASR include automotive industry, health sector, military (intelligence), mobile device assistants and call centres.

Speech recognition performs well, when it is performed under laboratory conditions that are clean and the noise is very low [4]. This performance decreases substantially when the conditions are changed from ideal laboratory conditions to real life where a lot of background chatter and noises are present [5]. Regarding the accuracy, the system can be trained using small chunks of speech of various different speakers, which improve its the robustness and make it less dependent on speaker.

When a systems detects a speech signal, the quality of spoken words is examined. A reliable system has the capability to estimate how well a particular set of words is spoken to it. It evaluates the reliability of the spoken words by assigning them some scores, which are then used to prioritise some words over others. This measure is called confidence score and assists the system decide which spoken words are better than the others. Posterior probability, acoustic stability and hypothesis density [3] are few of the confidence scoring measures.

In the beginning, most of the speech recognition systems used to be speaker-dependent (SD) systems [6]. Those systems perform well with a certain (set of) speaker(s) only. At the same time, this property raises a considerable issue because it restricts the performance of the systems to those speakers only. In recent years, speaker-independent (SI) systems have been developed to overcome this flaw of SD systems. These are able to perform well with a variety of speakers. As [6] describes, word error rate (WER) of a SD system is generally two or three times lesser than that of a SI systems, hence SD systems perform better than SI systems in case of only the specific speakers. SI systems are more robust and adaptive towards a wider

range of speakers as compared to SD systems, hence these are a better choice when the system is used by a number of different people. To introduce this independence from the speaker-specific properties, SI systems are embedded with some additional features. The most important of those features is known as speaker adaptation and is an essential requirement of a SI system. Since most of the systems are used by a number of different users, it is a good idea to incorporate the adaptation algorithm in the system to make it more responsive in case a new speaker interacts with it. The improvement in adaptation quality is usually directly proportional to the degree of mismatch between the data used for training and adaptation.

1.2 Target

This thesis is divided in two tasks that aim towards improvement of an existing speech recognition system used by Devoca Oy. The first task is to understand confidence scoring and based on that understanding, build a measure from the acoustic model that calculates the respective confidence scores from given hypothesised words. This score indicates the reliability of the recognised words in output transcription. This thesis develops a confidence score based on posterior probabilities of words in a recognition hypothesis. Word graphs are used for calculating the posterior probabilities and those probabilities are compared to a specified threshold value. The output hypothesis contains only those words which have higher posterior probability values than the threshold.

The second task of this thesis is to develop the basic knowledge of speaker adaptation and its implementation. Two of the most popular methods are implemented, namely Maximum Likelihood Linear Regression (MLLR) and Maximum a Posteriori (MAP) to adapt the acoustic model. Since, these two are well established methods in the field of ASR, a general behaviour of the outcome is already known. This thesis is an effort to verify these methods on the system and data. These techniques and their respective results are discussed. Based on the results, this thesis is concluded with the method that fits the data well and produces lesser word error rate (WER).

1.3 Organization

This thesis can be divided into following chapters.

Chapter 1 analyses the research problem and briefly mentions the methods which are used to solve the problem.

Chapter 2 explains the fundamental concepts and presents the idea of speech recognition process. This chapter introduces ASR systems and their building blocks, and describes the concerning toolkit along with its usage.

Chapter 3 briefly discusses small vocabulary systems, their limitations and their practical aspects.

Chapter 4 and 5 give relevant theoretical knowledge and explain the details of confidence scoring and speaker adaptation algorithms respectively. These include different methods used to implement these algorithms.

Chapter 6 describes the building process and features of the acoustic model along with all the steps during training and evaluation phases. It also introduces the experimental setup which is used for the implementation of the methods mentioned in the respective chapters with their associated results and necessary underlying details.

Chapter 7 provides discussion on the results and some recommendations about further experimentation.

2 Speech Recognition

This chapter introduces the speech recognition and important building blocks of a speech recognising system. The discussion starts with the evolution of ASR systems over the past hundred years and significant milestones in the history of this field are mentioned. The next section focuses on explaining different phases of the system such as feature extraction, acoustic model and language model. Finally, a brief introduction of the toolkit used in this thesis is described.

Following discussion on the history of ASR system is largely based on [7].

2.1 Historical Evolution

Speech technology has been under the scientific radar since early 20th century. Since when a system model for speech synthesis and recognition was proposed, this field has encountered a lot of changes and improvements, and the progress in the last century has been nothing short of astonishing. The earliest recognisers were generally single digit recognisers. On the other hand, the systems currently in use are state-of-the-art and take into account all the varying statistical aspects of the signal and handle complex speech with good accuracy.

In 1930, the first major milestone was achieved when Homer Dudley at Bell Laboratories developed a speech synthesizer known as VODER. In 1952, Bell Laboratories developed a single digit recogniser for a single speaker [8]. Homer Dudley and Harvey Fletcher were the first ones to introduce the idea of processing the speech signal in frequency domain [9, 10]. The systems in modern day are based upon this approach. Dudley and Fletcher can be regarded as the pioneers of speech recognition, since they emphasised the significance of signal spectrum and the phonetic nature of a speech signal. Since phonemes carry a separate and independent mapping in frequency domain, it is relatively easy to process those in frequency domain, than in time domain.

Successful results in the statistical framework resulted in the development of several new systems, including CMU Sphinx [11] and DECIPHER [12]. The last decade of 20th century saw a rapid trend in development of software technology, which initiated speech research worldwide. Systems were being improved with the passage of time and made capable of handling complex speech signals and models. With the growing research needs, a standard system was required which would serve as a baseline software and is usable for future development of new algorithms and methods. In this regard, HMM Tool Kit (HTK) [13] developed at Cambridge University exists as the most widely used software in speech systems community around the world.

2.2 Framework of speech recognition

Recognising speech is a complicated task, since most recognition tasks involve a large vocabulary. When different words are spoken in a distinct and separate manner, it is fairly an easy task to recognise those. The actual condition appears to be otherwise in routine conversations. Words are spoken fluently and their exists no clear boundary that may help the system distinguish between different words. This makes an already tough recognition task even tougher. Speech is generally represented in terms of waveform which is not ideal for recognition task. The signal possesses a complex structure and conveys superfluous of information as well. The initial goal is to represent this signal in terms of features which preserve only crucial information which is essential for speech recognition. Systems can use the knowledge of the feature vector

$$\mathbf{O} = \mathbf{o}_1\mathbf{o}_2...\mathbf{o}_t \tag{1}$$

where $\mathbf{o}_t$ is the feature at any given time frame t. The aim is to obtain the word sequence

$$\boldsymbol{W} = \mathbf{w}_1\mathbf{w}_2...\mathbf{w}_m \tag{2}$$

where m is the total number of words. This problem is probabilistic i.e. to find the word sequence that is most probable.

$$\hat{\boldsymbol{W}} = \arg\max_{W} P(\boldsymbol{W}|\mathbf{O}) \tag{3}$$

If the feature likelihood, word probabilities and observation probabilities are known, Bayes' formula is used as follows:

$$P(\boldsymbol{W}|\mathbf{O}) = \frac{P(\mathbf{O}|\boldsymbol{W})P(\boldsymbol{W})}{P(\mathbf{O})} \tag{4}$$

Since $P(\mathbf{O})$ is associated with all of the word sequences so it can be ignored [14] in equation 4 for convenience in calculations which results in:

$$\hat{\boldsymbol{W}} = \arg\max_{W} P(\boldsymbol{W})P(\mathbf{O}|\boldsymbol{W}) \tag{5}$$

Acoustic models are compared with observations, and language models provide possibilities of different word sequences and their respective probabilities. Decoder processes all this information and results in the recognised text. Figure 1 describes the structure of a commonly used system, constructed upon aforementioned components.

2.2.1 Feature Extraction

Similar to all the signals that exist, speech waveform also contains some relevant and irrelevant information These pieces of information form properties of the signal.

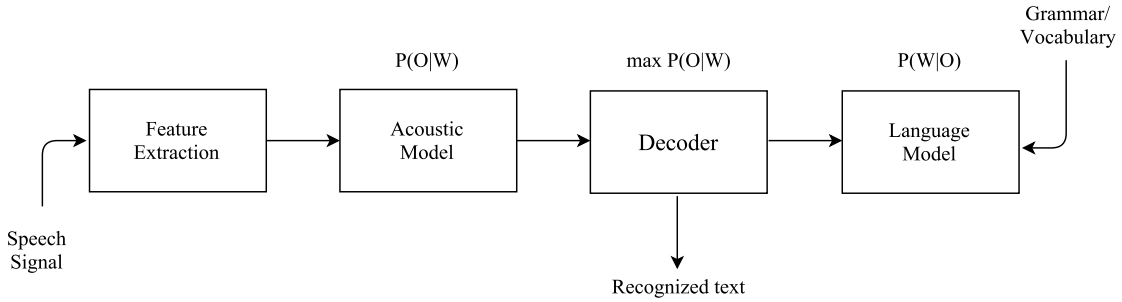

Figure 1: Typical structure of a modern ASR system

On a larger scale, sentences or words and on a smaller scale, phones or phonemes can be termed as the relevant information. In this thesis, those relevant pieces of information are termed as *features*. Inconsistent information such as fillers, cough and sneeze, and consistent information e.g. environmental noise and intrinsic variation of speech can be classified as the irrelevant information. Process of recognising speech initiates with the parametrization or feature extraction of the signal. These features are quite tricky to define and deal with. This thesis attempts to extract the features that represent the signal and contain the maximum relevant information about the words. Feature extraction produces usable and compact signal from speech. This provides the system with the optimum knowledge of the content and lessens the recognition complexity. Therefore it is vital to find the features which consist of least amount of dimensions and provide the most important information with minimum loss. These features should undergo no changes if the environmental or speaker conditions fluctuate.

As mentioned in section 2.1, the idea of processing speech in frequency domain has been in practical use for the some time [9, 10]. The main reason why time domain signal is converted to frequency domain is, that this signal contains high correlation among its different components. whereas frequency domain components correlate very less and can be independently processed. Each sound part generates a distinct spectral counterpart, which makes its easier to infer the original content from the respective spectral component. The spectrograms of all the vowels and the word *kaksi* (Finnish equivalent of number two) are depicted in the Figures 2 and 4. It can be observed that *a* and *ä* generate visibly different spectra in Figure 4. Similarly, *o* and *ö* differ substantially in their frequency ranges.

An important step, referred to as spectral shaping, is done as pre-processing and involves two fundamentals operations: analog to digital conversion and filtering. Former is a well known term and its explanation is out of the scope in this thesis. Latter can be described as a frequency pre-emphasis filter, which blocks relatively less important frequencies and allows only useful frequencies to pass. This is usually a first order FIR high-pass filter. The mathematical expression associated to this filter is given as follows:

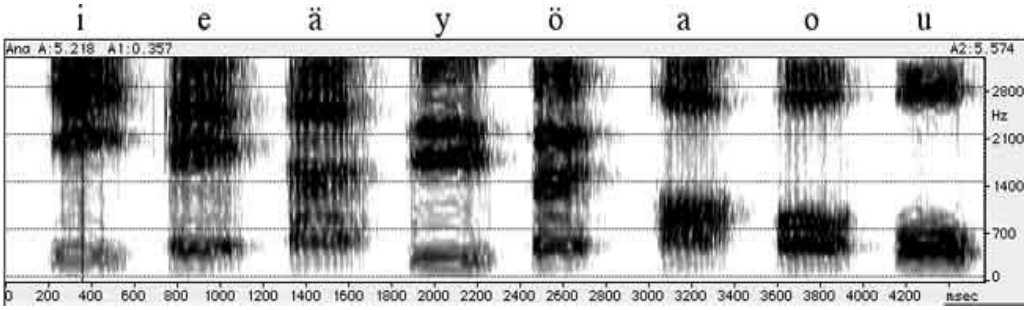

Figure 2: Vowels in Finnish [1]

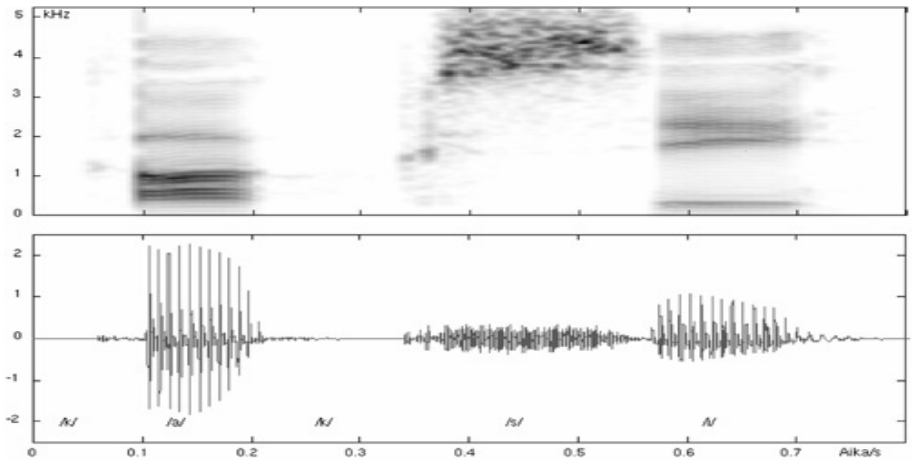

Figure 3: Time domain (below) and frequency domain (above) signals of the word *kaksi*. Different phones have distinguishable spectra [2]

$$H_{pre}(z) = 1 - a_{pre}z^{-1} \tag{6}$$

whereas the value of coefficient a_{pre} usually ranges from 0.9 to 1.0 according to [15]. It is important to have some information in high frequency region as well. This filter attempts to flatten out the spectrum and neutralises this natural offset.

Next step is framing and windowing of the signal. Speech signal is a slow time-varying process. Generally slow transitions are found in the signal and the signal can be considered almost stationary over a very brief time span. This short period often lasts in the range of 10 ms to 25 ms. Framing process divides this signal into those short time intervals (frames). These are essentially very small time frames which overlap with each other. Overlap is approximately 10 ms for these frames. Moreover, there exist some discontinuities at the frame edges, which are minimised by the application of Hamming window [15]. This serves as a necessary procedure before spectral analysis. The reason is, that presence of these abrupt transitions or discontinuities results in redundant high frequency outliers afterwards. This windowed frame is the typical input for feature extraction process, which then applies different spectral schemes such as Discrete Fourier Transform (DFT) or Fast Fourier

Transform (FFT) to obtain the spectrum. In a certain frame t, short time DFT can be calculated as follows:

$$S(k,t) = \sum_{n=0}^{N-1} w(n)s(n,t) \exp \frac{-j2\pi kn}{N} \tag{7}$$

where n denotes the discrete time, k represents the frequency and N is the total length of the DFT. Hamming window $w(n)$ is applied to the time domain speech signal $s(n,t)$. The resulting signal $S(k,t)$ is the spectrum of the original signal.

Naturally, humans have good distinguishing capability in low frequency regions, and doing so in high frequencies is somehow more tedious. This emphasises the need to introduce a perceptual frequency scale that conforms with the non-linear hearing ability of human beings. One of the most common perceptual scale translates linear scale frequency into Mel-frequencies as follows:

$$m = 2595 \log_{10}(1 + \frac{f}{700}) = 1127 \log_e(1 + \frac{f}{700}) \tag{8}$$

Spectral components are compressed in a logarithmic manner to compensate the non-linearities in the loudness of auditory system. This compression also helps in speech enhancement. Applying discrete cosine transform (DCT) to this logarithmically compressed spectrum results in Mel Frequency Cepstral Coefficients (MFCC). In order to reduce the dimensions, only those coefficients are used which consist of lower orders. MFCC is the most widely used type of the features, though some alternative choices exist as well [16, 17].

Moreover, logarithmic signal energy can directly be computed from time domain signal. This replaces c_0 as the first coefficient of MFCC. In the very final stage, a normalization scheme is applied to handle the issue of noise robustness which is associated to these features.

2.2.2 Acoustic Models

Phonemes constitute the words in speech. Each phoneme is expressed by features, and those features should be matched with phonemes. In practice, phonemes correspond with feature chains rather than certain individual features. Acoustic models are utilized to model those feature chains and required as an elementary structural component in a typical modern ASR system. They characterize the time structure of phonemes. GMM based HMMs are widely used for acoustic modelling [18] because these contain a certain degree of flexibility and accuracy. Both simple as well complex HMM systems can be trained.

2.2.2.1 Hidden Markov models

Section 2.1 discusses that the idea of practically applying a statistical framework using HMM was first put forward in 1980s, though those were already known in theory [19]. Before this era, speech was modelled by systems using simple template-oriented techniques. This modelling proved to be insufficient in order to account for all the variations involved. Instead, a different approach based on stochastic processes, called Hidden Markov Model is used. HMM is an effective statistical method in defining the individual samples of a discrete time series data [20].

Phonemes are modelled as HMM state chains. Normally a HMM consists of three left-to-right states. For different features, probabilities associated with states are defined as *observation probabilities*. States of HMM are statistical models and some probabilities are associated with transitions from one state to another. In a Markov chain, the output of a state at any given time is not random. Naturally, it can be extended to a framework where observation of a state is a probabilistic process. This extension can be termed as HMM, which is associated with a non-observable stochastic process. This process in turn is further affiliated with another stochastic process, which generates an observable feature sequence. Since the former process is invisible, the word *hidden* is an important part of the notation.

HMM is based upon modelling the underlying inconsistency and variations of speech and its spectrum in a statistically consistent and integrated modelling framework. Speech is essentially highly inconsistent because of differences in accent and pronunciation. Many environmental elements e.g. noise and reverberations contribute to its degradation. Hence, when people speak identical words, the corresponding signals are acoustically different, though the linguistic structural elements such as grammar may remain similar. HMM suits well to model these variabilities of speech due to its use of stochastic modelling [21].

In HMMs, emission PDFs assign probability to the observations. One of the most widely used functions for assigning these probabilities are Gaussian mixtures. Gaussian Mixture model (GMM) calculates the observation probability for state i as:

$$b_i(\mathbf{o}(t)) = \sum_{n=1}^{N} \omega_{in} \mathcal{N}(\mathbf{o}(t)|\boldsymbol{\mu}_{in}, \boldsymbol{\Sigma}_{in}) \tag{9}$$

whereas n^{th} component which is essentially a Gaussian in state i, is obtained by using normal distribution with ω_{in} as the mixture weight vector which should satisfy the condition

$$\sum_{n=1}^{N} \omega_{in} = 1 \tag{10}$$

These probabilities are referred to as *emission distributions*. GMM involves normal distribution, defined as:

$$\mathcal{N}(\mathbf{o}(t)|\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) = \frac{1}{\sqrt{2\pi}\boldsymbol{\Sigma}_n} \exp[-0.5(\mathbf{o}(t) - \boldsymbol{\mu}_n)^T \boldsymbol{\Sigma}_n^{-2}(\mathbf{o}(t) - \boldsymbol{\mu}_n)] \tag{11}$$

whereas $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ serve as the mean vector and covariance matrix respectively in equations 9 and 11. Gaussian distributions essentially consist of the same dimensions as that of feature vector. The amount of training data affects the number of feature vectors and consequently the number of Gaussian component mixtures. For a large training dataset, more mixture components can be robustly trained and modelling accuracy IS improved.

HMM critically supposes that signal can be characterized in terms of a random parametric process. Foundation of HMM lies in Markov chains, states and their inter-state transitions. System is supposed to appear in one of the states at any given instant of time t.

$$P(q_t = S_j|q_{t-1} = S_i, q_{t-2} = S_k, \ldots) = P(q_t = S_j|q_{t-1} = S_i) \tag{12}$$

$$a_{jk} = P(q_t = S_j|q_{t-1} = S_i), \quad 1 \leqslant i, \quad j \leqslant N \tag{13}$$

where a_{jk} denotes transition probabilities. These depend only on the observation probabilities. The main difference between a typical Markov model and HMM is the visibility of the state to observer. HMM tends to hide the states as suggested by the nomenclature, but in classic Markov model the states remain visible. This essentially means that the particular discrete state is unknown which the system lies in. Only the feature vector of that hidden state is accessible. The simplest, first order HMMs suppose two vital assumptions. First is the basic assumption as is used in the Markov chains and described in equation 12. The second assumption is the output independence assumption, as follows:

$$P(X_t|X_{t-1}, q_{t-1}) = P(X_t|q_{t-1}) \tag{14}$$

The assumption in equation 14 implies that emission probability at any given time t only depends upon the current state s_t. It can be stated that conditionally all the previous states contribute nothing towards the outcome of next state. Essentially this assumption restricts the memory, thus makes the model efficient.

So far, phonemes have been considered as the basic unit. Occasionally phonemes alone, are inadequate for the acoustic modelling because sometimes recognising a phoneme can present great amount of hindrance. Hence those are put in context and considered in the form of triphones.

Many fundamental limitations of HMM root from its assumptions [18]. A significant drawback is encountered when it is assumed that successive observations in the sequence $\mathbf{O} = \mathbf{o}_1\mathbf{o}_2...\mathbf{o}_t$ are independent of each other. This assumption is not entirely correct always, because there is dependence or relation to some extent that always exists between those observations.

$$P(\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_t) = \prod_{n=1}^{t} P(\mathbf{o}_n) \tag{15}$$

Markov assumption [22], that probability of the current state is affected by immediate previous state only and not by other previous states is another major issue. Some states affect the outcome of distant states and the dependencies usually stretch beyond one state. In essence, both these assumptions theoretically narrow down the use of HMM in speech recognition. Since HMM relies on a statistical framework, it works notably accurately despite aforementioned limitations. Many solutions have been presented in order to further reduce the impact of these limitations, but they have produced no outstanding improvement in practical speech recognition applications [20].

2.2.3 Language Models

Every language has a different set of rules. It is important to provide the system with the knowledge of a language while constructing a speech recogniser. Although acoustic model is trained for a particular language, some detailed information about the language is still needed. From the point of view of formal language model, grammar and parsing algorithm are the two notable ways to do so [20]. Initially collecting the necessary knowledge regarding the language in the form of grammar or dictionary is a good tarting point. It must be considered that a huge number of word combinations from the given grammar are possible. Some of those combinations are more important than others. Some commonly used combinations show up frequently, some appear occasionally while some of those never occur. Intuitively the system should be equipped with some knowledge as to which word sequences or combinations are more likely to appear. Assume if the recognised words in order of appearance are (`"not"` or `"knot"`), (`"is"`) and (`"tied"` or `"tide"`). LM should result in `"knot is tied"` as is the expected outcome. Other possibilities such as `"not is tide"`, `"not is tied"` and `"knot is tide"` should not be allowed as they make no sense and are unlikely. If this sequence slightly changes as (`"is"`), (`"not"` or `"knot"`) and (`"tied"` or `"tide"`), the possible outcomes are: `"is not tied"`, `"is not tide"` and `"is knot tied"`. Former of the combinations is the most probable one, next combination has a better probability of occurrence and latter is possible but highly improbable at the same time. This means that LM should assign high probabilities to former two.

Language model performs two important tasks. Primarily, it gives the information which restricts the output of recogniser to the permissible sentences only. In this way, those word or phoneme sequences are discarded that are outside the premise of language. For different languages, dictionaries may vary in terms of pronunciation. For instance, English lexicon is rather complicated because at times, many phones are pronounced different and often different phones sound the same, both in different contexts. As an example of the latter, the words `"distinguished"` and `"distinction"` produce a similar `"sh"` sound near their ending, though the corresponding phones are different. Secondly, it calculates the probabilities $P(\boldsymbol{W})$ associated with different word sequences. Generally N-gram models [23] are used to execute this task. A typical N-gram model statistically expresses the grammar. Suppose if N

words in a sequence i.e. $\boldsymbol{W} = \mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_N$ are available and the history of N-1 previous words is known, according to N-gram models the probability of a word at a time instant t can be obtained by:

$$P(\mathbf{w}_t) = P(\mathbf{w}_t|\mathbf{w}_{t-1}, \mathbf{w}_{t-2}, ..., \mathbf{w}_{t-N+1}) \tag{16}$$

and the probability of word sequence would be calculated as the product of individual word probabilities as follows:

$$P(\boldsymbol{W}) = \prod_{t=1}^{N} P(\mathbf{w}_t|\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{t-1}) \tag{17}$$

According to this formulation, the word sequence probability of three words is $P(\mathbf{w}_3) = P(\mathbf{w}_3|\mathbf{w}_1\mathbf{w}_2)P(\mathbf{w}_2|\mathbf{w}_1)P(\mathbf{w}_1)$. In this expression, $P(\mathbf{w}_2|\mathbf{w}_1)$ represents a bigram and $P(\mathbf{w}_3|\mathbf{w}_1\mathbf{w}_2)$ a trigram. These probabilities are estimated in the maximum likelihood framework.

For any LM, a grammar or dictionary with k words potentially implies k^2 bigrams and k^3 trigrams. This exponential factor is unnoticed when the size of vocabulary in question is small. However this factor leads to enormous growth in size of N-grams in LM when the size of dictionary ranges in a few thousand or even hundred words. Maximum likelihood estimates assign zero probability to the N-grams that appear nowhere in the training corpus [24]. Different smoothing methods are applied for shifting the probability mass slightly from seen N-grams to unseen ones [25], since those with zero probability may create problems after recognition process.

Language models models grow complex depending upon the dictionary size as is mentioned earlier. Assuming that the language model allocates $P(\boldsymbol{W})$ probability to a word sequence $\boldsymbol{W}$, the cross-entropy of the model in equation 16 may be calculates as

$$H(\boldsymbol{W}) = -\frac{1}{N_W} \log_2 P(\boldsymbol{W}) \tag{18}$$

whereas N_W represents the number of words in the sequence $\boldsymbol{W}$. Now perplexity can be defined as

$$PP(\boldsymbol{W}) = 2^{H(\boldsymbol{W})} \tag{19}$$

Essentially perplexity is the probability of test set normalized by the number of words. This inference becomes evident once equations 18 and 19 are combined. The higher the perplexity in equation 19, the more the number of statistically weighted word branches decoder has to process and more error prone the recognition results can be. Generally this means that lower perplexity corresponds to comparatively higher recognition performance than that of higher perplexity [20].

2.3 CMU Sphinx toolkit

CMU Sphinx is a software tool for speech recognition, developed at Carnegie Mellon University. This tool is an essence of two decades of research at this university. It is a collection of ASR engines developed at the university's speech research group. CMU Sphinx toolkit consist of several different components, designed for specific applications [26].

- Pocketsphinx - a low-resource speech recognising library developed in C
- Sphinxtrain - an open source acoustic model trainer.
- Sphinx4 - a flexible recogniser, developed in Java.
- Sphinxbase - a support library for Sphinxtrain and Pocketsphinx.

Pocketsphinx is one of the open source [27] speaker-independent LVCSR engines developed as a part of the toolkit. It is written in C language and uses standard unix autogen systems. Pocketsphinx is continuously being developed and incorporates features such as fixed-point arithmetic and effective algorithms for computing GMM. Packages, such as Sphinx4 are eventual result of evolution of older packages such as Sphinx3.

This toolkit relies on HMM technology as its basic framework. Currently, CMU Sphinx is capable of supporting many languages such as English, Dutch, French. German, Italian, Mandarin and Russian, and accents within a language such as English (UK) and English (US). In the form of Sphinx4, CMU Sphinx can handle Java libraries in addition to C libraries. This makes incorporation of ASR using Java applications an easier task. Pocketsphinx comes with quick and portable implementation, whereas Sphinx4 provides a modestly flexible and adjustable platform that suits the needs of the application. Modifiability and continuous development make the use of CMU Sphinx toolkit advantageous.

3 Small Vocabulary Systems

An important factor that attributes to the quick response of an ASR system is the higher computational power and faster processing capacity embedded in modern devices such as personal computers or smart phones. Another main reason why current ASR systems work efficiently is because the size of vocabulary is not very large. Recognising speech becomes a harder task when the number of words to be considered for recognition is very high. The mobile applications for which the development work is done in this thesis are used on smart mobile phones or tablet systems. These products `Talk'n'Label` and `Talk'n'Pick` are developed by a Finnish company Devoca Oy. These applications use the offline speech recognition engine of Pocketsphinx. These are used in labelling of electrical equipment and storage facilities respectively.

Based on the requirements of the application, the dictionary consists of 150 words, most of which are commands, names, numbers and some other special terms needed in such working environments. This thesis is restricted to the modification of recogniser only, even though the synthesis can also be performed in these applications. There are a couple of operating modes available for the Pocketsphinx engine. Usually, the full recognition mode is enabled and recogniser continuously listens to the input audio signals. As soon the speech is detected, it is recognised and then the recogniser stops when the speech is finished. The decoder decodes the content in speech and text-to-speech module plays the speech back for the user to verify the recognition. Meanwhile the transcription is displayed on the screen of the device in use and is

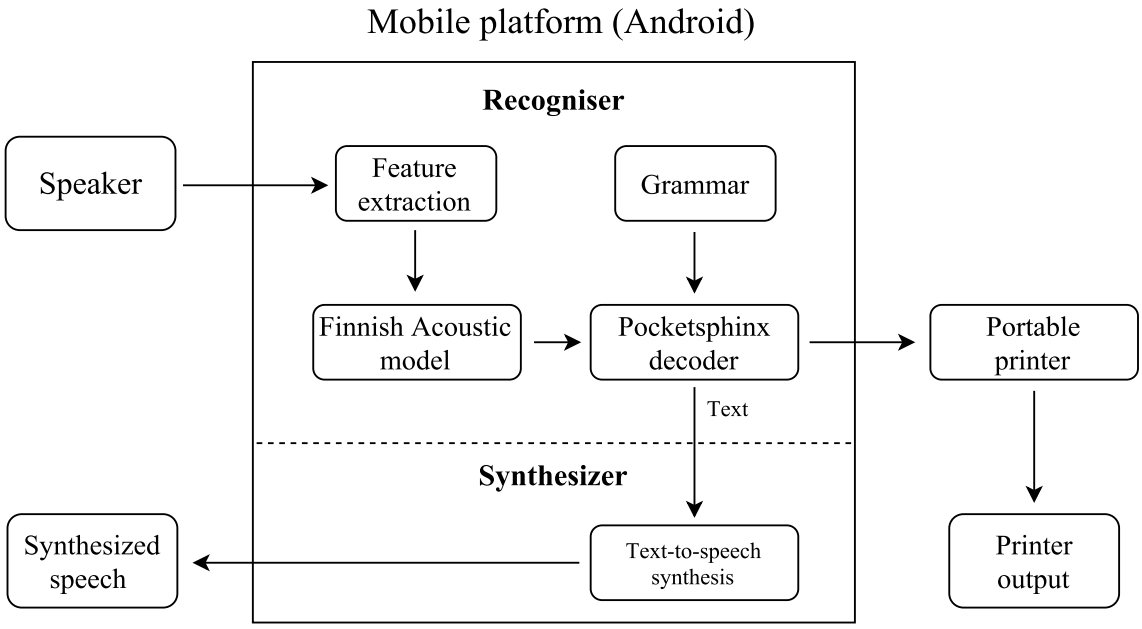

Figure 4: Architecture of Talk'n'Label

printed on the command of the user. At the end of this activity, the recogniser starts listening once again. The recognition works properly if speech of range of 10-15 seconds is detected. Beyond that limit, the recogniser may produce some errors in the output.

The application is currently designed to work on android platform. The efficiency and speed of recognition are of vital importance. Since the user requires the commands printed without losing much time, the results should be reliable and provided in the minimum possible time.

4 Confidence Scoring

With advancements in ASR technology, our interaction with machines through the means of speech has increased enormously. Almost all of the current mobile devices are equipped with one of speech recognition applications, which otherwise would have seemed impossible a few decades ago. The embedding of speech recognition in computers and mobile devices is leading towards a hands-free world. In order for these machines to be completely operated by speech commands, speech recognition should be reliable enough to produce the accurate results.

Current state-of-the-art systems perform on consistently good levels of accuracy [4]. These perform adequately when tested in laboratory conditions. However, these systems are far from accurately perfect and occasionally generate recognition outputs that contain errors when used in real-time conditions [5]. Assuming that the output contains some errors, it becomes more relevant to measure how much confidence can be put in the output being correct. Bourlard et al. [28] argue that accuracy of the systems can be improved if certain criteria are set and the output hypotheses are rejected that fail to fulfil those. Measuring the confidence scores can serve as one such criterion, against which the output accuracy can be evaluated. This score indicates the reliability of a recognition decision by an ASR system. Confidence scoring is the process of obtaining such mathematical number for the output transcription hypotheses using certain confidence measures. Equipping the recogniser with a measure of confidence is a precious extension. Such measure can be sufficient enough to provide a platform for many other applications such as detecting out-of-vocabulary (OOV) words [29] and keyword spotting [30]. Basically this sort of applications has sparked the initial motivation to build such methods which measure the confidence. Confidence measures play an important role where the system dialogue with the user is involved. The reliability induced by these measures leads towards a faster and reasonably rational interaction. Confidence measuring may also come into play when some OOV or unclear spoken words are found in speech. This measure discards those segments of speech by assigning them low degree scores. The error in recognition hypothesis is generally caused by either non-speech or other words uttered with poor acoustic characteristics. Confidence measures detect and reject this kind of words in output. As mentioned above, developing this measure can play a crucial part in increasing the ability of ASR systems to evaluate the recognition results [31].

Usefulness of systems can be assessed in terms of their capability to reliably evaluate the recognition results. Problem of measuring confidence can be divided into many different levels [32]. Estimating the confidence on word level is the most intuitive method. As the size of the unit of estimation is increased, the confidence of segments, sentences and eventually complete audio files can be measured. It needs no mention that going from word level to complete audio level means that faith is put in the entire data being correct. This seems slightly optimistic, as there may be several individual errors inside the speech though its confidence as a whole is high.

4.1 Methods

Jiang [5] argues that the current confidence scoring methods can be split in three categories. The first one can be termed as classification approach. As suggested by its name, the confidence scoring techniques in this category develop a classifier which takes the output of the concerning ASR system as its input. The main idea behind this approach is that some features are extracted from speech, generally referred to as *predictor features*. Considering those features the classifier determines whether a particular recognition hypothesis has a high likelihood of being correct or incorrect. This is an effective technique, since each of those represents certain system properties. There are various such features available that can be used in this process to estimate the confidence score [20, 33], which provides with a degree of flexibility.

Choice of these features varies from one situation to another, since every feature signifies distinct acoustic and system information. Many features have been proposed in this regard. Hypothesis density [34, 35] is based on lattice structures. Utilizing the acoustic properties of speech such as normalized acoustic score likelihood per phone [36] and acoustic stability [37] are among prominent methods. The features such as LM score and back-off behaviour [38] are effective ones as well. Next step is to evaluate these features by feeding them into a classifier which calculates a composite confidence score. One of the initial tasks of the classifier is to get rid of some statistical dependency left in the features and then produce a usable score. Keeping in mind all the steps so far, this method may prove very inefficient and resource-hungry as calculating the feature(s) and building a classifier on top of that may add extra computational complexity.

In the second category, confidence scoring as a hypothesis testing problem in statistical framework is considered. In this method, the ratio of the evidences for a word being correct and being incorrect evaluated. This approach belongs to the framework of Utterance Verification (UV). The problem of speaker verification motivated the use of UV as a measure of confidence [31]. Assume that for an acoustic observation X, the output of the recogniser is a word W symbolised by the HMM λ_W

Hypothesis 1 H_0: X is correctly recognised and comes from λ_W

Hypothesis 2 H_1: X is incorrectly recognised and comes from λ_Z

whereas λ_Z represents the HMM associated with all incorrect word hypotheses. Now by testing null hypothesis H_0 against alternate hypothesis H_1, the system determines whether the recognition results should be rejected or accepted. According to Neyman-Pearson lemma, this testing can be formulated as a likelihood ratio (LR) testing problem.

$$LR(X, \lambda_W, \lambda_Z) = \frac{P(X|\lambda_W)}{P(X|\lambda_Z)} \underset{H_1}{\overset{H_0}{\gtrless}} \varepsilon \tag{20}$$

where the parameter ε is the critical decision threshold. Lleida et al. [39] suggest that this LR score can be used as a confidence measure. A considerable shortcoming of this method is the estimation of alternate hypothesis which generally represents complex event with an unknown distribution of true data. One solution to tackle this challenge is to select the alternate hypothesis such that it has the same HMM λ_W as that used for null hypothesis [40, 41].

In the third category, the confidence score of a word is interpreted as its posterior probability in transcribed hypothesis. Further details and theory of this category are presented in the section below.

4.1.1 Posterior Probability

Modern ASR systems aim to produce recognition results that contain no errors in the output. Given that no system can achieve perfect accuracy of 100%, a more viable goal is to obtain the results which contain minimum amount of errors. In section 2.2.2.1, it is already discussed that current systems use statistical methods and principles to recognise the speech. In an ideally perfect system, the generated output hypothesis would always be assigned full confidence, leaving no need to measure the confidence separately. On the contrary, this is impractical in real systems and reliability needs to be checked. It can be safely said that since current systems use statistical techniques, speech recognisers put a significant amount of belief in the output they generate which indicates the actual accuracy of the system output. However, this implies that there is always some margin for error.

Confidence scoring can be applied on different levels. The most common ones are on document, sentence segment and word level. The document level and sentence level scoring generally suits well to the tasks such as machine translation [32], where quality of a document or its constituting utterances is to be judged. Either of the latter three can be implemented in speech recognition applications, depending upon the nature of the task.

The structure of a typical statistical ASR system is described in Figure 1. It can be observed that the speech recognition is generally modelled as a problem of pattern classification. Decoder uses MAP decision rule to check the posterior probabilities and selects the most likely word sequence as final hypothesis. In fact, this solution is found using Bayes' rule as presented by equation 5. This solution is originally derived from equation 3 which is reproduced here.

$$\{w_1^N\}_{opt} = \arg\max_{w \in \Theta} P(w_1^N | o_1^T) \tag{21}$$

$$\{w_1^N\}_{opt} = \arg\max_{w \in \Theta} P(o_1^T | w_1^N) P(w_1^N) \tag{22}$$

where Θ encompasses all the allowed possible sentences. The language model provides word probabilities in the form of $P(w_1^N)$ and the probability of obtaining a particular acoustic observation is given by $P(o_1^T)$. The probability $P(o_1^T|w_1^N)$ of the acoustic observation o_1^T with the word sequence w_1^N can be looked up from the acoustic model.

The output of a typical ASR system is generated by the decoder. In order to select the word sequences pertaining to the maximal posterior probability, MAP rule is utilized for decision. This probability may serve as a good measure for CE, however there is an important aspect to be considered. The calculation for this decision is devised using Bayes' theorem, as can be observed in equation 4 because it corresponds to the HMM structure and its generative layout [4]. One important assumption in obtaining the solution is that the system omits the term $P(o_1^T)$, for all word sequences have same observation probabilities [14]. Another reason for disregarding this terms is that it affects the decision process in no significant manner [4]. This is generally practised in ASR systems fundamentally owing to the fact that modelling the probability $P(o_1^T)$ necessitates its summation over all possible hypotheses which itself is a demanding task for LVCSR. The respective outcome eventually lacks the accuracy in modelling of this posterior distribution. This given presumption is the main reason that the posterior probability $P(w_1^N|o_1^T)$ can not be directly used as a solid framework for estimating confidence, as this solution provides inaccurate scores for assessing the recognition reliability.

Therefore the posterior probability estimation should be carefully re-considered and the normalization term $P(o_1^T)$ should be incorporated. Mathematically, this distribution can be calculated as follows:

$$P(o_1^T) = \sum_W P(o_1^T, w_1^N) = \sum_W P(w_1^N) P(o_1^T | w_1^N) \tag{23}$$

Theoretically, using the above equation 23 the precise value of posterior probability $P(w_1^N|o_1^T)$ can be calculated and can be re-written as follows:

$$P(w_1^N | o_1^T) = \frac{P(w_1^N) P(o_1^T | w_1^N)}{P(o_1^T)} = \frac{P(w_1^N) P(o_1^T | w_1^N)}{\sum_W P(w_1^N) P(o_1^T | w_1^N)} \tag{24}$$

This probability can be utilized as a viable confidence measure. $P(o_1^T)$ can be taken into account either by considering the assumptions which involve this probability or adopting the methods which approximate the explicit modelling of this distribution. In the form of equation 26, a strong platform is available for computing the level of confidence in the system output. One of the most widely used methods to calculate these posterior probabilities is by using word graphs [34, 42]. Another method which is commonly used is N-best lists [43].

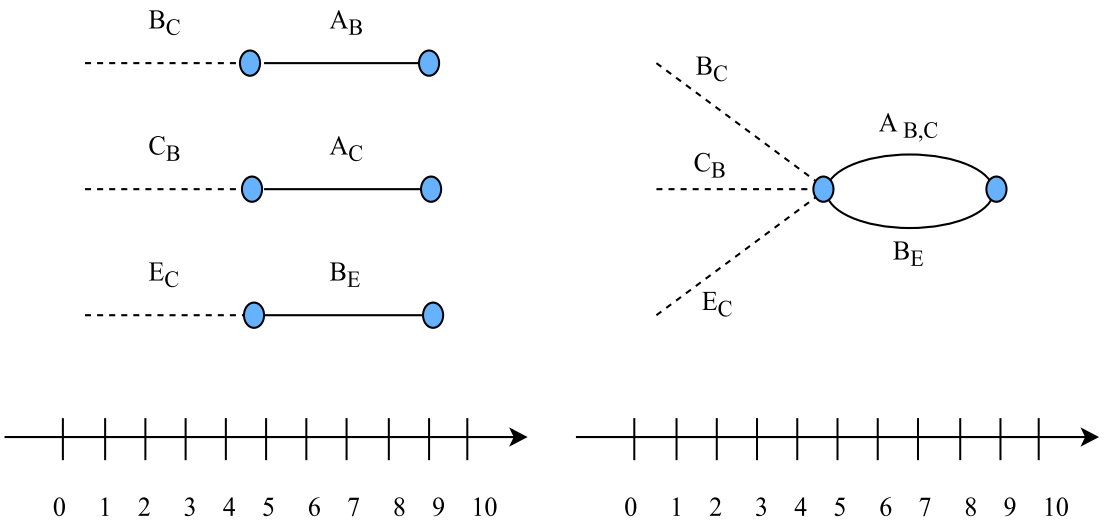

Figure 5: On the left side, six hypotheses with vocabulary [*A, B, C, D, E, F*] are presented. The right side shows the word graphs obtained after merging the hypotheses with similar parameters e.g. word indices and time frames. The horizontal axis represents time.

4.1.1.1 Word Graphs

So far there has been no mention of time as a factor in confidence scoring. One of the important methods to calculate the confidence score is the use of word graphs, in which time holds significance. For using word graphs as a method to calculate the posterior probabilities, distinct boundaries between words in a sequence are introduced. Assume that τ and t represent the starting and ending time frame of a word w respectively. The representation of the word is modified to $[w; \tau\ t]$. Thus the word sequence may be written as $[w;\tau,t]_1^N = [w_1;\tau_1,t_1],\dots,[w_N;\tau_N,t_N]$. This revises the equation 21 as follows:

$$\{[w;\tau,t]_1^N\}_{opt} = \underset{[w;\tau,t]_1^N \in \Theta}{\arg\max}\ P([w;\tau,t]_1^N|o_1^T) \tag{25}$$

$$\{[w;\tau,t]_1^N\}_{opt} = \underset{[w;\tau,t]_1^N \in \Theta}{\arg\max}\ \frac{P(o_1^T|[w;\tau,t]_1^N)P(w_1^N)}{P(o_1^T)} \tag{26}$$

Summing up the posterior probabilities of all the utterances which include the word hypothesis $[w;\tau,t]$ gives the posterior probability of this word, keeping in mind the above mentioned boundaries [3]. This may be expressed as follows:

$$P([w;\tau,t]|o_1^T) = \sum_{[w;\tau,t]_1^N:\ \exists\ m\in\{1,\dots,N\}:[w_m;\tau_m,t_m]=[w;\tau,t]} \frac{\prod_{n=1}^N [P(o_{\tau_n}^{t_n}|w_n)P(w_n|w_1^{n-1})]}{P(o_1^T)} \tag{27}$$

A word graph is a directed and weighted graph. Different parts of its structure are used to represent different quantities, e.g. time by nodes, word hypotheses by

edges and acoustic probabilities by weights. Several alternative paths exist from the first node to the last node in time which are termed as *alternative hypotheses*. Given the boundary times, the above mentioned representation is used for the the word sequence as $[w;\tau,t]_1^N = [w_1;\tau_1,t_1],\ldots,[w_N;\tau_N,t_N]$. In the process of recognition, most likely word hypothesis is saved for each time frame t. In the next step, merging multiple nodes with identical related time frames into one node provides with the word graph. On each edge, a list of predecessor words absorbs all the words that are stored in addition to the original word hypotheses. One of the multiple edges associated with the same word is preserved and made a part of the word graph. The list of predecessor words can be utilized to optimise the process of pruning and creating word boundaries. However, this list carries no affect or significance as far as the estimation of confidence score is concerned [3].

A word graph aims to represent infinite solution space in a finite manner, under the framework of output maximization governed by Bayes' rule. A word graph which is created in this manner is highly likely to consist of most probable sentence hypotheses.

Next step is to compute the hypotheses probabilities of word on those word graphs that have been constructed. In order to obtain those values, backward and forward probabilities are calculated and combined. For this purpose, forward-backward algorithm is utilized. For calculating the forward probabilities effectively, hypothesis for a word should be directly accessible either from starting frame τ or ending frame t. The forward and backward probabilities are obtained in a chronologically ascending and descending manner respectively. Forward probabilities take into the account the immediate predecessor (history) and backward probabilities the immediate successor (future) of the word [44].

$$P([w;\tau,t]|o_1^T) = \sum_{h_2^{n-1}} \sum_{f_1^{n-2}} \frac{\Phi(h_2^{n-1};[w;\tau,t])\ \Psi([w;\tau,t];f_1^{n-2})}{P(o_1^T)P(o_1^t|w)} \cdot \prod_{m=1}^{n-2} P(f_n|h_{m+1}^{n-1}w\ f_1^{m-1}) \tag{28}$$

whereas h_2^{n-1} and f_1^{n-2} represent history and future of the word hypothesis respectively. The last term in equation 28 denotes the language model probabilities and is ignored for all $n \leqslant 2$ [44]. The functions $\Phi(h_2^{n-1};[w;\tau,t])$ and $\Psi([w;\tau,t];f_1^{n-2})$ represent forward and backward probabilities respectively. For a specific edge, if forward and backward probabilities are multiplied without normalizing with $P(o_1^T)$ given that all the language and acoustic probabilities are equal to 1, the result correlates with the number of paths which pass through this particular edge. In the similar manner, $P(o_1^T)$ becomes proportionate with the number of paths through the complete word graph.

The word graphs obtained so far, as depicted by the figure above are optimised. If several nodes are associated with a single time instance, they are absorbed into one node and only one of the multiple parallel edges with similar corresponding word is preserved. In this process, the information provided by the language model context is

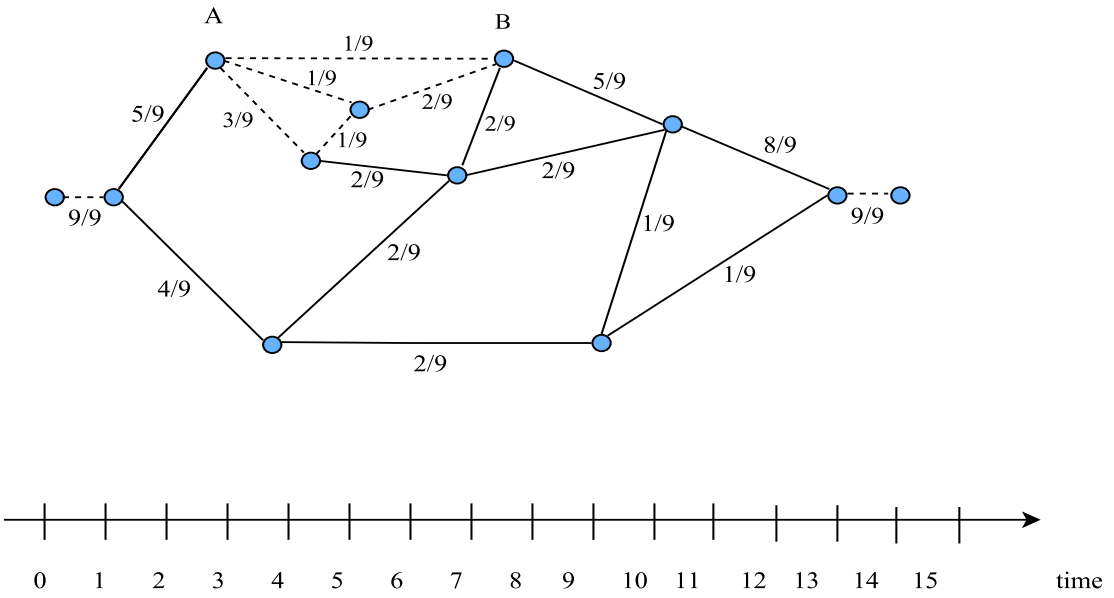

Figure 6: Example of a simple word graph. Word and silence hypotheses are illustrated by solid and dotted edges respectively. Probability values at any given time add up to one. [3]

lost and an aperiodic graph is obtained. The impetus behind ignoring language model history of word edges is mainly to obtain more hypotheses for sentences. This paves the way for implementing the algorithm in a candid fashion, since all the inter-edge transitions are handled uniformly. One of the drawbacks of this approach is that it yields slightly worse results than the one which considers the language model histories. One plausible reason for this phenomenon is as follows: when distinct language model contexts are involved in the consideration, transition from one edge to another is prohibited since both possess differing language model predecessors. When this constraint is removed, it becomes possible to jump from one edge to another and as a results multiple sequences of silence edges can formed. In figure above, it can be observed that there are four possible paths from node A to node B. Two of those are formed by different sequences of silence and all three of them are parallel. If those two are eliminated, only one path would remain. According to Wessel et al. [45], those silence edges which can be replaced by shorter silence sequences, are eliminated from the word graph. Neither the presence nor elimination of those two unnecessary path results in either some damage or loss of information. However it alters the posterior probabilities of all the edges, given the assumption that all edges' probabilities present at any given instance of time must add up to unity.

From the perspective of building an algorithm, removing those redundant edges is a good idea as it reduces the required time for computation and memory resources. The significance of this effect can be observed when the number of those inessential edges is large. Now use the measure found in equation 28 can be used as confidence measure.

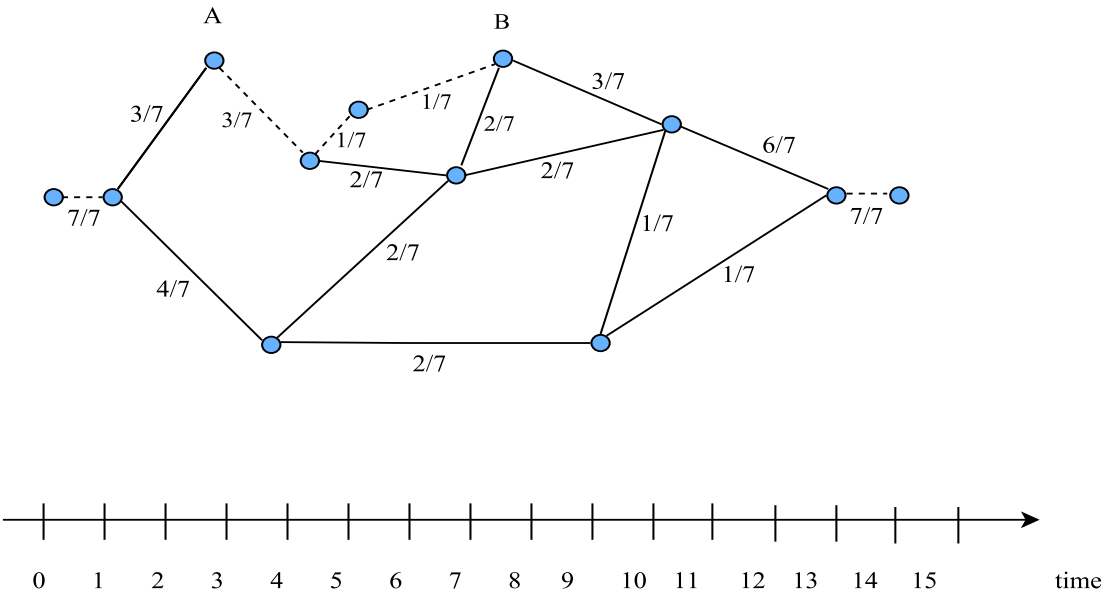

Figure 7: Modified word graph after omitting the redundancies [3]. Probability values are affected.

$$\mathcal{C}([w;\tau,t]) = P([w;\tau,t]|o_1^T) \tag{29}$$

4.1.1.2 N-best hypotheses

Rueber [46] and Weintraub [47] suggest that posterior probabilities of recognised sentences can be calculated using N-best lists. The same approach can be extended to the level of individual words. There are two main ways in which the N-best lists can be defined.

According to the first definition [3], N-best list can be represented as the set of N-best sentence hypotheses. Each of the sentences consists of the sequence of words only. In this case, there is no information available regarding either the starting or ending time of words. These hypotheses are established on the basis of word boundaries or word positions only. This helps circumvent the problem of segmentation faced in the use of word graphs. This may also help calculate the confidence score without incorporating the posterior probabilities of multiple hypotheses.

The second definition, a constrained one, describes the N-best lists as the set of N best sentence hypotheses $[w;\tau,t]_1^{M_1}, ..., [w;\tau,t]_1^{M_N}$. Generally it can be seen that using this definition, a certain number of the top N-best hypotheses may be identical. The only difference is that those have different starting and ending frames. It is simple to observe that if such an N-best list is considered, equation 27 can be used to calculate the posterior probabilities using summation over all the hypothesised sentences which contain the hypothesised word $[w;\tau,t]$. If word graphs and N-best

lists result in the identical sentence hypothesis, this eventually leads to the same posterior probabilities obtained using either of the methods.

In this scenario, it makes sense to use the word graphs rather than N-best lists because it calculates the probabilities much more dynamically using forward-backward algorithm and is more efficient than the latter. In case of this definition, using word graphs seems a better choice also because they require no such explicit summation of sentence hypotheses probabilities as that in case of the N-best lists.

5 Speaker Adaptation

The most important purpose of ASR systems is to convert speech into text. However speech is composed of many more underlying properties such as gender, context and environment information.A recogniser is not required to detect all that redundant information, rather it's task is to convert the spoken message into simple text.

In the past, speech recognition systems were trained using some specific speakers. Those systems use a huge amount of training data [18] and consequently match with the acoustic signal attributes of those speakers. This category of systems is accordingly termed as speaker-dependent (SD) systems. Those systems perform accurately when only those specific speakers are considered. The downside of such systems is that performance generally decreases significantly when a new speaker with unknown signal attributes is introduced. On the contrary, in real conditions, sufficient amount of data is not available to train the system. A more recent and advanced genre of systems is being used, known as speaker-independent (SI) systems. They are well-suited for the cases where training is difficult or entirely impossible. Generally these systems are outperformed by their speaker-dependent counterparts because SD systems are well-trained using the speaker-specific data [48]. Apart from this one superiority, the main drawback of SD systems is that the speaker who is to be recognised is required to speak several hours of transcribed training utterances. This is infeasible in circumstances where the task restricts the time and resources to be consumed. One might consider, that ideally a system starts with SI characteristic and adapts to a particular speaker with the passage of time without the need of huge training process [48]. This can be achieved by using small amount of data and tuning the SI system parameters to bring it closer to SD system. This process is referred to as *adaptation*

Speech signal properties vary from one person to another. Speaker mismatch [6] becomes a critical problem because a new speaker is not represented by any data in training process and no models can be created for and allocated to that speaker. This mismatch causes problems which lead to significant degradation in performance of ASR systems. A solution to tackle this problem is to incorporate an adaptation algorithm that is robust to the change of speakers. Thus the system is able to perform with better accuracy and provide results that approximate the SD system. Adaptation uses a small part of new speaker's speech data in order to tune the SI system because it models the speech from some speakers very poorly [49]. This process of including a small amount of speech data from target speaker in tuning of SI system is termed as speaker adaptation. This tuning makes the acoustic space more closely matched to the target speaker [50].

Adaptation methods are used where reference speech patterns are required to help the models adapt to scenarios that are unknown in the training of model. In speaker adaptation, the demand of speech data does not just simply disappear, but

it can be safely said that the adaptation process does not require as much data as training does [51, 52].

Adaptation can be divided into two main categories [49]; *speaker normalization* normalizes the input speaker's speech parameters to the target speaker which is to be modelled by the system, and *model adaptation* updates the model parameters to enhance the ability of acoustic model to characterise the new speaker. This thesis discusses model adaptation only. Model adaptation usually considers either language model [53] or acoustic model [49, 54, 55, 56] in order to perform the task at hand. In this thesis, discussion is restricted to adaptation using acoustic model.

Adaptation can be executed in two main modes. If the transcriptions are already known, the method is termed as supervised adaptation, e.g. when a speaker reads from a provided manual and utterances are manually transcribed. A positive aspect of this adaptation is that large amounts of data can be trained. Since the data is usually several hours of speech, the associated disadvantage with this method is that transcribing it correctly consumes a lot of time. Naturally, the amount of data should be within acceptable limits. If systems contains no knowledge of transcriptions, the type of adaptation is referred to as unsupervised adaptation. There is no prior labelling involved in this approach and transcription is generated by the initial recogniser output provided by training data. This methods is obviously outperformed by its supervised counterpart, but provides with the capacity to train huge data. This procedure is usually performed with two-pass decoding [55].

Lee et al. [6] formulate some interesting adaptation schemes such as sequential adaptation, and cluster adaptive training (CAT). They define speaker adaptation as the process of training SI model using SD data from the new speaker. Speaker adaptation can be performed using various methods. The most common methods so far have been linear transformation techniques such as Maximum Likelihood Linear Regression (MLLR) and Maximum a Posteriori (MAP). A slightly modified form of MLLR termed as Constrained Maximum Likelihood Linear Regression (CMLLR) is also used. This thesis implements the former two techniques. These methods are assessed in terms of their requirements of adaptation data and the extent of change these bring to the acoustic model.

5.1 Linear transformation methods

5.1.1 Maximum Likelihood Linear Regression

Model adaptation is one method to carry out the adaptation task [49]. Linear transformation methods perfectly illustrate this term and can be applied either in model-space or feature-space [57]. In current ASR systems, continuous density HMMs are used for acoustic modelling. Means vector and covariance matrix are

the standout parameters in any GMM-HMM based acoustic model. A set of linear regression transforms can be applied to the acoustic model in general and these two parameters in particular, which alters acoustic model in a manner that the adaptation data likelihood is maximised. Maximum likelihood linear regression (MLLR) transformation conforms with the criteria for constructing HMM.

The concept behind applying MLLR on Gaussian means is suggested by Leggetter and Woodland [49], and on covariances by Gales and Woodland [52]. According to the theory of MLLR, Gaussian means parameter of the model can be updated as

$$\hat{\boldsymbol{\mu}} = \mathbf{A}\boldsymbol{\mu} + \mathbf{b} \tag{30}$$

whereas $\mathbf{A}$ and $\mathbf{b}$ are the $N*N$ regression means transformation matrix and N-dimensional bias vector respectively. It is important to notice that feature vector $\mathbf{O}$ also consists of the same dimensions as that of bias vector. Often, updated means equation is also written in a compact form as

$$\hat{\boldsymbol{\mu}} = \mathbf{Y}\boldsymbol{\xi} \tag{31}$$

$\mathbf{A}$ and $\mathbf{b}$ are merged as extended transformation matrix which is written as $\mathbf{Y} = [\mathbf{b}^T \ \mathbf{A}^T]^T$. $\boldsymbol{\xi}$ represents the so-called extended means vector and can be defined as $\boldsymbol{\xi} = [1 \quad \boldsymbol{\mu}_1^T \quad \boldsymbol{\mu}_2^T \quad ... \quad \boldsymbol{\mu}_N^T]^T$. The purpose here is to find $\mathbf{Y}$ that maximises the adaptation data likelihood. [58] implements expectation-maximization algorithm for obtaining the solution to this problem. The transform can be applied to the covariance matrix as follows [55]:

$$\hat{\boldsymbol{\Sigma}} = \mathbf{H}\boldsymbol{\Sigma}\mathbf{H}^T \tag{32}$$

$$\hat{\boldsymbol{\Sigma}} = \mathbf{L}\mathbf{H}\mathbf{L}^T \tag{33}$$

$\mathbf{H}$ in above mentioned equations represents the transformation matrix to be obtained and $\mathbf{L}$ is inverse of the Choleski factor of covariance matrix $\boldsymbol{\Sigma}^{-1}$. Both equations 32 and 33 yield the same result if and only if the diagonal matrix $\mathbf{H}$ is considered. However, when this matrix is considered with non-zero off-diagonal values, a full covariance matrix $\hat{\boldsymbol{\Sigma}}$ is obtained and the transform is declared as *normalized-full* covariance transform [55]. However, it has been shown by Leggetter and Woodland [59], that applying the transform to the covariances guarantees no such remarkable improvement in terms of WER reduction as that of transforming means vector. This slightly helps increase the likelihood of adaptation data [52]

For full covariance transform case as suggested by [49], the transform and final covariance transformation matrices differ based upon the choice from either of two equations above. The method with full transforms has been shown to produce better results than that with diagonal transforms, though covariances are not modified [59].

As a general practice, if means and covariances are already adapted, the transformation parameters are optimized in two steps. The covariances are kept fixed and

the updated means parameters are obtained in the first step, and vice versa in the next one. This is governed by equations 30 and either of equations 32 or 33. The terms variances and covariances are interchangeably used throughout this thesis.

The transformation matrix can be of different types i.e. it can be full or diagonal. This variety can significantly alter the nature of results and their underlying complexity. Their is no such thing as the perfect number of transforms that can be applied to all possible scenarios, rather appropriate number of transforms are selected depending upon the amount of adaptation data. If this amount is limited, a global transform across all Gaussians can be used. With the increment in adaptation data, Gaussian components are often clustered together based on their closeness in the acoustic space. A particular transform $\mathbf{A}_r$ is assigned to those based on their regression class r [50]. Gales [60] studies these different grouping schemes in a detailed manner.

At minimum, adaptation requires an amount of at least 10 sentences (utterances) [61]. If not so, the data fails to accurately characterise the target speakers. As the amount of data tends to increase slightly, the performance of MLLR improves, because of its superior modelling capabilities.

A regression tree determines [62] the number of transforms for a particular adaptation data, as illustrated in Figure 8. Each node is the representation of a class. This class includes all the Gaussian components which a single transform is applied to. Only those nodes are chosen which are associated with sufficient amount of adaptation data. With the increase in amount of adaptation data, the number of HMM components which share a particular transform decreases and adaptation performance improves correspondingly.

5.2 MAP estimation

Adaptation of continuous density HMMs can generally be divided into two categories. Indirect adaptation methods, [63] such as Maximum Likelihood methods are most widely used in this regard. These are based on transformation of the reference model or acoustic space as a whole and accordingly can be considered as *global* methods of adaptation. Their approach is to simultaneously transform all the model parameters using a transformation function. A critical consideration is to ensure that transformation function is sturdily estimated based upon provided adaptation data. This consideration implies that if data availability is insufficient, a transformation is shared across relatively more number of Gaussians, than that in the case of sufficient amount of data [52]. This infers that the methods based on maximum likelihood framework manage to estimate a transform function given insufficient amount of adaptation data. They usually require large amounts of the data to accurately estimate and extract the required statistics from it, as illustrated by Figure 9.

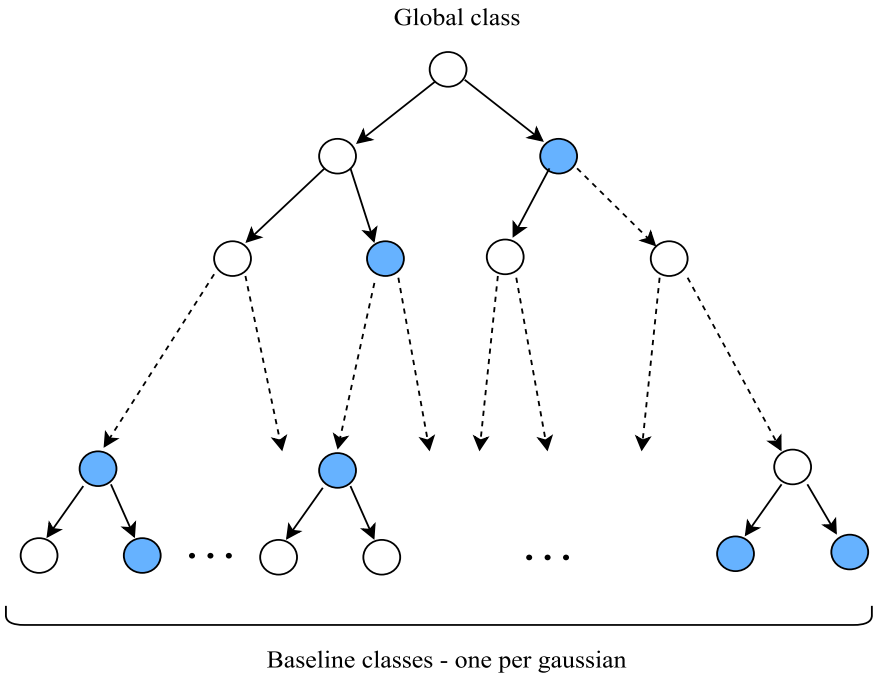

Figure 8: A regression class tree

However, the data available for adaptation in real life scenarios is normally very less. This essentially means that insufficient data is available in order to correctly estimate the transformation matrix [64]. A transformation matrix is typically relatively less accurate if it is determined using small amount of data. This eventually causes degradation in adaptation performance. This performance tends to attain improvement with the increment in adaptation data [65]. In case of the former, a different adaptation method known as Maximum a Posteriori estimation can be used to help alleviate this problem. It can be termed as a Bayesian estimate of feature vector [56].

Bayesian parameter learning [56, 66] is one of the direct adaptation methods. It uses a *local* approach which updates or re-estimates the individual model parameters separately rather than the global one which tends to jointly modify all the parameters using a single function. This process, depicted in Figure 10, ensures that only those parameters are updated which are observed in the adaptation data. Unseen parameters ones are left untransformed.

Bayesian learning can be implemented as MAP estimate, as follows:

$$\theta_{MAP} = \arg\max_{\theta} P(\theta|\mathbf{x}) \tag{34}$$

whereas $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$ represents a sample of T i.i.d. observations drawn from a mixture of K p-dimensional multivariate normal densities. The equation 34 above exhibits the likelihood of model parameters $P(\theta|\mathbf{x})$, which can be re-written using

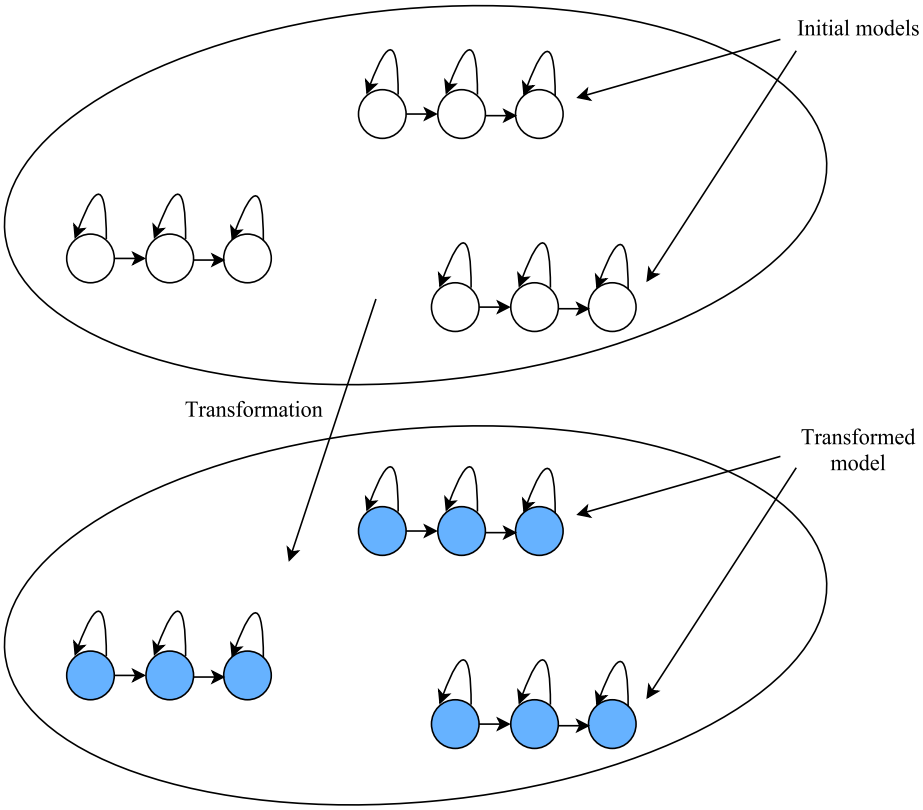

Figure 9: Global transformation using indirect adaptation of parameters

Bayes' rule as:

$$\theta_{MAP} = \arg\max_{\theta} P(\theta)P(\mathbf{x}|\theta) \tag{35}$$

In equation 35, $P(\mathbf{x}|\theta)$ indicates the adaptation data likelihood, combined with the prior distribution $P(\theta)$ which provides with the model parameters information. Prior information is the factor that distinguishes MAP from ML estimates in model parameters to be estimated. Gauvain and Lee [56, 67] suggest that the joint PDF of adaptation data likelihood can be specified as

$$P(\theta|\mathbf{x}) = \prod_{t=1}^{T} \sum_{k=1}^{K} \omega_k \, \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) \tag{36}$$

Mixture weight components ω_k and normal distribution are subjected to satisfy the condition in equations 10 and 11 respectively. Prior distribution in equation 35 provides with the statistics of model parameters. MAP adaptation updates the Gaussian means parameter of a mixture component as follows:

$$\hat{\boldsymbol{\mu}}_{MAP} = \frac{\tau \boldsymbol{\mu}_{prior} + \sum_t c_t \mathbf{x}_t}{\tau + \sum_t c_t} \tag{37}$$

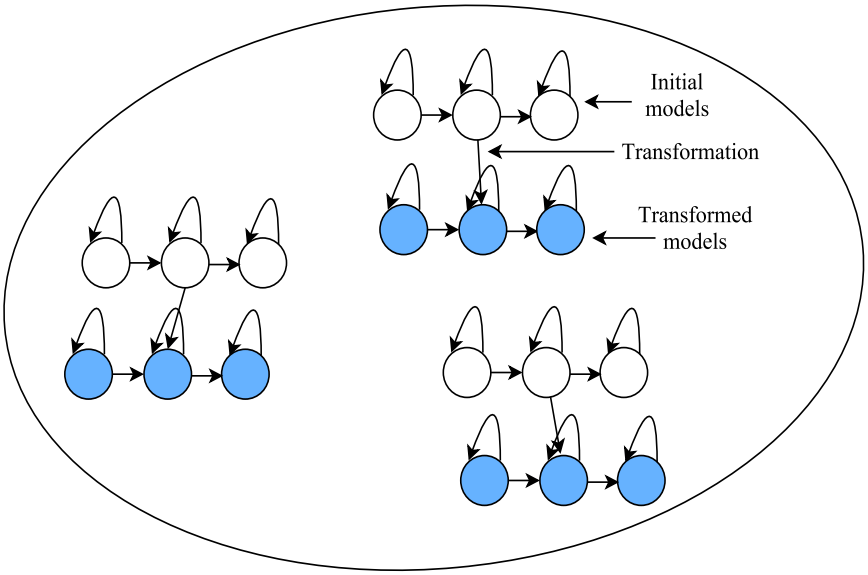

Figure 10: Local transformation using direct adaptation of parameters

If those parameters are unknown but fixed, they have to be estimated from the data, which implies that the prior is a *non-informative* one. Such a prior is as effective as no prior at all and leads to the convergence of MAP to ML estimates in equation 35 [56, 66]. Same convergence is observed when the amount of adaptation data increases [20], dominating any effect of the value of τ which maintains the balance in prior mean values and ML estimates. In theory, this means that SI systems can be converted to SD systems using sufficient amount of some speaker-specific adaptation data. This convergence can be a slow process since all the HMM models are not adapted.

MAP estimation is a supervised model adaptation method. It lies in the category of direct adaptation methods [63] since it can be applied to all the model parameters individually. MAP re-estimates the individual model parameters using the priori information of SD adaptation data. Due to this very reason, MAP is not an efficient method for adapting the acoustic model when the available adaptation data is scarce. The performance of this method increases in direct proportions with the increase in data. Section 6.3 discusses this in further detail.

6 Experiments

This thesis aims towards developing an acoustic model, adapting that acoustic model to match the characteristics of particular speakers using a small amount of adaptation data and applying confidence scoring to obtain the output transcription with high confidence. The methods and algorithms related to individual tasks are separately described.

6.1 Evaluation Criteria

The output of decoder is simple text, as is mentioned in section 2.2. This text is referred to as *output transcription*. In order to evaluate the system's performance, this transcription is compared to the reference input transcription and errors in output file are spotted. Word error rate (WER) is the edit distance [68] between the reference and automatically generated transcriptions, and is the most commonly used performance criterion for ASR systems [69]. The basic unit here is a word itself, which is classified either as correct or incorrect. Incorrect words are further divided into deletions (D), insertions (I) and substitutions (S). The WER is calculated as follows:

$$WER(\%) = \frac{D + I + S}{N} \cdot 100\% \tag{38}$$

whereas the total number of erroneous words is normalised by the total number of words in reference transcription N. Smaller WER means that systems performs well. There is one problem associated to this measure. If the number of incorrect words exceeds than that of total words, then the resulting WER is likely to be more than 100%, which makes on sense. This phenomenon occurs rarely, and generally the word error rate is likely to range within acceptable limits. It is a fairly straightforward task to assess the performance of a speech recognition system using above mentioned WER. However, choosing one evaluation metric may not suit every case that is encountered during an ASR task.

The language of the speech also affects the choice of this criterion as well. Take an example of two completely different languages. One may utilize WER as far as English language is concerned, for no significantly long words are used in daily conversations. If the language is Finnish, the situation seems to change. Finnish vocabulary appears to contain many compound words, which consist of several morphemes. Each of those morphemes correspond to meaningful individual English words. As an example, the word `"ravintola+päällikkö"` translates to `"restaurant manager"`. Even this long word can be further extended. In this scenario, a more suitable measure is letter error rate (LER), which considers letter as the basic unit. One may be tempted to think that perhaps LER may be the suitable measure for Finnish, but unfortunately the case seems to be otherwise. The reason is, that even though systems might

generate output transcriptions with tremendously low LER, the resulting words would be of no use since varying one letter in some words entirely alters their meaning. As an illustrating example, assume that `"tapaan sinut (I will meet you)"` is spoken to the recogniser and is transcribed as `"tapan sinut (I will kill you)"`. In this scenario, a minor LER of less than 10% indicates better performance than that of WER which is 50%. Visibly, these two words differ in terms of the contrasting meanings. LER cannot be selected just because it produces an acceptable error, for it corresponds to acceptance of an entirely incorrect output. Considering WER, a half correct output is obtained, hence 50% error. Thus, no measure can be considered absolute, even for a particular language. Several elements such as language, context and nature of the task must be examined before deciding a suitable criterion.

Since the concerning task is of small vocabulary and that of not very large or complex words, WER is used for reporting the performance throughout this thesis.

6.2 Building Acoustic Model

Acoustic models are statistical models which represent the acoustic properties of speech signal. There is no need to develop new models in Pockestsphinx if the system is to recognise a language which is already supported by this software. This case also holds true if the vocabulary of the system in question is large. For the scope of this thesis, this situation is bound to change for two reasons. First is that a model is needed specifically for Finnish language. The other important motivation behind this training is that the concerning vocabulary is small and carefully designed to match with the working environment of the mobile applications `Talk'n'Label` and `Talk 'n'Pick`.

An acoustic model in pocketsphinx is a set of files which represent individual aspects of the model.

- *feat.params* - contains parameter specifications regarding feature extraction process. In addition to training, this file is also used for configuring feature extraction during adaptation process. Model type, number of filter banks and CMN values are few of those parameters.
- *mdef* - defines mapping scheme between triphones and GMMs.
- *means* - Gaussian means file.
- *variances* - Gaussian variances file.
- *mixture_weights* - Gaussian mixtures file.

- *transition_ matrices* - comprises of matrices encapsulating information about transitions among HMMs in the model.
- *noisedict* - incorporates the filler words encountered during training, including cough and breathing silence.
- *sendump* - a compressed and quantized version of *mixture_ weights* file. It serves as the alternative for actual *mixture_ weights* file as well.

An acoustic model is trained so as to enable the system to recognise the input speech by itself. The extent of speech recognition ability largely depends upon the amount and nature of the data available for training the model. Training data consists of two sets. One set includes the speech recordings, and the other one comprises of precise and correct transcriptions of the respective speech recording. In general, several sorts of variabilities are naturally found in speech signal. In addition to those, another layer of complexities is added by the condition in which speech is recorded. This is an important aspect to address, for these mobile applications are designed for real working environments which may offer many unexpected noises. In order to address this concern, the training data must be as similar or close to the actual recognition conditions as possible.

An acoustic model is likely to perform adequately well when the training data resembles more or less the evaluation or test data, even though these are entirely mutually exclusive sets. The performance decreases slightly if the speech signal contains high amount of noise or unwanted inputs along with the actual speech content. On the other hand, presence of these unnecessary components in the data equips the system with better handling capability if it is to recognise the speech with similar challenging characteristics. As discussed earlier in the section 5, there are two types of system. Speaker-dependent (SD) systems remained as widely used model until recent times. Speaker-independent (SI) models are a recent advancement. These are a relatively robust form of SD systems and are trained on multiple different speakers. These serve well for several speakers contrary to merely the specific speakers in case of SD models. The datasets used to train the model along with other sets are described in section 6.2.2. The noise and unnecessary contents in the speech signal are generally filtered during feature extraction stage, which is discussed in 6.3.1. Even if there is some amount of those unnecessary signals left in the recordings, SI training is capable of handling it.

6.2.1 Model Types

Generally three types of models are used in acoustic modelling, namely continuous [70], semi-continuous [71] and phonetically-tied mixture (PTM) model [72]. The main difference lies in the model structure itself. As a standard practice, acoustic models use Gaussian mixtures for scoring frames. The element which distinguishes

these three is the manner in which those mixtures are constructed. This depends upon the total number of Gaussians. For continuous model, the total number of mixture components in the model is excessively large; approximately $1.5 * 10^5$. In case of semi-continuous model, 700 Gaussians are used. PTM model operates in the middle and uses almost 5000 Gaussians.

The number of mixture components or Gaussian associated with each model defines the behaviour of the respective model. Continuous models are accurate and precise, but the computation time to calculate score of each speech frame using such a high amount of mixtures is far from normal, hence these models are slow. The case is different for semi-continuous models, for these process a frame swiftly but the output is generally less accurate as compared to that of continuous model. In this case, PTM model serves as a suitable trade-off between high performance level of continuous model and quick execution time of semi-continuous model. It provides approximately same level of accuracy as that of a continuous model with similar processing speed as that of a semi-continuous model.

6.2.2 Dataset Division

The data and corpora used for these experiments are provided by Devoca Oy. This data contains a variety of speakers depending upon gender and accent. The data is collected in different real time environments such as warehouses and some storage facilities of such manner. Those conditions include all sorts of noises and distortions which contribute towards the usability of this data and consequently the acoustic model. The dataset is classified into three categories i.e. training, evaluation and testing dataset. Speech recordings of 153 speakers is collected and the transcriptions are written in Finnish language which have been carefully designed for the applications. The recording of each speaker lasts approximately 10 minutes and each recording contains 60 utterances on average. In general, an utterance refers to a sentence which last in the range of 10 to 15 seconds. These recordings are `mono` audio files, sampled at 16,000 Hz and recorded in `.wav` format.

Table 1: Training dataset

Gender	Native	Non-native
male	77	8
female	50	8

The training data comprises of 143 speakers and spans approximately 24 hours of recordings overall. The recordings of all these speakers, regardless of gender or speaking accent, contain some degree of noise and other characteristic inputs from surroundings. It is observed that the speech data of females contains relatively less noise than that of males because it is recorded in relatively silent offices rather than actual working spots. Certain background interferences and noises appear in this

data as well. The testing dataset is not as distinctly defined in the same manner as training data. This dataset contains 5 utterances from each of the speakers and in total 828 utterances which are not used during training process. Overall this data consists of 10,100 words which approximate to 2 hours of speech.

There is another dataset which is labelled as evaluation dataset. This set consists of the speech recordings of all 10 speakers which are left unused in the training process. This data consists of 577 utterances and 6289 words, which translate into 90 minutes of speech data. The purpose of this set is to evaluate the performance of the model on speakers which are completely unknown and are not involved during the training phase.

Table 2: Evaluation dataset

gender	native	non-native
male	4	1
female	3	2

An important aspect to notice is that all these sets are essentially mutually exclusive and have no common elements whatsoever. This ensures that the model is actually assessed on fair grounds and receives no assistance or prior knowledge from any data in training which may possibly be common to some portion of evaluation or testing sets. In the training and testing sets, there is one common element though i.e. speakers are the same. Apart form that, the written transcriptions and corresponding actual speech is completely different for all sets and no overlap or common element exists.

6.2.3 Alignment of corpora

As mentioned earlier, speech corpora consist of speech files and their respective transcriptions. The transcriptions, originally prepared by Devoca Oy., are provided in the form of a fairly long text and speech in large audio files. Different datasets are associated with different portions of that long text and audio files. It happens rarely that two different people speak from exactly the same portion of the huge transcription file. A transcription file is usually a fairly simple text file with some standard markers in each utterance, as given below.

```
<s> eemeli kolme kertomerkki vaihda kuusitoista </s> (utterance_1)
<s> määrä ät neljä yhdeksän nolla neljä kolme </s> (utterance_2)
```

These markers are generally used by many a software to initialise and terminate a single utterance, and separate one utterance from another. The naming convention for utterances is also pivotal and must be specified carefully. The reason is that Pocketsphinx uses a control file which enlists all the utterances in a transcription file. The process of training or adaptation as is explained later in section 6.3.1, is

suspended in case the utterance names in transcription file and the corresponding control file are found to contain some mismatch. The control file is created in the format such as follows:

```
set_1/utterance_1
set_1/utterance_2
```

whereas `set_1` and `set_2` here represent the speaker name which is the folder path of these utterances as well. Once it is ensured that all the files are synchronised with each other, the transcriptions should be harmonised with their corresponding speech recordings. In relatively less complicated tasks, this may be simpler to perform manually, but this case is slightly different. In this situation, where the data is as large as 24 hours, one is tempted to use some automatic means of aligning speech recordings with their transcriptions. Bash scripting and an efficient tool in the form of Sphinx4-aligner provided in Pocketsphinx, are used. Bash script automatically cuts large speech audio files into smaller and easily processable ones, and Sphinx4 aligns the transcriptions with the smaller speech files obtained so far. This tool is used to remove the unwanted parts of the speech as well such as coughing, throat-clearing or sneezing associated to actual speech, which may cause misalignment. The output after alignment contains words along with the stamps of time frame. This helps distinguish those words from others within actual speech and from unknown words or undesired chunks mentioned above.

The output of the aligner provides some important pieces of information which can be observed from an example below.

```
eemeli [00600:01610]
kolme [02710:03520]
kertomerkki [04220:05450]
+  vahvista [05990:06970]
-  vaihda
kuusitoista [07800:08730]
```

The values inside brackets represent *starting* and *ending* time frames (in milliseconds) respectively for each corresponding word. According to this alignment output, the word `vahvista` is present in speech with its occurrence span provided in terms of starting and ending time frames. This means that it is not included in the transcription file and should be added there. Similarly, the word `vaihda` is not present in the speech, hence it should be deleted from the `utterance_1` in the transcription above. Besides splitting the long speech files into smaller ones, bash scripting can also be used to align the transcription in the same manner as Sphinx4 above, however Sphinx4 is preferred for the reason that it is a standard tool provided in this software package.

6.2.4 Training

The process of training an acoustic model consists of estimating four important parameters (A, B, Π, Q) for the each of the HMMs present in the model. A represents the transition probabilities as defined in the equation 12. B is a set of emission probability distribution for all of the HMM states. Π consists of the set of initial probabilities π_q providing with the probabilities of all the states $q \in Q$ in the beginning of the process, whereas Q symbolises the total number of states including i, j and k from equation 12 and remains invariant with the passage of time. The number of states is set by hand before the start of this procedure.

The initial probabilities are either mixtures of continuous density functions or discrete distributions. The former is usually recommended [73] and trains the models which are most widely used for acoustic modelling. It is suggested that the covariance matrix should be diagonal in nature [20, 52, 55, 73] which makes the computation of Gaussians simpler. The reason behind using this type of matrices is mentioned in section 6.3.2 in more detail.

The most widely employed method for calculating the parameters $\Lambda = (A, B)$ is maximum likelihood estimation (MLE). Provided that the training data $(X = \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_m)$ is available, maximizing the likelihood function $L(\Lambda|X)$ results in the updated parameter set as follows;

$$\Lambda' = \arg\max_{\Lambda} L(\Lambda|X) = \arg\max_{\Lambda} \sum_{Y \in \Gamma} \prod_{t=1}^{m} b_{y(t)}(\mathbf{x}_t) a_{y(t),y(t+1)} \tag{39}$$

whereas Γ symbolises the set of all paths permissible in HMM and $Y = y(1), ..., y(m)$ represents the sequence of HMM states. In the start of the process, parameter set Λ is manually set with the training transcription set X at hand. Baum-Welch algorithm is used to compute the values of Λ' in such a way as to correspond to equation 39. Using MLE, Baum-Welch algorithm obtains the values of transition probabilities as follows

$$\hat{a}_{i,j} = \frac{\sum_r \frac{1}{L_r} \sum_{t=1}^{M_r - 1} \alpha_i^r(t) a_{i,j} b_j(\mathbf{x}_{t+1}^r) \beta_j^r(t+1)}{\sum_r \frac{1}{L_r} \sum_{t=1}^{T_r - 1} \alpha_i^r(t) a_{i,j} \beta_j^r(t)} \tag{40}$$

whereas a and b are used during forward-backward procedure during Baum-Welch algorithm. Now emission probability function is determined for the state b_q which means that parameters $(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \omega)$ are estimated for corresponding GMM. For MLE, using the Baum-Welch algorithm results in this set as follows:

$$\omega_{q,i} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_{q,i}^r(t)}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_q^r(t)} \tag{41}$$

$$\boldsymbol{\mu}'_{q,i} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_{q,i}^{r}(t)\mathbf{x}_t^r}{\sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma_{q,i}^{r}(t)} \tag{42}$$

$$\boldsymbol{\Sigma}'_{q,i} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma_{q,i}^{r}(t)(\mathbf{x}_t^r - \boldsymbol{\mu}'_{q,i})(\mathbf{x}_t^r - \boldsymbol{\mu}'_{q,i})^T}{\sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma_{q,i}^{r}(t)} \tag{43}$$

where $\gamma_q^r(t)$ represents the probability of occupancy of state q at time t defined as: $\gamma_q^r(t) = \frac{1}{L_r}\alpha_q(t)\beta_q(t)$. R is the number of observations sequences and a time frame is given by t such that $t = 0, ..., R$. The parameters i, j, q all represents states in HMM such that $(i, j, q \in Q)$. These estimations should be carried out repeatedly until the saturation point where the increase in the function $L(\Lambda|X)$ becomes insignificant and this function is maximised.

Training is implemented using Sphinxtrain which is designed for training acoustic models. the training procedure consists of numerous modules. This process can be initialised as soon as the data is available in a specified layout [26]. Data refers to two major components which are to be collected i.e. speech recordings and their respective transcriptions. Several other files are just as much necessary as these two such as a filler dictionary (Sphinxtrain considers it as noise dictionary which is merely a different name) providing the list of filler words e.g. cough and a long silence, a phone-set file which contains the list of all basic phonetic units which are used for the model, a phonetic dictionary file which maps the allowed words to the individual phones and a language model which provides the information about probabilities of different word sequences occurring together. In addition to these, a control file as mentioned earlier, which enlists all the files for training is also a necessary one. Since Finnish is the relevant language which is written and pronounced in the exact same manner, the mapping from words to phones is kept simple. As far as language model is concerned, 3-gram model is used for all purposes throughout this thesis.

The training process starts with the `Module 000 Feature Computation` which computes the feature files from given audio files containing the training speech recordings and `Module 00 Verification` which verifies that there is no contradiction of any kind amongst dictionary, phone-set and transcription files, and no repetition of phones or words is present. `Module 05 Vector Quantization` provides with the common framework for features which are dependent on acoustic characteristics of the training data. `Module 10` and `Module 11` perform the forced aligned training of CIHMM in particular and forced alignment in general respectively, using Baum-Welch algorithm in former case. `Module 20` consists of general context-independent training of HMMs. In `Module 30`, model is trained for triphones using untied CDHMM states. The next two modules are related to decision trees. `Module 40` builds trees for all the states associated with basic phones. `Module 45` prunes the trees computed in the previous module, reads the leaves of those pruned trees and creates a file containing model definition parameters. In addition, a triphone model is built using the same model definition file. `Module 50` uses the model parameters provided in the file generated in the preceding module and implements the triphone model training as

well, however unlike `Module 30`, CDHMM states are tied ones. The reason behind the process is that the training is undertaken in multiple stages. First, CI and then CD models are processed. Even each model has its own training stage, e.g. CD model are trained in two different modules i.e. `Module 30` and `Module 50` which are designed considering the nature of HMM states. The splitting of the training procedure reduces or in some cases, minimises the software initialization problems. The modules which are not mentioned here are skipped in an attempt to keep the size of the model to minimum and avoid computing excessive and unnecessary features. This trained acoustic model is used in the mobile application and a variety of tasks can be applied to this model.

6.2.5 Results

As described in section 6.2.1, acoustic models are of three types. Continuous model, the most accurate one, requires relatively huge amount of resources as compared to the other two. This is not trained because the model is going to be used in a mobile device. This discussion is limited to addressing PTM and semi-continuous models. The models are tested on evaluation and testing datasets and the performance criterion used here is WER. In case of evaluation data, the results are weighted average of 10 speakers and for testing dataset the results are weighted average for all the speakers form their respective sets. Number of senones is kept 3000 for both the trained models in all the experiments in this thesis.

Table 3: WER for different acoustic models

dataset	PTM	semi-continuous
evaluation	4.34%	4.87%
testing	4.30%	4.90%

It is observable that PTM model outperforms the semi-continuous model for both sets and this result corresponds with the theory associated with these models. Since the evaluation dataset is evenly obtained from male and females speakers alike, one may be tempted to investigate the error performance of both genders separately. These results are reported in tables 4 and 5 respectively.

Table 4: WER for males

evaluation dataset	PTM	semi-continuous
5 speakers	4.55%	5.27%

Table 5: WER for females

evaluation dataset	PTM	semi-continuous
5 speakers	4.14%	4.51%

It is also interesting to know that error rates of females speakers are relatively lower than those of male speakers. This phenomenon is related to the fact, that the evaluation data for females speakers contains no such surrounding noises and sudden disturbances as present in the data of male speakers. In general, male speakers have recorded their speech in such places which are in the vicinity of some machinery or equipment and females usually have done so in comparatively quiet office environments. This causes poor results for a couple of males and the overall performance is low for males speakers.

Generally, PTM model gives better results than those of semi-continuous model and provides approximately 10% relative improvement over its counterpart. In actual usage, PTM model appears to perform slowly, for the number of Gaussians is significantly higher than those of semi-continuous model which is a considerable issue for a mobile device. Hence, for this reason, semi-continuous model is preferred for usage in the actual application.

6.2.6 Comparison

The acoustic models which are trained so far, attempt to improve the mobile application currently in use. It seems that the errors these models produce are already significantly low. However it makes sense to check the performance of the models in retrospect and put forward a comparison between new and old model. In this way, the relative performance can be evaluated.

The model which has previously been in use, is semi-continuous with 4000 senones. This model is applied to the same evaluation and testing datasets, and results are presented in the Table 6 below. The poor performance of previous model is due to the fact, that it is trained on clean speech data whereas the model trained in this thesis is based on the data which contains all possible elements a working environment may offer. Thus this model has better capability to tackle several sorts of noise present in the data.

Table 6: WER for previous model

datasets	semi-continuous model (old)
evaluation	10.94%
testing	13.00%

6.3 Speaker Adaptation

In this section, the aspects of the experiments pertaining to speaker adaptation are discussed. Both methods which are used during adaptation experiments and their results are reported. Two methods; MLLR and MAP are discussed as far as the speaker adaptation is concerned. The amount of adaptation and test data, and the

resulting WERs on the test set are mentioned as well.

6.3.1 Framework

The first and foremost step to adapt an existing acoustic model is to collect necessary amount of adaptation data. The term adaptation data refers to the same sort of audio files as that of training data. These recordings, along with other parameters, are used to create the output MFCC feature file in `.mfc` format. The `feat.params` file from the acoustic model provides the parameters for feature extraction using the `sphinx_fe` tool provided by sphinxbase.

The value used in Pocektsphinx for pre-emphasis coefficient in equation 6 is 0.97. One of the parameters enlisted by `feat.params` defines which transform is to be used as the second transformation for computing MFCCs. Generally, DCT is used for experimentation.

When DCT has been applied, the obtained features are still vulnerable to the noise or convolutional distortions present in the received signal by the recogniser. Generally normalization techniques attempt to reduce this vulnerability by statistically matching the characteristics of the conditions used for training and testing. Cepstral mean normalization (CMN) is one of the most prominent methods [20]. In this technique, the mean of each cepstral vector is calculated and subtracted from the current vector. The continuous computation of cepstral mean is preferred. This ensures that the algorithm is recursive [74] and is applicable for real time systems, for cepstral vector is highly unlikely to remain stationary for an utterance of any given length. In essence, a more appropriate term is cepstral mean subtraction (CMS).

Among other parameters provided by this file is the switch for using cepstral mean normalization (CMN) and vector containing the initial estimate of its values. In the latest version of pocketsphinx software, this vector consists of 13 values. Taking the noise levels into account, these values generally vary with the input and are evaluated every time a new input utterance is received by the recogniser, as explained earlier.

The next step is to obtain the necessary count files required `bw` program. The transcriptions for the adaptation data are also required in Baum-Welch forward-backward algorithm. In addition to all these input files, the MFCC file from previous step is a necessary requirement. The output of this process are three files which contains statistical properties of the data. At this stage, the basic framework for adaptation is set and the actual algorithms for both adaptation methods can be implemented, which are described below with their results.

6.3.2 Maximum Likelihood Linear Regression

As discussed earlier, MLLR is used in the situations where the available amount of adaptation data is limited. So far, the required features from the adaptation data have been extracted successfully. Now, the MLLR transformation matrix can be obtained by using the output provided in the previous step by Baum-Welch algorithm. These feature files and the acoustic model parameters namely *means* and *variances* are used by the program `mllr_solve` to perform the MLLR adaptation. This process consists of calculating the means transformation matrix **A** and bias vector **b**. In other words, calculating the extended transformation matrix **Y** should be sufficient to obtain the required matrix and vector, as expressed by equation 31. Algorithm 1 provides with the basic idea behind computing MLLR transformations.

Algorithm 1 MLLR algorithm

Require: means $\boldsymbol{\mu}$, covariances $\boldsymbol{\Sigma}$, BW statistics
Ensure: MLLR matrix
1: **if** $\boldsymbol{\Sigma}_{ij} \neq 0 \; \forall \; i \neq j$ **then terminate**
2: **else**
3: **if** $\boldsymbol{\Sigma}_{ij} = 0 \; \forall \; i \neq j$ **then**
4: **for** each i **do**
5: **for** each t **do**
6: $C = \sum_t \gamma_t \boldsymbol{\Sigma}^{-1}$
7: **for** each r **do**
8: $G = \sum_r c_r \boldsymbol{\xi}_r \boldsymbol{\xi}'_r$ ▷ $D = \boldsymbol{\xi}\boldsymbol{\xi}'$ is symmetric
9: $Z = \sum_t \gamma_t \boldsymbol{\Sigma}^{-1} x_t \boldsymbol{\xi}'^{(t)}$
10: $g_i \; \mathbf{y}_i = \mathbf{z}_i$
11: **end all**

The final step of this process is the same as doing $\mathbf{y} = G^{-1} \; \mathbf{z}$. In order to execute this algorithm, the number of MLLR classes should be equal to one and this is achieved by setting the value of the switch `-cb2mllrfn` to `.1cls.`, which itself vaguely depicts that one class is used for codebook-to-mllr mapping function. The reason for using one global class is that all the mixture components get associated with a single regression matrix, which is the global transformation matrix.

As discussed in section 5.1.1, that MLLR adaptation algorithm produces no transformation matrix for the covariance matrix, because it is little effective as compared to that of means adaptation, hence the covariances are not adapted. In reality, the covariance matrices actually play an important role in the adaptation process, since their nature alters the course and ultimately the performance of adaptation process. As can be seen in step 5 of the algorithm that the resulting outer product D of extended means vectors is symmetric. The reason is because the covariance matrix is diagonal in nature. If covariance matrix were to be full matrix, the estimation formulae in tied matrix case would have a resource intensive closed form solution

[52]. For this reason, diagonal covariance matrices should be considered in order to perform this algorithm. This is also illustrated by step 1 of the algorithm, in which, for simplification reasons, this process is terminated if the non-diagonal entries in covariance matrix contain any value other than 0.

The scaling factor γ in step 4 is obtained from the adaptation data itself. Assuming that the adaptation data is available and labelled, the Baum-Welch algorithm, using the forward-backward approach, provides with the mixture component occupation probability γ which scales the inverse of covariance matrix in MLLR adaptation process in step 4 [59]. The resulting matrix `mllr_matrix` $\mathbf{Y}$ is the one which contains the transformation matrix $\mathbf{A}$ and bias vector $\mathbf{b}$. This matrix can be used according to the equation 31 to transform the `means` parameter of the model.

MLLR produces promising results by reducing WER significantly, given the amount of adaptation data is very limited e.g. a few utterances. This is largely due to the fact, that MLLR is a global transformation method and adapts the acoustic model as a whole using a single matrix, which in this case is `mllr_matrix`.

Generally, applying one iteration of MLLR seems to improve WER of the baseline acoustic model, but some room for improvement can be found. In this case, a recommended strategy is to repeat the process again and obtain the matrix which is the result of two iterations. In this way, the second repetition attempts to correct any errors left after the first iteration. This process considers approximately similar usage of above mentioned algorithms. However, the minute changes are done in `bw` algorithm in which the `mllr_matrix` is provided as well and the Baum-Welch statistics are updated based on a better understanding of the data. The other inputs of this algorithm remain the same. The updated `bw` output files are then used by the program `mllr_solve` and the transformation matrix after two iterations is obtained.

6.3.3 Maximum a Posteriori

MAP adaptation is a feasible option in situations where relatively large amount of adaptation data is available, as mentioned in the section 5.2. This data can be used as a prior for the adaptation process, since a SI acoustic model is already available. The amount of data used in training is always greater than that used in adaptation. In other words, MAP adapts an already present trained acoustic model using relatively small amount of adaptation data, whereas ML training computes the initial model from training data which is generally several hours of speech recordings, as discussed in section 6.2. In this MAP implementation, all four parameters of the acoustic model are adapted, i.e. *means*, *variances*, *mixture_ weights* and *transition_ matrices*. The program `map_adapt` take existing parameters as input and provide the updated parameters as the output.

All four of these model parameters are in binary format, but it is important to

notice that means is treated as a vector and variance as matrix. The model definition file `mdef.txt` and tied-states-to-codebook-function mode `.semi.` is provided because we are using semi-.continuous model. In case of continuous and PTM models, this input is different. The latter of the four arguments create the model which is an adapted version of original model using MAP estimation. The algorithm that is used for MAP Implementation is as follows:

Algorithm 2 MAP algorithm

Require: acoustic model parameters $\mathbf{\Gamma}(\boldsymbol{\mu},\ \boldsymbol{\Sigma},\ \omega,\ a)$, BW statistics $(\gamma,\ \boldsymbol{\xi})$
Ensure: adapted acoustic model parameters $\mathbf{\Gamma}'(\boldsymbol{\mu}',\ \boldsymbol{\Sigma}',\ \omega',\ a')$

1: **for** each t **do**
2: **for** each k **do**
3: $c = \gamma_t \dfrac{\omega \mathcal{N}(\mathbf{x}_t|\mu, r)}{\sum_k \omega \mathcal{N}(\mathbf{x}_t|\mu, r)}$
4: **for** each j **do**
5: $a' = \dfrac{(\eta - 1) + \sum_t \boldsymbol{\xi}_t}{\sum_j (\eta_j - 1) + \sum_j \sum_t \boldsymbol{\xi}_{jt}}$
6: $\omega' = \dfrac{(\nu_k - 1) + \sum_t c_t}{\sum_k (\nu_k - 1) + \sum_k \sum_t c_{kt}}$ ▷ $\nu = \omega \sum_k \tau_k + 1$
7: $\boldsymbol{\mu}' = \dfrac{\tau \boldsymbol{\mu} + \sum_t c_t \mathbf{x}_t}{\tau + \sum_t c_t}$
8: $\boldsymbol{\Sigma}' = \dfrac{\beta + \sum_t c_t (\mathbf{x}_t - \boldsymbol{\mu}')(\mathbf{x}_t - \boldsymbol{\mu}')^T + \tau(\boldsymbol{\mu} - \boldsymbol{\mu}')(\boldsymbol{\mu} - \boldsymbol{\mu}')^T}{(\alpha - p) + \sum_t c_t}$ ▷ $\beta = \tau \boldsymbol{\Sigma}$, $\alpha = \tau + 1$
9: **end all**

γ is the statistic of mixture weights and $\boldsymbol{\xi}$ in step 4 is the statistic regarding transition matrices and both are obtained from `bw` algorithm. η is the parameter set for transition probability. ν in step 6 represents the Dirichlet density parameter set for mixture gains. This parameter set is estimated in such a manner that speaker independent mixture weight is the mode of posterior distribution. τ is the decisive factor which balances the prior means and ML means estimate. Similar phenomenon can be observed for the covariance estimation formula. It is imperative that the prior densities should be chosen such that it belongs to conjugate family, for EM algorithm can be applied for MAP estimation in that case.

This algorithm is executed in such a manner that the above mentioned re-estimation equations generate the parameters which maximize the posterior density. Essentially this means that algorithm tries to find the local maximum for this density which is as close to the global maximum as possible or equal to it, given the minimum number of iterations for EM algorithm. In this regard, choosing initial estimate properly is of crucial significance. Adapting the variances guarantees no promising results. These are re-estimated regardless, because higher degree of variance in the SI models leads to a more robust adaptation. The value of τ is overestimated

for semi-continuous HMMs which contain a large number of mixtures. For this reason, it is preferable to use a fixed value of the factor τ for semi-continuous models. In this implementation, the value of τ is fixed to 10. The process of adapting `mixture_weights` and `transition_matrices` makes negligible difference, yet these parameters are adapted because sufficient amount of the adaptation data is available. The `mixture_weights` file is then compressed and used in the actual acoustic model. In this implementation, only one iteration of MAP adaptation is considered.

6.3.4 MAP and MLLR

There is another mode which combines both the methods so as to obtain slightly better results since both methods should complement each other given the sufficient availability of adaptation data. This method is not vitally different from either of these methods. The basic idea is to perform MAP adaptation first and then repeat the process of MLLR adaptation in the same manner as explained in the section 6.3.2 above. The basic difference between simple MLLR and MAP followed by MLLR is that MLLR creates the transformation matrix from the baseline acoustic model, whereas in the latter of the two methods, MLLR adapts a model whose parameters are already updated by MAP estimation.

6.3.5 Results

The terminology used in the Figure 11 is described in the following. When one adaptation utterance is mentioned, it refers to a sentence consisting of two minutes of speech. Similarly two and three adaptation utterances represent approximately four and six minutes of speech respectively.

The recogniser processes transcription in a relatively smooth manner if the number of sentences is reduced though on the expense of increase in the length of each sentence. This may not be a problem of paramount scale, for modern recognisers are far more superior than their predecessors from a decade ago in terms of processing speed with accuracy. The number of adaptation utterances is kept to 3.

The results illustrated here are weighted average of 10 speakers from evaluation dataset. This dataset contains equal number of female and male speakers. The amount of adaptation data for each speaker is approximately the same. As the next step, this model is examined on the test set and the resulting WER is reported.

First column in the Table 7 represents the number of utterances used to adapt the model and remaining columns illustrate different adaptation methods which are implemented in this thesis. All entries in the table show % WER. The entry *none* signifies that no adaptation has been performed and the model is baseline. MLLR1 and MLLR2 in this table indicate that MLLR has been performed for one

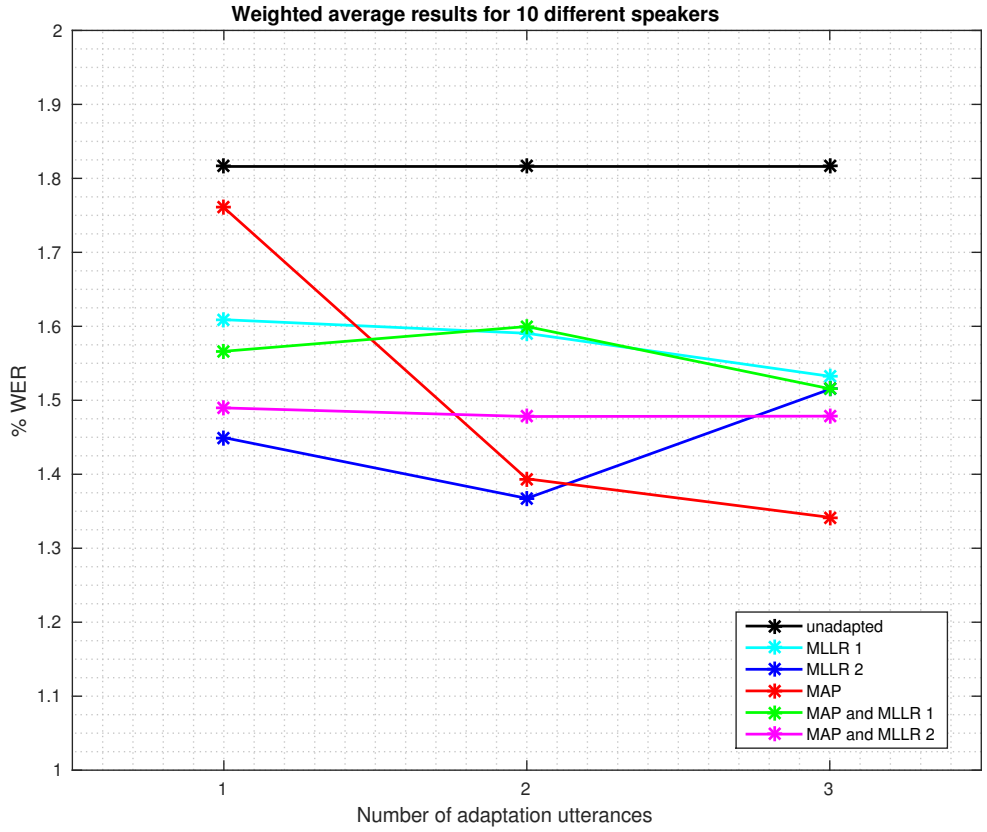

Figure 11: Relation between size of adaptation data and WER for different methods of speaker adaptation

Table 7: WER for speaker adaptation

	none	MLLR1	MLLR2	MAP	MAP+MLLR1	MAP+MLLR2
one	1.81	1.61	1.44	1.76	1.56	1.48
two	1.81	1.59	1.36	1.39	1.59	1.47
three	1.81	1.53	1.51	1.34	1.51	1.47

and two times respectively. The average WER is 1.81% for baseline model. In the beginning, it can be observed that methods which involve MLLR in any capacity, perform better than MAP. The reason behind this is that context independent (CI) phonetic classes are associated with the MLLR method. The presence of these classes ensures that when the amount of data is limited, MLLR outperforms all other methods. This conforms with the theoretical understanding that MLLR can adapt a model with less amount of adaptation data because it is a global adaptation method. MAP adaptation improves the performance slightly to 1.76% as compared to that of baseline model from 1.81% error, yet performs worst of all methods in the beginning. As the amount of data is increased, the adaptation methods decrease the

WER smoothly as graphs see steady fall from left to right.

For two utterances, MAP makes up for the errors left in the transcription so far and recovers to a decreased WER of 1.39% which is second best in all of the methods and 23% efficient relative to the model with no adaptation. When the third utterance is used, MAP outperforms all other methods by a significant margin and removes 25% more errors as compared to those of baseline model. This can also be related to the fact, that once MAP obtains sufficient amount of data, system attempts to adapt most, if not all, of the parameters located in the model. This means that more erroneous words are corrected and corresponding WER is lowest among all. MLLR can not achieve such degree of freedom as MAP does, because of its global approach for adaptation. On the contrary, MLLR, which produces a reasonable WER for the first utterance, seems to saturate quickly and does not improve the results in a promising manner even after three utterances have been provided for adaptation. However, there is an exception in the form of second iteration of MLLR. This method produces the least WER of all for one utterances i.e. 1.36% and progresses further in the right direction when two utterances are introduced. This trend suddenly changes when the data of third utterance is included for adapting the model and WER takes a spike in the upward direction. The reason for this phenomenon can be attributed to the fact, that the third utterances contains some words which are highly confusing for the recogniser and only MAP seems to remove or correct those erroneous words.

If first and second iteration of MLLR are compared in context of all three utterances, it can be observed that second iteration of MLLR improves the results of the first iteration. The margin of this gain is modest, yet a reasonable indication of the fact, that repeating MLLR for a number of times corrects those few errors left after previous iteration. Although the WER of second iteration of MLLR shifts surprisingly upwards for third utterance, it still reduces the error as compared to that of first iteration of MLLR.

The test set, on which these results are computed, consists of approximately 400 words which means that it is almost 5 minutes in length. The errors reported above are significantly low, e.g. the baseline acoustic model produces almost 8 incorrect words in the output transcription. These incorrect words are further improved as these adaptation methods attempt to reduce the number of errors remaining in the transcription whatsoever. One aspect of these results is, that the WERs are already low such that the margin for improvement is small. On the other hand it also translates into the fact, that recognising even one extra word incorrectly may cause substantial degradation in performance relative to these visibly low error rates.

When the experiments are separated for male and female speakers, the results for adaptation are as follows. It appears that at any given instant, adaptation works better for female speakers more than it does for male speakers. It can be observed from Table 9 that results are consistent for each method and slightly vary for any given adaptation utterance. It means that for a particular method, increasing or

Table 8: WER for males in speaker adaptation

	none	MLLR1	MLLR2	MAP	MAP+MLLR1	MAP+MLLR2
one	2.20	1.96	1.73	2.28	1.84	1.67
two	2.20	1.84	1.49	1.54	1.84	1.50
three	2.20	1.79	1.82	1.42	1.73	1.50

Table 9: WER for females in speaker adaptation

	none	MLLR1	MLLR2	MAP	MAP+MLLR1	MAP+MLLR2
one	1.50	1.32	1.22	1.34	1.34	1.33
two	1.50	1.38	1.26	1.28	1.39	1.45
three	1.50	1.32	1.26	1.28	1.34	1.45

decreasing the amount of adaptation data makes no difference whatsoever. One adaptation utterance performs better in some methods than two or three utterances. Apart from the right most column, the results are approximately same for any given number of utterances. However, the case is different for males. Table 8 shows that results for any given number of utterances adapted using any methods vary considerably. First iteration of MLLR is further improved by second iteration for both males and females. MAP starts with the worst result of all, decreases the WER and eventually gives the best results of all for three adaptation utterances. For MAP only, the relative decline in WER is 38% from one to three adaptation utterances. This is a significant improvement considering the amount of noise. This shows the importance of noise and amount of data for MAP adaptation. Both methods which combine MAP with MLLR provide consistent results as is the case for females.

Overall, MAP adaptation followed by two iterations of MLLR produces the most consistent results among all methods. However, the purpose is to obtain the transcriptions which contain least number of errors. For this reason, MAP is selected as the method for speaker adaptation in the mobile applications `Talk'n'label` and `Talk'n'pick`. The application-specific adaptation data is prepared and recommended for the mobile application, which conforms with the experiments and results presented so far.

6.4 Confidence Scoring

The value of confidence scoring is a rational number between 0 and 1 which can be used to determine if the output transcription of the system is sufficiently reliable to be used in any given task. The algorithm for confidence scoring is fairly simple. The word graphs are constructed based on language scores and acoustic score of each respective path (edge) containing two different nodes. Combining all the edge probabilities provides with the posterior probability value of a certain hypothesized word. This score is then compared to a pre-determined threshold and all the words which

have lesser probability are discarded, and the remaining ones having equal or higher probability are accepted. Those words are used as the final output transcription.

One important element of those errors which has not been discussed so far is the nature of an error in transcription. One type of errors is *false acceptance*, which means that either a false word is labelled as correct or recogniser generates a word even though there is none. The other type is *false rejection* in which system rejects a word which is indeed correct and should be present in the final transcription. The ratio of these varies with the choice of tagging threshold. A trade-off should be considered between these two types of error before arriving at a conclusion for a suitable threshold value. For speaker adaptation purposes this difference carries no effect on the performance since the total number of incorrect labels is counted and WER is computed. The performance remains the same as long as the total number of inaccurate labels is the same. For confidence scoring in general, this is an important concern.

The confidence error rate (CER) is also used as a metric for evaluating the confidence scoring performance [3]. CER is defined as follows:

$$CER(\%) = \frac{I + S}{N} \cdot 100\% \tag{44}$$

Wessel et al. [3] consider this definition using the number of insertions and substitutions only. On the other hand, a slightly modified definition is provided in [32] which uses the total number of incorrectly classified words. In these experiments, deletions affect the accuracy so the both these definitions are combined for this thesis. This thesis interprets CER as the number of false acceptances and false rejections divided by total number of words. This makes CER similar to WER, thus we use WER in confidence scoring as well.

The selection of criterion mainly depends upon the desire of the user or requirement of the concerning task. Since both the mobile applications are used in actual working environments where a variety of noises are present, the ultimate goal of this task is to produce the outcome with a reasonable confidence and least number of total errors. For this reason, WER serves as a suitable criterion for assessment of the results.

6.4.1 Results

Since two important tasks are implemented on the acoustic model, it is logical to merge these tasks and observe the combined results as well. In this case, both adapted or unadapted acoustic models are used for confidence scoring purposes. Confidence scoring is applied using the threshold values on different models and resulting WERs are presented in tables 10-12. These results are an average of the same 10 speakers from evaluation dataset. The results against a particular threshold value show that all the words having lesser posterior probability than that value are discarded and the WER of resulting transcription is reported. In these tables, the results against the

threshold value 0.1 approximately represent the same as those of speaker adaptation. These are not exactly the same models because there are a few words with posterior probability value less than 0.1 which are discarded in this case, hence there is a slight difference in the respective values of WER.

Table 10: WER for one adaptation utterance using different confidence threshold values

	0.1	0.2	0.4	0.6	0.8	0.9
semi	1.90	1.92	2.04	2.35	2.86	3.21
MLLR1	1.66	1.68	1.89	2.17	2.52	2.93
MLLR2	1.47	1.55	1.60	1.98	2.44	2.70
MAP	1.68	1.75	1.97	2.15	2.71	3.33
MAP+MLLR1	1.57	1.64	1.82	2.12	2.46	2.88
MAP+MLLR2	1.49	1.56	1.64	1.91	2.41	2.95

Table 11: WER for two adaptation utterances using different confidence threshold values

	0.1	0.2	0.4	0.6	0.8	0.9
semi	1.90	1.92	2.04	2.35	2.86	3.21
MLLR1	1.61	1.66	1.86	2.16	2.58	3.04
MLLR2	1.30	1.50	1.55	1.98	2.36	2.91
MAP	1.41	1.52	1.71	1.99	2.54	2.98
MAP+MLLR1	1.60	1.65	1.86	2.20	2.59	2.94
MAP+MLLR2	1.53	1.58	1.70	2.06	2.41	2.81

Table 12: WER for three adaptation utterances using different confidence threshold values

	0.1	0.2	0.4	0.6	0.8	0.9
semi	1.90	1.92	2.04	2.35	2.86	3.21
MLLR1	1.61	1.63	1.79	2.11	2.53	3.04
MLLR2	1.48	1.51	1.64	1.92	2.38	2.92
MAP	1.39	1.41	1.62	1.82	2.31	2.92
MAP+MLLR1	1.55	1.59	1.87	2.14	2.55	2.96
MAP+MLLR2	1.53	1.58	1.67	2.01	2.52	3.03

It is evident from the speaker adaptation results that in the beginning of the process when less amount of data is provided, MAP produces worst results of all and improvement is achieved gradually. The same trend can be observed in confidence scoring experiments. As the number of adaptation utterances increases to three, it can be seen that all the models are behaving similar as in the case of speaker adaptation

experiments. The model which is adapted using MAP, is improving consistently for all given adaptation utterances and produces the least WER for three adaptation utterances. The same results are visible for all mentioned tagging threshold values. In these tables, the overall WER of all model continues to rise as the threshold value is increased. The reason is that different words have different posterior probability values. Some words have high probability than others. Sometimes, there are words which are correctly recognised but the probability value assigned to those is not very high. For instance, when a correctly recognised word has a probability value of 0.55 and the threshold is set to 0.6, the word is simply discarded. This applies to all other similar words which are recognised correctly but may not have a higher probability value with respect to the set threshold and are rejected from the output transcription. As the acceptance threshold surges, more words are disqualified from final transcription which in turn increases the WER.

In the first columns of these tables, the same trend can be observed as is the case in speaker adaptation. The relative increment in performance of MLLR-adapted model compared to semi-continuous model is 12.6%, while MAP followed by two iterations of MLLR produces the best result for 0.1 threshold value. Similar trends continue for remaining threshold values. If 0.9 tagging threshold value is considered, most of the words with good recognition are unable to fulfil this highly strict benchmark. As a result, even the words with generally good recognition and higher confidence are not allowed in the output. A positive aspect of this higher threshold is that the number of incorrect substitutions and insertions in the output drops down, as is shown in the following figures. This means that number of words, which are falsely accepted, plummets. The resulting WER should remain approximately similar to that of original model because one of the involved quantities, is increasing and other is decreasing. This is however not the case here. The reason is that, as the tagging threshold increases, the number of incorrect acceptances becomes less and less. On the other hand, the number of words still remains the same which are closer to but lesser than the threshold value and are discarded afterwards. This means that the increase in false rejection rate is significant as compared to the decrease in false acceptance rate. Hence, the overall WER increases.

In general, the output decrease gradually from second to fifth column in these tables, which shows WER for 0.6 threshold value. Moving from threshold value 0.1 to 0.6, a relative decrease of 27% in all the models can be seen on average. This means that in this range of posterior probability values, there are not a lot of correct words which are discarded. When 0.8 is selected as the probability threshold, further decrease in performance is of 25% on average as compared to that of 0.1 and similarly, approximately 20% further decline is observed when threshold is set to 0.9. This infers that in the posterior probability value range $0.6 - 0.9$, there are a significant number of words which are eliminated if higher threshold values such as 0.8 and 0.9 are imposed. Though the resulting transcription may carry highest confidence, the associated WER is a major concern. Based on these results, a suitable threshold value can be selected depending upon the requirements of the task.

In following figures, the term `false acc semi` represents the number of incorrectly accepted words for the semi-continuous model and `false rej MAP+MLLR2` shows the number of incorrectly accepted words for the model which is adapted using MAP followed by two iterations of MLLR. Similarly other entries represent different combinations of adaptation and tagging thresholds. On the horizontal axis, the threshold value for acceptance is plotted against the number of incorrect tags on the vertical axis. Different models are assessed in terms of these errors in recognition using different amounts of adaptation data and the corresponding results are visualized.

In these experiments, the results in Figures 12-14 have helped decide what should be a viable tagging threshold value for both the mobile applications. This value should be selected so as to minimize the total number of resulting errors. At the same time, it should be sufficiently high because it represents the amount of confidence. In general, the optimum threshold value can be found in the vicinity of the intersection of two curves i.e. false acceptance and false rejection. For a different task, the value can be chosen differently which may either reduce the number of false acceptance on the cost of false acceptances and vice versa.

It can be inferred from the following figures that the number of false acceptances is plummeting as the threshold is increased. On the other hand, the number of words which are incorrectly rejected is growing. To comprehend this result, consider the posterior probabilities of individual words. For the lowest tagging threshold of 0.1, the WER appears to be the least because usually there are no false rejections and the most number of false acceptances at this point. In case of higher threshold values e.g. 0.8 and 0.9, the situation changes and all those incorrectly accepted words are blocked. The expense here is the number of incorrect rejections which increases in this case. Now consider the range $[0.2, 0.6]$. Several correct words lie in this range and some incorrect words as well. As the threshold increases beyond 0.6, those incorrect words are eliminated which were present before. At the same time, the actual correct words lying in this range are also ignored due to the binary classification. As the value increases, there are less and less number of words which fulfil the strictness of this criterion. Although the system may have reassuring confidence in the obtained transcription, the WER is elevated which may not be the requirement in some cases. In this situation, the trade-off should be considered and the amount of confidence has to be compromised in order to keep the WER within acceptable range.

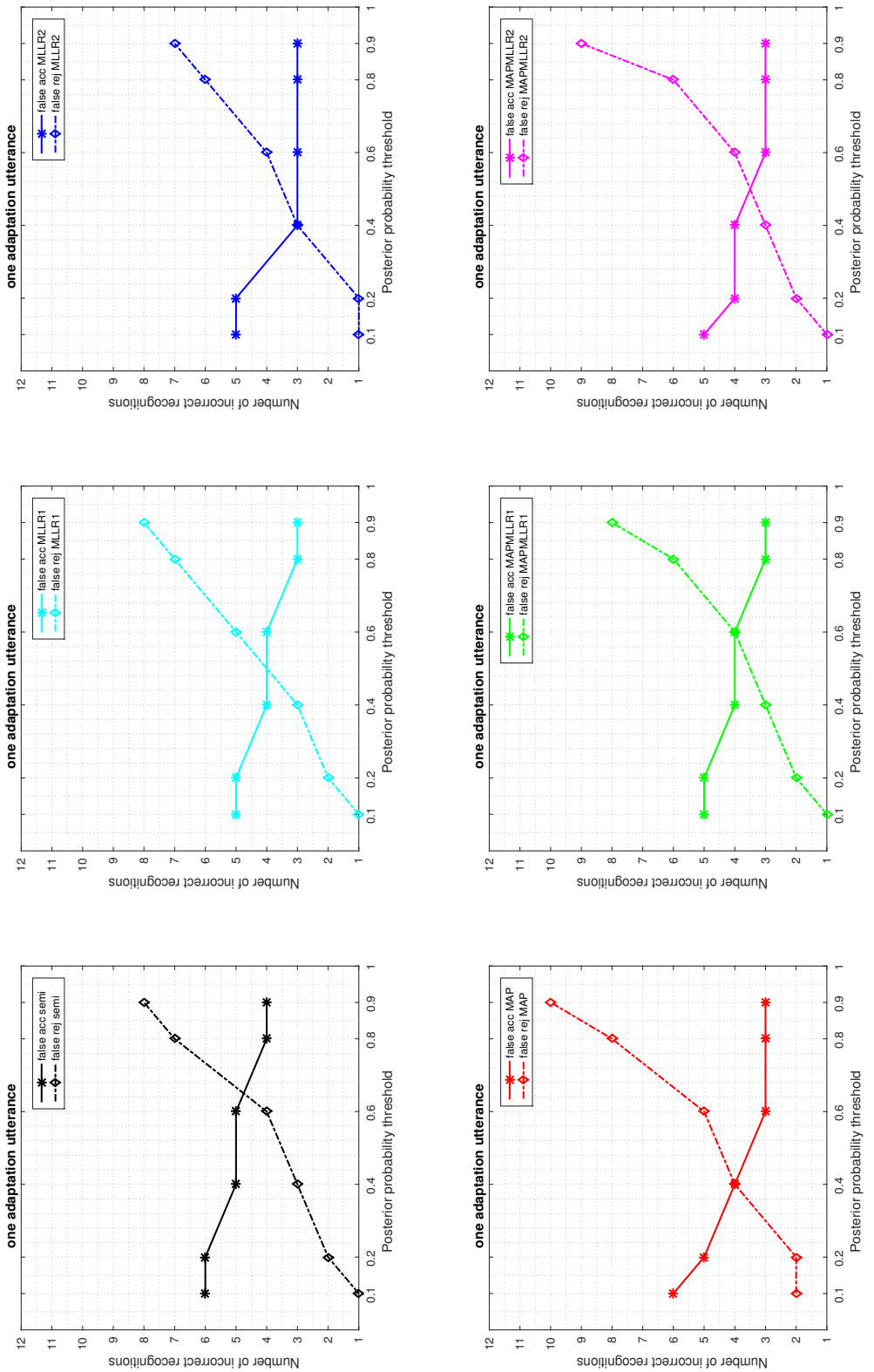

Figure 12: Individual errors for confidence scoring using one adaptation utterance

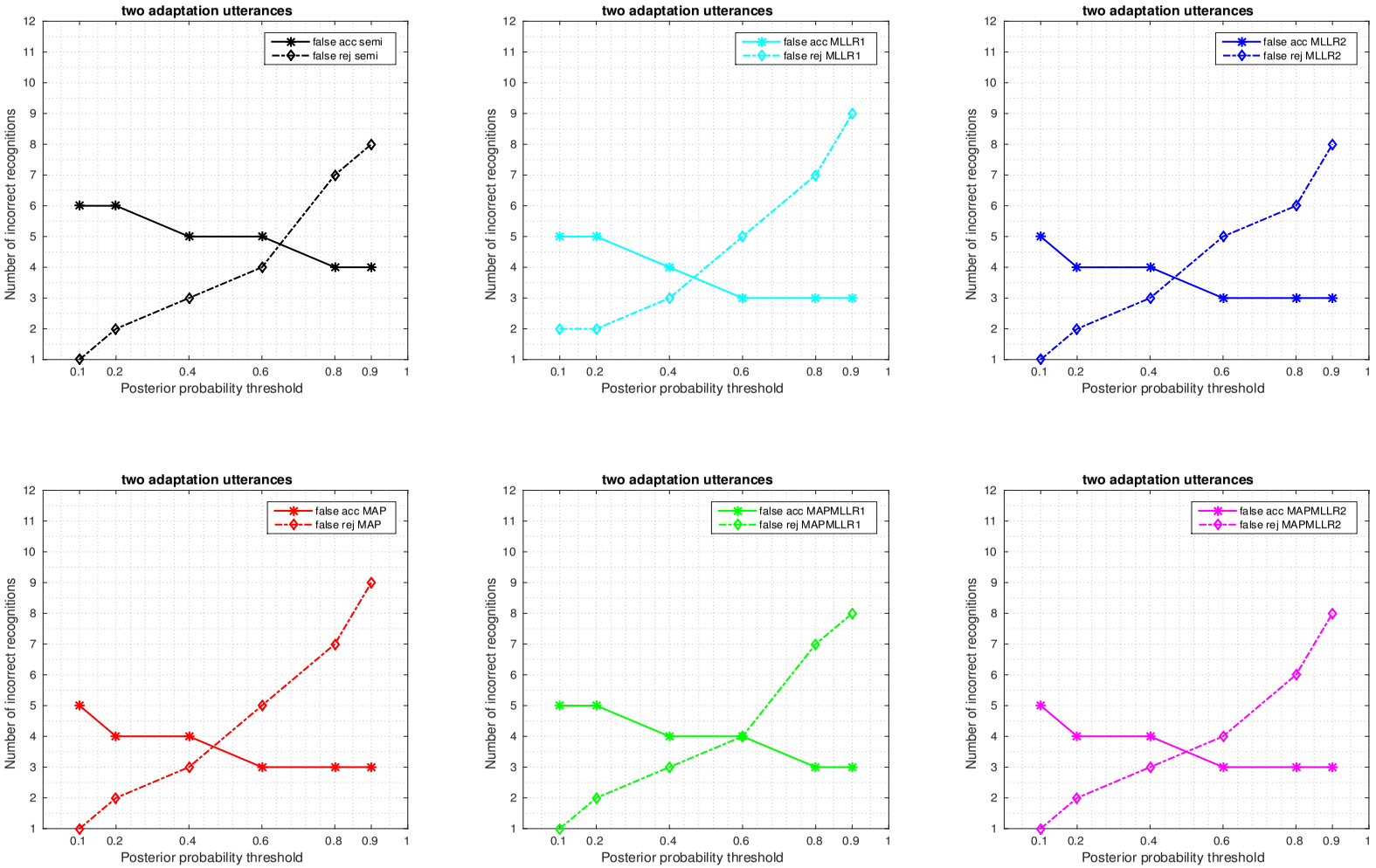

Figure 13: Individual errors for confidence scoring using two adaptation utterances

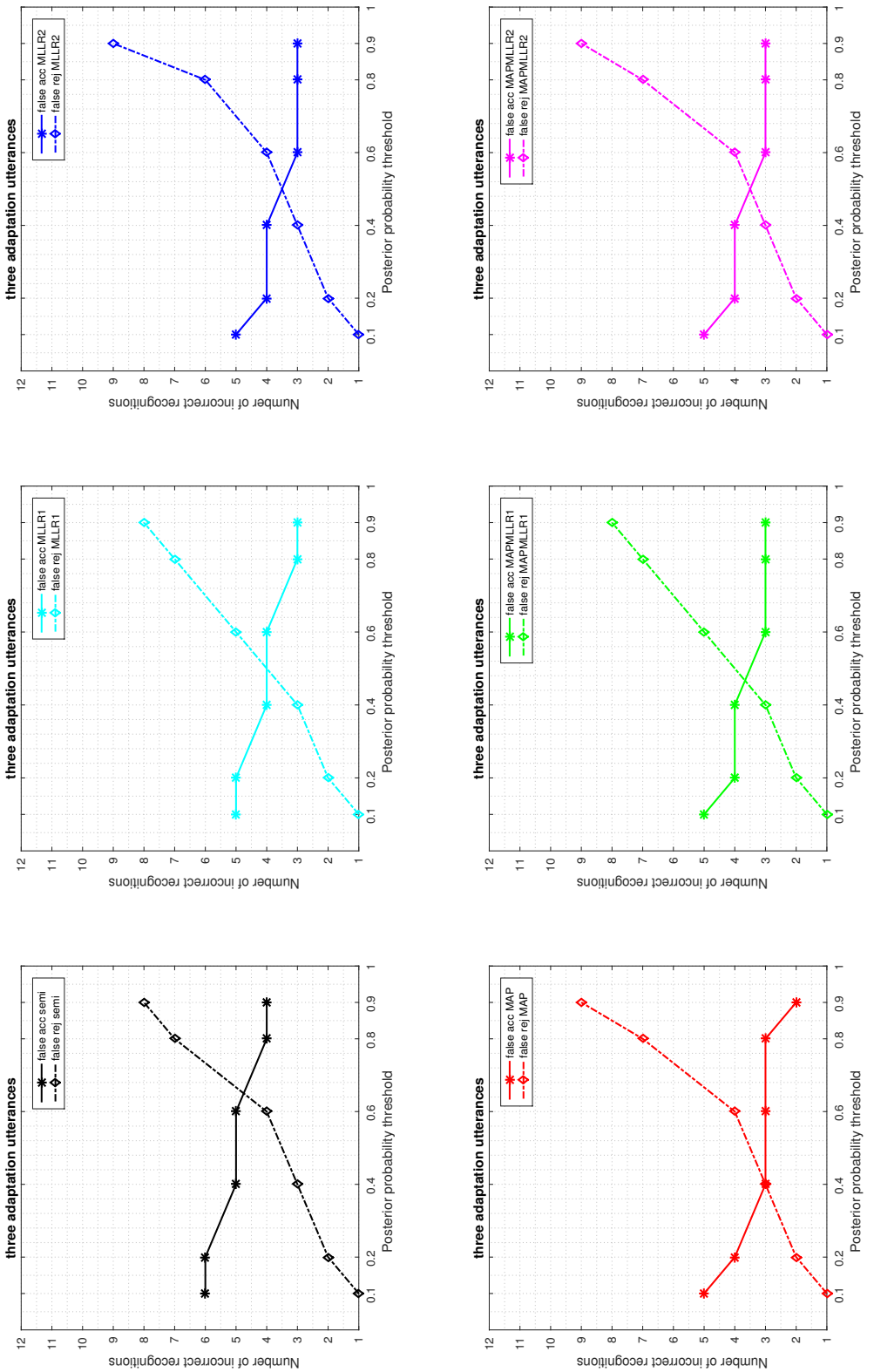

Figure 14: Individual errors for confidence scoring using three adaptation utterances

7 Discussion

This thesis performs three tasks which address to improve an existing mobile application. First, an acoustic model is trained given the training data of actual working environments. The entire data is in Finnish language. The standard tools from CMU Sphinx toolkit are used for all the tasks in this thesis. Two different models are trained and compared in terms of % WER. Both PTM and semi-continuous models have strengths and weaknesses. The trained semi-continuous model is compared with the model previously used in the application. This new semi-continuous model outperforms the previous model and brings 60% and 65% relative improvement in WER on evaluation and testing sets respectively. This shows that robust models can be trained using noisy data and specified vocabulary. In current training scheme, PTM model outperforms semi-continuous model by smaller margins. The users of the application on mobile platform have provides their feedback on both models. Taking into account the processing speeds and accuracy of both models, semi-continuous model is recommended for use in the application.

Then next step is to apply two different techniques on the acoustic model in order to improve the accuracy of output. Firstly, speaker adaptation is implemented using two methods. The most notable ones i.e. MAP and MLLR are applied in different combinations. In theory MLLR adaptation gives better results in case of small amount of adaptation data and then MAP adaptation surpasses MLLR adaptation once the data is sufficient enough. The results of experimentation performed on the data are found to be consistent with the theoretical concepts. Two iterations of MLLR are applied and the second iteration seems to further reduce the WER. On the other hand, only one iteration is considered for MAP adaptation in this thesis. For one adaptation utterance, first and second iterations of MLLR produce 8% and 18% relative improvement respectively over MAP. Once the number of utterances is increased to two, MAP matches the performance of MLLR and finally provides better results for three utterances. Both MAP and MLLR adaptation are combined to find the extent of improvement and it increases the performance by 18% relative to the baseline model. In general, MAP appears to perform better than all other methods. MAP adaptation reduces 26% compared to the unadapted model for three adaptation utterances. As the final suggestion, MAP is proposed as the technique to be used in the mobile application for adaptation purposes.

Second task is the application of confidence scoring on the acoustic model. Confidence scoring is implemented on the baseline model and different adapted versions of the model and, performance of the model is examined. Word posterior probability is presented as the implemented method. These probability values are calculated using word graphs. A binary classifier is then used to obtain the output words with the probability value higher than the threshold. Word error rates of different models against different threshold values are presented. A suitable threshold value is recommended for usage in both the applications, based on the results obtained

during the experiments. The main consideration is to maintain a low WER while retaining a reasonable degree of confidence in the output transcription. The optimum value for this threshold can be found by maintaining the number of false acceptance to a minimum and not allowing the number of false rejections to increase drastically. Confidence scoring is proposed as an optional feature because it increase the amount of confidence in the output hypotheses on the expense of increased number of false rejections. In order to reduce the number of false rejections, the user can be requested to utter the rejected words more clearly. The use of confidence scoring may depend upon the requirement of user and nature of the task. If overall WER is the only concern for the system, then this may not be an ideal feature to be included. However, confidence scoring can play an important role in the cases where the main consideration is to reduce the number of false recognitions.

One reason that the results of the trained acoustic model are better than that of previous year is that the training data contains noise to a certain extent. This makes the model comparatively effective and robust towards degradation. One suggestion is to include relatively more diverse data for all purposes. In particular, adding more non-native speakers and increasing the number of non-native females to the data may help the cause. In case of training phase, if the ratio between males and females is more balanced, the model is likely to produce better outcome. Compared to the males speakers, generally noise is present in lesser amount in the speech of females. If female speech data with some more noise is made available, the resulting acoustic model may perform even better. Further experiments are needed to check how the reductions of data affects the model performance. One may also collect even more data for adaptation and especially training purposes to evaluate what is the model performance.

Considering the amount of adaptation data, it may be interesting to evaluate the results if the model has less data available for adaptation. Experiments may show the trend if the adaptation data is reduced because it is laborious for the user to record long speech files for a very negligible improvement in WER. A reasonable compromise may be attained here between the amount of adaptation data and the errors in output. For confidence scoring purposes, other mentioned methods can be considered for application. In this way, different methods can be compared and that method can be chosen which meets the requirements of the user and the application.

References

[1] "Finnish phonetics, department of phonetics university of helsinki," available at http://www.helsinki.fi/puhetieteet/projektit/Finnish_Phonetics/vokaaliakustiikka_eng.htm. Accessed on 15.07.2016.

[2] M. Karjalainen, "Kommunikaatioakustiikka." Helsinki University of Technology, 1999.

[3] F. Wessel, R. Schluter, K. Macherey, and H. Ney, "Confidence measures for large vocabulary continuous speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 3, pp. 288–298, Mar 2001.

[4] M. S. Seigel, "Confidence estimation for automatic speech recognition hypotheses," Ph.D. dissertation, Ph. D. thesis, University of Cambridge, 2013.

[5] H. Jiang, "Confidence measures for speech recognition: A survey," *Speech communication*, vol. 45, no. 4, pp. 455–470, 2005.

[6] C. H. Lee, C. H. Lin, and B. H. Juang, "A study on speaker adaptation of the parameters of continuous density hidden markov models," *IEEE Transactions on Signal Processing*, vol. 39, no. 4, pp. 806–814, Apr 1991.

[7] B.-H. Juang and L. R. Rabiner, "Automatic speech recognition–a brief history of the technology development," *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara*, vol. 1, p. 67, 2005.

[8] K. Davis, R. Biddulph, and S. Balashek, "Automatic recognition of spoken digits," *The Journal of the Acoustical Society of America*, vol. 24, no. 6, pp. 637–642, 1952.

[9] H. Dudley, R. Riesz, and S. Watkins, "A synthetic speaker," *Journal of the Franklin Institute*, vol. 227, no. 6, pp. 739–764, 1939.

[10] H. Fletcher, "The nature of speech and its interpretation," *Journal of the franklin institute*, vol. 193, no. 6, pp. 729–747, 1922.

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

[12] H. Murveit, M. Cohen, P. Price, G. Baldwin, M. Weintraub, and J. Bernstein, "Sri's decipher system," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1989, pp. 238–242.

[13] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey *et al.*, "The htk book," *Cambridge university engineering department*, vol. 3, p. 175, 2002.

[14] L. Bahl, P. F. Brown, P. V. De Souza, and R. L. Mercer, "Maximum mutual information estimation of hidden markov model parameters for speech recognition," in *proc. icassp*, vol. 86, 1986, pp. 49–52.

[15] L. Rabiner and B.-H. Juang, "Fundamentals of speech recognition," Prentice Hall, 1993.

[16] X. Zhao and D. Wang, "Analyzing noise robustness of mfcc and gfcc features in speaker identification," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 7204–7208.

[17] B. T. Meyer and B. Kollmeier, "Optimization and evaluation of gabor feature sets for asr." in *INTERSPEECH*, 2008, pp. 906–909.

[18] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[19] L. E. Baum, J. A. Eagon *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*, vol. 73, no. 3, pp. 360–363, 1967.

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

[21] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE transactions on acoustics, speech, and signal processing*, vol. 37, no. 3, pp. 328–339, 1989.

[22] M. Rosenblatt, *Markov Chains*. New York, NY: Springer New York, 1974, pp. 36–67. [Online]. Available: http://dx.doi.org/10.1007/978-1-4612-9852-6_3

[23] G. A. Fink, "n-gram models," in *Markov Models for Pattern Recognition*. Springer, 2014, pp. 107–127.

[24] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 179–190, 1983.

[25] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.

[26] "CMU Sphinx, a project by Carnegie Mellon University," 2016, available at http://cmusphinx.sourceforge.net/. Accessed on 25.07.2016.

[27] "CMU Sphinx's open source code," 2016, available at https://sourceforge.net/projects/cmusphinx/. Accessed on 18.04.2016.

[28] H. Bourlard, H. Hermansky, and N. Morgan, "Towards increasing speech recognition error rates," *Speech communication*, vol. 18, no. 3, pp. 205–231, 1996.

[29] S. R. Young, "Detecting misrecognitions and out-of-vocabulary words," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994*, vol. ii, Apr 1994, pp. II/21–II/24 vol.2.

[30] J. G. Wilpon, L. R. Rabiner, C.-H. Lee, and E. Goldman, "Automatic recognition of keywords in unconstrained speech using hidden markov models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 11, pp. 1870–1878, 1990.

[31] M. K. Abida, "Fuzzy gmm-based confidence measure towards keywords spotting application," University of Waterloo, 2007.

[32] N. Q. Luong, "Word confidence estimation and its applications in statistical machine translation," Ph.D. dissertation, Grenoble, 2014.

[33] T. J. Hazen, S. Seneff, and J. Polifroni, "Recognition confidence scoring and its use in speech understanding systems," *Computer Speech & Language*, vol. 16, no. 1, pp. 49–67, 2002.

[34] T. Kemp, T. Schaaf *et al.*, "Estimating confidence using word lattices." in *EuroSpeech*, 1997.

[35] A. Sanchis, A. Juan, and E. Vidal, "Estimating confidence measures for speech recognition verification using smoothed naive bayes model," in *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2003, pp. 910–918.

[36] J. Pinto and R. Sitaram, "Confidence measures in speech recognition based on probability distribution of likelihoods." in *INTERSPEECH*, 2005, pp. 3001–3004.

[37] T. Zeppenfeld, M. Finke, K. Ries, M. Westphal, and A. Waibel, "Recognition of conversational telephone speech using the janus speech engine," in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, Apr 1997, pp. 1815–1818 vol.3.

[38] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig, and A. Stolcke, "Neural-network based measures of confidence for word recognition," in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, Apr 1997, pp. 887–890 vol.2.

[39] E. Lleida and R. C. Rose, "Likelihood ratio decoding and confidence measures for continuous speech recognition," in *Fourth International Conference on Spoken Language, 1996. ICSLP 96. Proceedings.*, vol. 1, Oct 1996, pp. 478–481 vol.1.

[40] R. C. Rose, B.-H. Juang, and C.-H. Lee, "A training procedure for verifying string hypotheses in continuous speech recognition," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1. IEEE, 1995, pp. 281–284.

[41] M. G. Rahim, C.-H. Lee, and B.-H. Juang, "Discriminative utterance verification for connected digits recognition," *IEEE transactions on speech and audio processing*, vol. 5, no. 3, pp. 266–277, 1997.

[42] F. Wessel, K. Macherey, and R. Schluter, "Using word probabilities as confidence measures," in *IEEE International Conference on Acoustics, Speech and Signal Processing, Proceedings of 1998.*, vol. 1, May 1998, pp. 225–228 vol.1.

[43] F. Wessel, K. Macherey, and H. Ney, "A comparison of word graph and n-best list based confidence measures." in *EuroSpeech*, 1999.

[44] F. Wessel, "Word posterior probabilities for large vocabulary continuous speech recognition," Ph.D. dissertation, Bibliothek der RWTH Aachen, 2002.

[45] F. Wessel, R. Schluter, and H. Ney, "Using posterior word probabilities for improved speech recognition," in *IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings of 2000*, vol. 3, 2000, pp. 1587–1590 vol.3.

[46] B. Rueber, "Obtaining confidence measures from sentence probabilities." in *Eurospeech*, 1997.

[47] M. Weintraub, "Lvcsr log-likelihood ratio scoring for keyword spotting," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, May 1995, pp. 297–300 vol.1.

[48] X. Huang and K.-F. Lee, "On speaker-independent, speaker-dependent, and speaker-adaptive speech recognition," *IEEE Transactions on Speech and Audio processing*, vol. 1, no. 2, pp. 150–157, 1993.

[49] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[50] M. Gales and S. Young, "The application of hidden markov models in speech recognition," *Foundations and trends in signal processing*, vol. 1, no. 3, pp. 195–304, 2008.

[51] C. Leggetter and P. Woodland, "Flexible speaker adaptation using maximum likelihood linear regression," in *Proc. ARPA Spoken Language Technology Workshop*, vol. 9. Citeseer, 1995, pp. 110–115.

[52] M. J. Gales and P. C. Woodland, "Mean and variance adaptation within the mllr framework," *Computer Speech & Language*, vol. 10, no. 4, pp. 249–264, 1996.

[53] M. Bacchiani and B. Roark, "Unsupervised language model adaptation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003*, vol. 1, April 2003, pp. I–224–I–227 vol.1.

[54] L. Lamel, J. L. Gauvain, and G. Adda, "Unsupervised acoustic model training," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2002.*, vol. 1, May 2002, pp. I–877–I–880.

[55] M. J. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer speech & language*, vol. 12, no. 2, pp. 75–98, 1998.

[56] J.-L. Gauvain and C.-H. Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *IEEE transactions on speech and audio processing*, vol. 2, no. 2, pp. 291–298, 1994.

[57] A. Sankar and C.-H. Lee, "Robust speech recognition based on stochastic matching," in *International Conference on Acoustics, Speech, and Signal Processing,*, vol. 1, May 1995, pp. 121–124 vol.1.

[58] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[59] C. Leggetter and P. C. Woodland, "Speaker adaptation of continuous density hmms using multivariate linear regression." in *ICSLP*, vol. 94. Citeseer, 1994, pp. 451–454.

[60] M. J. Gales, *The generation and use of regression class trees for MLLR adaptation.* University of Cambridge, Department of Engineering, 1996.

[61] K.-t. Chen, W.-w. Liau, H.-m. Wang, and L.-s. Lee, "Fast speaker adaptation using eigenspace-based maximum likelihood linear regression." in *INTERSPEECH*, 2000, pp. 742–745.

[62] M. Gales and S. Young, "The application of hidden markov models in speech recognition," *Found. Trends Signal Process.*, vol. 1, no. 3, pp. 195–304, Jan. 2007. [Online]. Available: http://dx.doi.org/10.1561/2000000004

[63] O. Siohan, C. Chesta, and C.-H. Lee, "Joint maximum a posteriori adaptation of transformation and hmm parameters," *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 4, pp. 417–428, 2001.

[64] S.-J. Hahm, S. Watanabe, A. Ogawa, M. Fujimoto, T. Hori, and A. Nakamura, "Prior-shared feature and model space speaker adaptation by consistently employing map estimation," *Speech Communication*, vol. 55, no. 3, pp. 415–431, 2013.

[65] Q. Huo and B. Ma, "Online adaptive learning of continuous-density hidden markov models based on multiple-stream prior evolution and posterior pooling," *IEEE transactions on speech and audio processing*, vol. 9, no. 4, pp. 388–398, 2001.

[66] C. H. Lee, C. H. Lin, and B. H. Juang, "A study on speaker adaptation of continuous density hmm parameters," in *International Conference on Acoustics, Speech, and Signal Processing*, Apr 1990, pp. 145–148 vol.1.

[67] C. H. Lee and J. L. Gauvain, "Speaker adaptation based on map estimation of hmm parameters," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, April 1993, pp. 558–561 vol.2.

[68] I. A. McCowan, D. Moore, J. Dines, D. Gatica-Perez, M. Flynn, P. Wellner, and H. Bourlard, "On the use of information retrieval measures for speech recognition evaluation," IDIAP, Tech. Rep., 2004.

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

[70] X. Huang, H.-W. Hon, and K.-F. Lee, "Large-vocabulary speaker-independent continuous speech recognition with semi-continuous hidden markov models," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1989, pp. 276–279.

[71] X. D. Huang and M. A. Jack, "Semi-continuous hidden markov models for speech signals," *Computer Speech & Language*, vol. 3, no. 3, pp. 239–251, 1989.

[72] A. Lee, T. Kawahara, K. Takeda, and K. Shikano, "A new phonetic tied-mixture model for efficient decoding," in *IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, vol. 3, 2000, pp. 1269–1272 vol.3.

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

[74] P. Pujol, D. Macho, and C. Nadeu, "On real-time mean-and-variance normalization of speech recognition features," in *IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 1, May 2006, pp. I–I.