Muhammad Salman Ishaq

Voice Activity Detection and Garbage Modelling for a Mobile Automatic Speech Recognition Application

School of Electrical Engineering

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

Espoo 23.1.2017

Thesis supervisor:

Prof. Mikko Kurimo

Thesis advisors:

Matti Varjokallio

Leo Hämäläinen

Aalto University
School of Electrical Engineering

AALTO UNIVERSITY SCHOOL OF ELECTRICAL ENGINEERING	ABSTRACT OF THE MASTER'S THESIS

Author: Muhammad Salman Ishaq

Title: Voice Activity Detection and Garbage Modelling for a Mobile Automatic Speech Recognition Application

Date: 23.1.2017 Language: English Number of pages: 8+55

Department of Signal Processing and Acoustic

Professorship: Speech Recognition Code: S3013

Supervisor: Prof. Mikko Kurimo

Advisors: Matti Varjokallio, Leo Hämäläinen

Recently, state-of-the-art automatic speech recognition systems are used in various industries all over the world. Most of them are using a customized version of speech recognition system. The need for different versions arise due to different speech commands, lexicon, language and distinct work environment. It is essential for a speech recognizer to provide accurate and precise outputs in every working environment. However, the performance of a speech recognizer degrades quickly when noise intermingles with a work environment and also when out-of-vocabulary (OOV) words are spoken to the speech recognizer.

This thesis consists of three different tasks which improve an automatic speech recognition application for mobile devices. The three tasks include building of a new acoustic model, improving the current voice activity detection and garbage modelling of OOV words.

In this thesis, firstly, a Finnish acoustic model is trained for a company called Devoca Oy. The training data was recorded from different warehouse environments to improve the real-world speech recognition accuracy. Secondly, the Gammatone and Gabor features are extracted from the input speech frame to improve the voice activity detection (VAD). These features are applied to the VAD decision module of *Pocketsphinx* and a new *neural-network* classifier, to be classified as speech or non-speech. Lastly, a garbage model is developed for the OOV words. This model recognizes the words from outside the grammar and marks them as unknown on the application interface.

This thesis evaluates the success of these three tasks with Finnish audio database and reports the overall improvement in the word error rate.

Keywords: Voice activity detection, Garbage modelling, Out of vocabulary, Neural network classifier

Preface

I want to thank my Professor Mikko Kurimo and instructors Leo Hämäläinen and Matti Varjokallio for their good and valuable guidance which helped me a lot in completing this master's thesis. I also want to thank Muhammad Hashim Abbas for all the help and support during this thesis.

I also want to pay my gratitude to my parents Mr & Mrs. Ishaq Hussain and beloved wife Mehak Salman for their consistent support and prayers. Special thanks to my elder sister Rabia Toseef and brother Awais Ishaq for motivating me frequently to complete this thesis.

Most importantly, I would like to thank Almighty God, for his blessings and support during my studies.

Otaniemi, 23.01.2017 Muhammad Salman Ishaq

Contents

Abbreviations

AM	Acoustic Model
ASR	Automatic Speech Recognition
ANN	Artificial Neural Network
DCT	Discrete Cosine Transform
DNN	Deep Neural Network
ERB	Equivalent Rectangular Bandwidth
FFT	Fast Fourier Transform
FST	Finite State Transducer
FPA	False Positive Alarm
GFCC	Gammatone Frequency Cepstral Coefficient
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
IV	In Vocabulary
LER	Letter Error Rate
LM	Language Model
MFCC	Mel Frequency Cepstral Coefficient
MLE	Maximum Likelihood Estimation
MLP	Multi Layer Perceptron
MS WAV	Microsoft Waveform Audio File
OOV	Out of Vocabulary
PLP	Perceptual Linear Prediction
PNCC	Power Normalized Cepstral Coefficient
PTM	Phonetically Tied Model
ROC	Receiver Operation Characteristics
SNR	Signal to Noise Ratio
SVSR	Small Vocabulary Speech Recognition
TPA	True Positive Alarm
TTS	Text to Speech
VAD	Voice Activity Detection
WER	Word Error Rate

List of Figures

List of Tables

Chapter 1

1 Introduction

In recent years, plenty of high-quality research and experiments have been conducted in the field of automatic speech recognition (ASR) technology, which certainly improve and provide the stability to the speech recognition systems. The ASR systems have become extremely important as they can perform precise recognition in real-world applications. The recent advancement in ASR technology enables the human to have effective and meaningful interaction or conversation with machines in a more natural way. Despite all the good work being done recently, the present day ASR systems still have some persistent challenges which can degrade their accuracy and performance during their use in adverse environmental conditions.

One of the most important challenges for an ASR system is to work fairly in a non-stationary noisy environment. When the level of background noise increases, most of the voice activity detection (VAD) algorithms fail as the classification of speech and noise becomes almost impossible. In order to tackle this issue, an efficient VAD algorithm is required which can differentiate between speech and noise to make an ASR more robust even in difficult environments. Many VAD algorithms have been developed so far, as a result of continuous and hard work done in this field of ASR systems. Most of the developed algorithms have different feature extraction techniques and decision mechanism. The recent trend is to extract multiple features from a speech signal and feed them to the decision module to classify between speech and non speech [3, 4, 5].

Another notable challenge for an ASR system is to identify every possible word, sequence of words and even out of vocabulary (OOV) words which belong to different topics. A comprehensive language model, trained and adapted from a very generic subject corpus can accomplish this task by assigning correct and reasonable probabilities to the word sequences during the speech decoding process. The generic language model can perform reasonably well for a large vocabulary speech recognition systems deployed on large servers with many available resources. However, it is not an ideal approach to use generic language model adapted from universal subject corpus for embedded ASR applications like dictation and programs searching by voice, which usually employ a limited vocabulary. Since these mobile applications can work much better and faster if the customized grammar is used and also most of the embedded platforms have not much available resources (i.e. CPUs) and computational complexity for searching word sequences might grow.

The work around for such a customized-vocabulary ASR system is the garbage modelling for OOV words. The basic idea behind garbage modelling is to find human produced non-words and broken words. Additionally, it can also be used to identify the words from outside the vocabulary. Once the word is recognized as unknown, the application should be able to notify the user and again ask for a correct input. The other task is the consistent working of an acoustic model for male and female speakers of every age. This can be handled by incorporating more speech data (with equal amount of male and female voices) during the training of an acoustic model.

1.1 Objectives of the work

The main objective of this thesis work is to bring an advancement in a speech recognition product owned by a Finnish company specialized in industrial ASR solutions. Improvement of the speech recognition product consists of three tasks which include the building of a new acoustic model, improvement of the current VAD module and the garbage modelling of OOV words that contributes towards improving an existing ASR system. The first task is to train an acoustic model by using CMU-Sphinx toolkit which works better for both male and female voices. Also the acoustic model should be able to work fairly in noisy environment. The second task is to replace an existing VAD module with a new and efficient VAD module which works better in all noisy environments. The final task is to develop a new garbage model to handle OOV words and non-words. This module will play its role when any unknown or new word is spoken to the system by rejecting that particular word.

1.2 Target audience

This thesis is mainly relevant for the master's students and developers who want to build a new acoustic model in their own language by using the CMU-Sphinx toolkit and also wish to improve the accuracy of their automatic speech recognition mobile applications by using the efficient voice activity detection and garbage modelling algorithms explained in this work.

1.3 Thesis breakdown

The thesis has been organized in the following manner. Chapter 2 gives an insight on the basics or fundamentals of speech and discusses the automatic speech recognition in detail. It also explains an open-source ASR engine i.e. CMU-Sphinx speech recognition toolkit. In Chapter 3, small vocabulary system is presented which embeds the developed ASR application. Chapter 4 illustrates the training and building of acoustic models in detail. In Chapter 5, different VAD algorithms and techniques are presented, and one of them is implemented in the mobile ASR application. Chapter 6 discusses the garbage modelling for OOV and non words. Chapter 7 concludes the important findings of this thesis work and also provides some recommendations for the future work.

Chapter 2

2 Background of Speech Recognition

This thesis discusses the problems mentioned in Chapter 1 in detail and provides respective solutions for a mobile based ASR application. Before proceeding to the solutions of the highlighted problems, the theory behind the basics of speech and ASR process is provided in this chapter.

2.1 Fundamentals of speech

Speech is an effective way of communicating ideas between humans. It is not a straightforward phenomenon as it is anticipated by most of the people. It is rather a complex process having dynamic structure which makes it difficult to distinguish between speech signal and its associated undesired noise [6]. Spoken language mainly comprises of sentences which are made up of words. Words consist of letters which produce some specific sounds. These sentences carry some meaning which the speaker tries to convey. When the speech reaches another person, the signal is received by the ear and transmitted to the recipient's brain for processing the message and decoding it. This is a fairly simple linguistic description and everyone can understand the structure. The Figure 1 shows the entire mechanism from speech production to speech perception by the humans. The message is made within the speaker's brain and the human vocal system converts the sequence of words into the speech signal waveform that is transferred via a noisy communication channel. On the listener side, When the speech signal is sensed by the human auditory system the listener's brain starts decoding the speech signal and recognizes the words sequence. [1]

The spoken language is composed of sounds which are defined as **phones**. Most of the languages follow the western style of writing and arrange letters from left to right to form words and eventually sentences. The text written in this pattern is called **transcription**, the term that will be used throughout this thesis. Whereas in order to constitute the speech, phones are arranged in such a manner that they follow

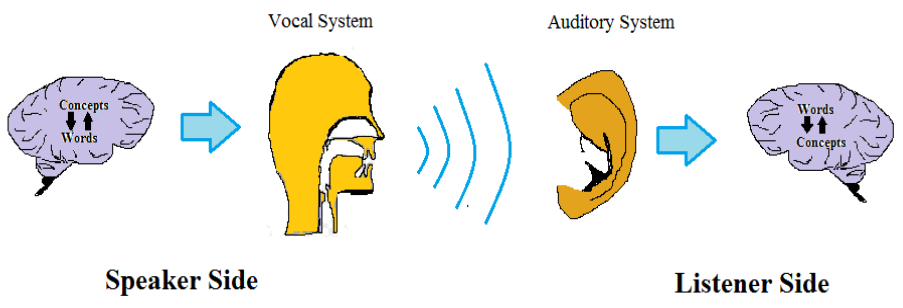

Figure 1: Human Speech Communication [1]

each other in time. This arrangement is universally applicable to all the languages either eastern or western. Strictly speaking, a **phoneme** is used instead of phone which is a more suitable representation of this class of sounds. Phone is merely an acoustic realization of phoneme which is the smallest unit of speech that contains meaning.

Phones are usually unique but in some cases they may vary depending upon the situation they are in. When phones are considered in reference to transition from one to another usually contain more information as compared to mere phones. They are referred to as **diphones**. As [7] explains, American English contains almost 50 phones and 2000 diphones. Those diphones are relatively distinct and occur fewer times than that of phones. Similar conclusions can be drawn for other languages such as Finnish. The context in which the phones occur often affects the meaning and pronunciation of phones, so it can be termed as a **triphone** i.e. a phone considered in a contextual scenario. As a simple example, when o is placed with d and m as its following phones and t as its preceding phone in both cases, `"tod"` and `"tom"` sound significantly different in their respective cases. The in-depth details of how humans produce and perceive speech require a separate discussion and those syntactic and linguistic level technicalities are not included within this thesis.

2.2 Automatic speech recognition

This section introduces the Automatic Speech Recognition (ASR) in detail along with its history and core processes.

2.2.1 History

In its early days, the ASR technology was restricted to government or military use. The field of speech recognition has shown promise and found various exciting applications in the everyday use, since when the first recognizer was built in 1950 [8]. Figure 2 from [2] shows the evolution of speech recognition technologies over the course of past seven decades. The use for automatic speech recognition applications has grown from simple to complex.

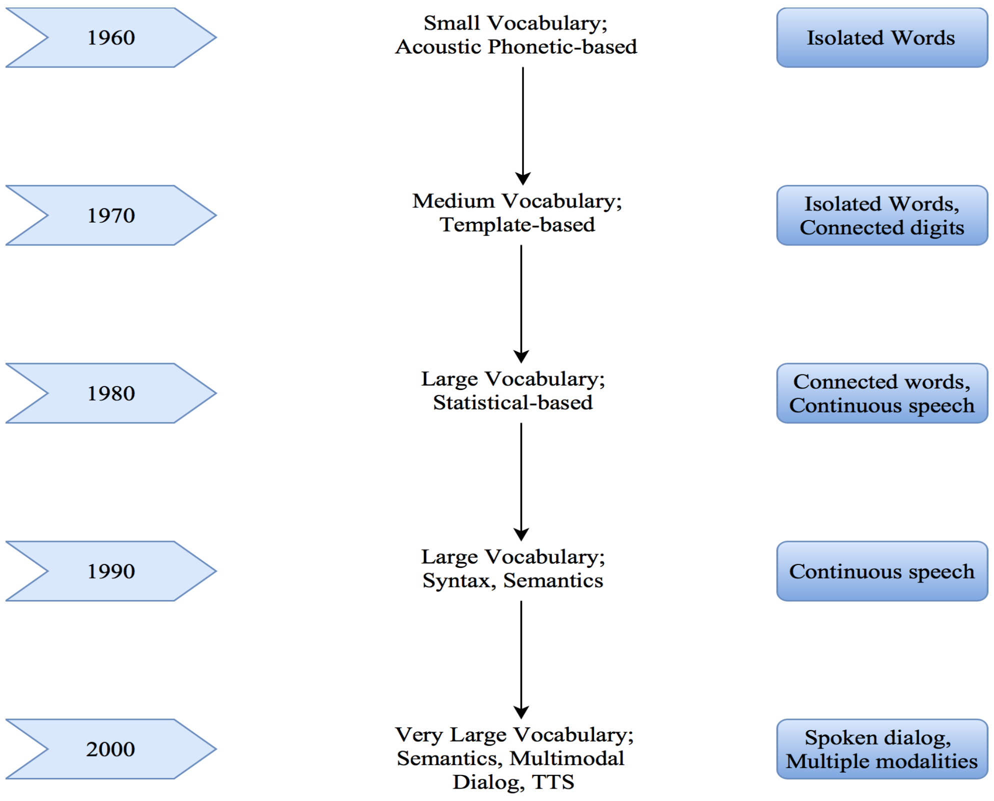
1960
Small Vocabulary; Acoustic Phonetic-based
Isolated Words
1970
Medium Vocabulary; Template-based
Isolated Words, Connected digits
1980
Large Vocabulary; Statistical-based
Connected words, Continuous speech
1990
Large Vocabulary; Syntax, Semantics
Continuous speech
2000
Very Large Vocabulary; Semantics, Multimodal Dialog, TTS
Spoken dialog, Multiple modalities

Figure 2: Milestones in Speech Recognition [2]

Most of the speech recognition systems developed in 1960 were able to recognize small vocabularies i.e. between 10 to 100 isolated words based on phonetic based properties of speech. Technologies developed and implemented during that time include filter-bank analysis, time-normalization methods [2]. One decade later, new systems were developed to recognize medium-sized vocabularies using template-based methods. The technologies used during that time were pattern recognition models and pattern clustering methods for speaker-independent recognizers. In its essence, these methods were very limited due to its inefficiency to handle large-sized vocabulary because building a separate template for each of those words required large number of computations.

Transition of speech recognition systems from the template-based approach to the framework of statistical methods and Hidden Markov models (HMMs) happened in early 1980s. HMM with stochastic language model enabled the speech recognition systems to recognize thousands of connected words. In 1990s, due to the introduction of statistical language models [9] large vocabulary systems were created for the continuous speech recognition. In 2000s, the integration of full semantics model and text-to-speech synthesis system with the very large vocabulary system happened, which enabled the spoken dialog systems with multiple input-output approaches.

Also in recent years, the development of Artificial Neural Networks (ANN) [10, 11] is the most illustrious milestone in this field so far. Initially, it was successfully used for the simpler speech recognition tasks like phoneme classification and isolated word recognition. After that ANNs techniques were successfully used for large-vocabulary continuous speech recognition systems. Currently, a lot of research work is being conducted in this field as it shows really good and promising results.

2.2.2 Overview

In simple words, speech recognition can be described as a process of transcribing the spoken language into the textual form by recognition system. In scientific terms, it is not a simple task. A speech recognition system consists of a number of building blocks which are a fruit of decades of research and are necessary for its proper functioning.

The schematic diagram of a typical ASR system is given in Figure 3. This conventional structure of an ASR includes two statistical models i.e. acoustic model and language model. *Acoustic model* provides the output on the basis of acoustical properties of the input speech signal. The speech signal is not directly fed to the acoustic model. Its useful and relevant characteristics are extracted using different feature extracting techniques and provided to the acoustic model for scoring. *Language model* has all the information about the legitimate and allowed words and provides the probability of words occurring together. *Lexicon* provides the phonetic-level pronunciation of all words present in the dictionary. *Decoder* utilizes the probabilities provided by the models and gives the transcription for the speech signal.

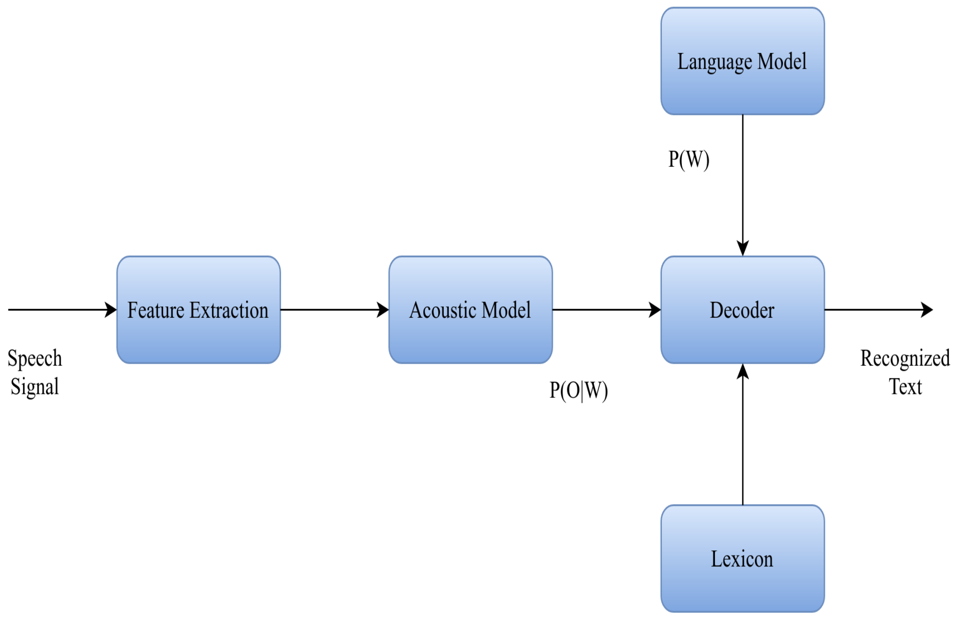

Figure 3: Schematic diagram of an ASR system

The essence of the problem for the system in statistical terms is finding the most likely word sequence W when a system has some observations O obtained from the given speech signal. This is an optimization problem and Bayes' formula can be used for solving this problem [12];

$$\hat{W} = \arg\max_{W} P(W|O) \tag{2.1}$$

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

Here *P(O)* is the observation probability, which is equal for all the hypotheses and therefore, can be omitted in the search.

$$\hat{W} = \arg\max_{W} P(O|W)P(W) \tag{2.3}$$

whereas, $P(O|W)$ is given by the acoustic model and $P(W)$ by language model. To obtain these, the task become of finding the optimal models that provide the best probabilities and hypotheses. These will be discussed in a rather detailed manner in the sections that follow.

2.2.3 Speech parametrization

When a human listens to someone's speech, he quickly processes the whole signal and respond to it appropriately. It all happens so quickly that one can hardly ever notice the functioning that human brain has to go through. In short, after hearing the speech human brain searches through all the content, discards the unnecessary information and only processes the information that is useful. As the next step, it generates a suitable response and vocal tract converts that response into speech that reaches the ear of another human. When this generated speech reaches the listener's brain, the same process starts all over again and conversation continue to happen at a remarkably fast pace in even in the most ordinary of situations. The process of searching the signal is only concentrated on the desired information by ignoring the irrelevant information and this all happens naturally in brain. But when it comes to a system, the same does not hold true and some methods have to be incorporated which focus only on the essence of the speech and discard the irrelevant parts of signal. This set of methods is known as **feature extraction** and is the counterpart of what human brain does to almost all of the signals it processes.

Feature extraction can be regarded as a pre-processing step to the actual speech recognition task. It aims to compress the audio signal data into a less heavy and processable signal that is concise and easy to process. This signal consist of the features that preserve the original signal information and do not contain any noise and background interferences. Speech is a non-stationary signal [13, 14] since it encompasses a variety of factors that affect its consistency. On a very minute level, speech can be treated as a stationary signal for a very short period of time e.g. 10 ms. These short-time windows are converted to their respective spectrum for further processing since time domain is not a suitable platform for processing. Frequency domain, on the other hand maps different parts of speech into separate spectral components known as **formants** whose spectrogram is clearly visible to human eye as well, whereas time domain signal is complex because words are indistinguishably merged with each other in the waveform.

One of the most popular features is Mel-frequency Cepstral coefficient (MFCC) [15, 16, 17]which is obtained by processing the transformed signal with Mel filter-bank. These features are very significant since they are obtained by processing the signal on a scale that considers the frequencies that are most important to humans [18]. Afterwards, compression and a Discrete Cosine transform (DCT) is applied to the obtained signal and resultant signal is decorrelated. The extraction of MFCC features is shown in the Figure 4.

The high-frequency bands of speech signal suppresses during the process of human sound production [18]. Pre-emphasis technique is used to compensate these bands and flatten the speech spectrum. Pre-emphasis filter boosts the speech signal spectrum by 20 dB per decade [19]. In order to achieve this, signal is passed through a Finite Impulse Response filter as described below by the equation 2.4, where the range of values for a_{pre} is $[-1.0, -0.4]$ taken from [19].

$$H_{pre}(z) = 1 + a_{pre}(k)z^{-k} \tag{2.4}$$

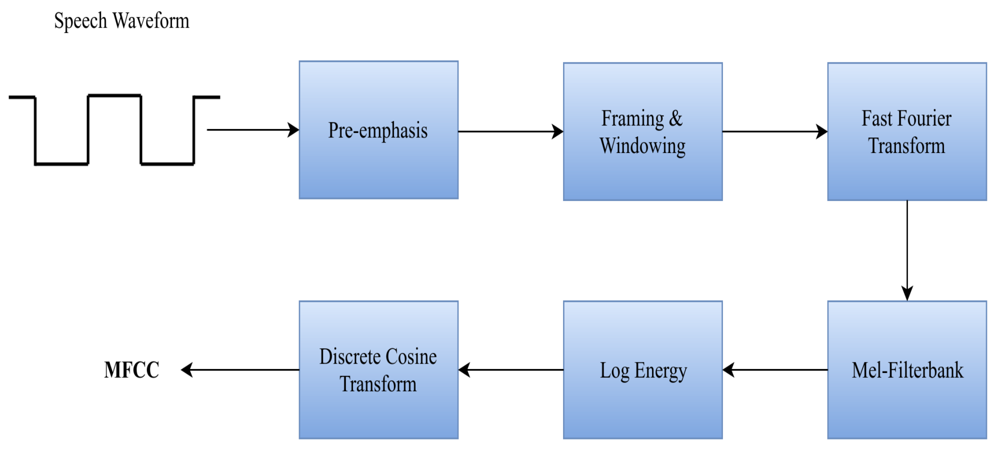

Figure 4: MFCC feature extraction process

In order to process the speech signal, it is important that it should be stationary instead of slow-time varying signal. So the stationary nature can be achieved by dividing the speech signal into sequence of frames, where each frame can be treated as an independent feature vector with stationary properties. Usually, the frame size is kept at $25ms$ with a frame shift of $10ms$. Windowing of each frame is required in order to reduce or eliminate the discontinuities present at the edges. If this process is not accomplished than these irregularities can be observed as high frequency component in the Fourier transform [20]. The most commonly used window is the Hamming window which is described in equation 2.5 below taken from [21].

$$w(n) = 0.54 - 0.46 * cos\frac{2\Pi n}{N}; \quad n = 0, 1, ..., N-1 \tag{2.5}$$

After the windowing process, the samples from each frame are converted from the time to the frequency domain. This conversion is achieved by using the Fast fourier transform (FFT). The next step is to bend or warp the speech spectrum with the Mel-Frequency scale, so that the frequency resolution can be adapted to the human ear properties [22]. This process relates to the Mel-Filterbanks stage where these banks are formed on a Mel-Frequency scale. These Mel-Filterbanks are set of overlapping triangular bandpass filters which get multiply with the spectrum, resulting in the amount of energy for each filter bank and the coefficients of each filter summed up [23]. Also in order to get the loudness realization of the human auditory system, logarithmic energy is calculated from the time domain which represents the energy of each frame. The final step is to apply the discrete cosine transformation (DCT) to the log-spectral energy and obtain the desired output i.e. Mel Frequency Cepstral Coefficients.

2.2.4 Acoustic Modelling

Acoustic model is the next building block in automatic speech recognition systems. $P(O|W)$ illustrates the acoustic model equation 2.2. Acoustic model is used to model the acoustic features given the set of underlying words. More precisely this model defines a statistical framework for the feature vector sequence. Speech is a highly complex signal and individual classes of sound or their corresponding feature vectors are highly correlated with each other, thus cannot be treated as isolated units. For this reason, acoustic model should be able to account for these temporal dependencies. In order to handle these dependencies, Hidden Markov Model (HMM) has been a popular choice in speech recognition community.

Generally, an acoustic model is estimated by speech utterances and their respective transcriptions. This training is discussed in more detail in Chapter 4. The acoustic model is trained by using a set of parametric word models with parameters θ. It is assumed that the observation vector sequence corresponding to a word is

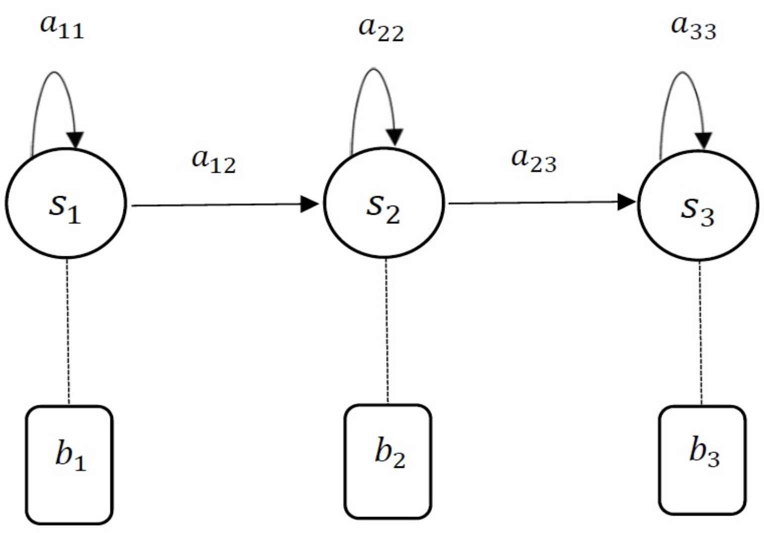

Figure 5: Inter-state transition diagram of a typical Hidden Markov model

possibly generated by a Markov chain or Markov model.

A Markov chain assumes Markov property and is used to model a random variable sequence. This sequence is generally represented by discrete time states. The Markov property confirms that the next state's distribution only depends upon current state. At time step t, it can jump from one state to another in two successive states, e.g. current state i and next state j, and **transition probabilities** a_{ij} govern these inter-state transitions. **Emission distributions** $b_i(\mathbf{x}_t)$ generate the output symbol $\mathbf{x}_t$. These symbols are usually vectors consisting of continuous variables. Provided a sequence of T states q_1^T, HMM defines a distribution over a respective observation vector sequence x_1^T. Emission distributions $b_q(\mathbf{x})$ and transition probabilities a_{ij} affect this distribution as

$$p(x_1^T|q_1^T) = \prod_{t=0}^{T-1} a_{q_t q_{t+1}} b_{q_t}(\mathbf{x}_t) \tag{2.6}$$

The state sequence q responsible for generating the output sequence is unknown which provides the notation *Hidden* in the Hidden Markov model. A Hidden Markov model mainly comprises of a discrete 1^{st} order Markov chain, different states connected using transition probabilities and their corresponding emission distributions. Figure 5 represents a typical left-to-right HMM with its states and their respective associated probabilities which is commonly used for acoustic modelling. When the acoustic model is trained, the main aim of the training process is to obtain the probabilities a_{ij} and distributions b_i that maximize the likelihood $p(\mathbf{x}|q)$ for training corpus. This problem is solved by using Baum-Welch algorithm [24]. The emission distributions are modelled using Gaussian Mixture Models (GMM). The parameter that encompasses probabilities a_{ij} and GMM parameters should be selected so as to maximize the likelihood of training data given the model. This estimation is termed as Maximum Likelihood estimation (MLE). Baum-Welch algorithm determines these parameters. Artificial Neural Networks (ANN) have become a more recent and advanced method for acoustic modelling [25]. For a given observation x_t, deep neural networks are trained to estimate the posterior $p(x_t|q_t)$ for the context-dependent state q_t of HMM and the probability $p(q_t|x_t)$ is derived from this estimation.

2.2.5 Language Modelling

According to the Figure 3 , language models are the next building block of a typical ASR system. The mere acoustic properties of the signal are insufficient and there is a need to incorporate the knowledge of language into the recognizer. Acoustic models cover the language aspect to some extent but recognizer needs deeper knowledge about the language in question. Since 1980s, mainly statistical language models have been used [9]. Their main purpose is to enhance the performance of various speech recognition applications [26], but in general speech recognition cases, the language models provide the system with the knowledge of linguistic structure. Statistical language modelling creates the models that estimate the prior probabilities associated with word strings. In particular, speech recognition employs these models in order to obtain the best estimate of the word sequence probability $P(W)$.

Table 1: Probabilities of different combinations in a trigram

sulku	auki	ota	0.153434
kerro	laatikko	ota	0.3424913
kerro	laatikko	sulku	0.8336139
työ	valmis	sulku	0.905149

This estimate is calculated based on the knowledge of the word frequency and context statistics and system is able to determine which word is more probable to appear after a particular sequence. Context is a very important element to consider, since many word combinations may acoustically exist but the grammatical structure of the language may not allow all of those e.g. `"it is reigning cats and dogs"` is likely to appear in acoustic model output but the only allowed combination is `"it is raining cats and dogs"`. This means that the language model should prevent the incorrect combinations from being hypothesized using the knowledge of grammar. The confusion occurs when the vocabulary size is large and numerous combinations of different words are possible. The task of language model here is to prioritize those and assign highest probabilities to those combinations which are most likely to occur. Table 1 explains this case, as it assigns the highest probabilities to different sequences in descending order. Language model also assigns some small probabilities as a result of smoothing, to those word sequences that are unlikely to occur or do not exist in the language. Using basic rules of probability, language model's word probability can be sequentially decomposed [27] in formal mathematical terms as

$$P(W) = \prod_{i=1}^{N} P(w_i|w_1, w_2, ..., w_{i-1}) \tag{2.7}$$

The aim of the model is to provide the word sequence probability that maximizes the equation 2.2. The expression in equation 2.7 symbolizes the N-gram language models which consider the N-1 previous words. Using this model, one can calculate the ML estimates of different N-grams [28]

$$P_{ML}(w_n|w_1, ...w_{n-1}) = \frac{C(w_1, w_2, ..., w_n)}{C(w_1, w_2, ..., w_{n-1})} \tag{2.8}$$

For a trigram model equation 2.8 reduces to

$$P_{ML}(w_i|w_{i-2}, w_{i-1}) = \frac{C(w_{i-2}, w_{i-1}, w_i)}{C(w_{i-2}, w_{i-1})} \tag{2.9}$$

The maximum likelihood estimate in the equation 2.8 above assigns zero probabilities to the sequences that do not exist in the training data. This leads to problems in speech recognition process, for the word sequences with zero probability cannot be recognized in this case. Different smoothing techniques are are used for assigning some probability to these unseen N-grams [29] but are out of the scope of this thesis.

It may seem a good idea to estimate higher N-grams e.g. five-gram, since there are quite many words in some sequences which influence immediate neighbouring as well as distant words. As a quick example, consider a phrase `"for the night is dark and full of --"`. The word `"night"` is pivotal here and affects the choice of word at the ending of the sequence and one of its likely continuation is `"terror"`, even though `"research articles"`, `"animals"` or `"ideas"` are likely to appear after `"full of"`. If this contextual effect of words on distant words is considered, a price has to be paid which is the complexity of N-grams for increasing values of N.

2.2.6 Decoding

The most important block of an ASR system is the decoder. This part is an actual responsible for recognizing or transcribing the input speech. It takes the input probabilities from both acoustic and language model and combine them to get the most probable word sequence as described in equation 2.3. When the number of words increase in the speech utterance, the search space of the decoder for the possible word sequences grows exponentially [30]. The decoder should provide the best search path and the search should be efficient. So this can be achieved by restricting the search to the most probable and likely path by using the pruning, which eliminates the unlikely hypotheses. Most of the decoders use Viterbi algorithm for the searches [30, 31]. In reality, the decoder carries multiple partial search hypotheses where each of these hypothesis is considered as a token. These tokens are propagated in the transition network structure or a graph with the permitted word sequences along with corresponding HMM sequences [32].

2.2.7 Evaluation of ASR performance

A common method to assess the performance of speech recognition compares the transcription generated by the recognizer to the original one which is known at the beginning of the process and to calculate the word error rate (WER). Assume that the original evaluation transcription consists of W_r words in total. The number of substitutions W_S, insertions W_I and deletions W_D are taken into the account, which are performed by the recognizer to calculate the output hypothesis. These counts are used to compute the most commonly used performance evaluation metric, referred to as **word error rate** (WER) [33, 34, 35], given as

$$WER = \frac{W_S + W_D + W_I}{W_r} * 100\% \tag{2.10}$$

This metric is widely used in speech systems community and this thesis also uses this measure for performance evaluation of the recognition or training tasks. The smaller the WER, better is the accuracy of the task. In theory, WER can be higher than 100% when the number of erroneous words combined is larger than the total number of words in original transcription. In order to understand, consider the following example which is of a shorter utterance but a compound word.

```
ref: geopolitical  ******    **** website
hyp: geo           political web  site
eval:S             I         I    S
```

For these two compound words forming a short utterance, if recognizer somehow recognizes the above mentioned hypothesis, the output WER is

$$\frac{2+0+2}{2} * 100\% = 200\% \tag{2.11}$$

This may seem unfair to obtain an error rate of 200%, but this is the principle of WER. In this case, a more suitable choice of performance criterion is **letter error rate** (LER). This is the same method of calculation, but on the letters instead of whole words.

```
ref: g e o  *  p o l i t i c a l   w e b  * s i t e
hyp: g e o  -  p o l i t i c a l   w e b  - s i t e
eval:       I                             I
```

For the same recognition results, the LER is

$$\frac{0+0+2}{19} * 100\% = 10.5\% \tag{2.12}$$

Although LER might seem more effective than WER in this case, unfortunately it does not hold true for entire languages especially when simple mappings from phones to letters or vice versa, do not exist.

Neither of WER and LER is an absolutely perfect measure to evaluate the performance. Both of these are not descriptive enough to estimate what exactly was wrong in the recognition. For WER, a mis-recognized letter may yield an incorrect word and consequently higher error rate, but the correctly recognized words will be error free. For LER, a slight change in letter recognition might not increase the overall error rate, but the meaning of the output word changes altogether. Grammatical structure of the concerned language and compound nature of words can help to decide which metric to choose for the performance evaluation of the system. For example, Finnish language contains long compound words, which means that a minor mistake in letter recognition might produce higher WER but less LER, whereas the case is different for English language. In some cases, both of these are considered and their relative improvement can also be taken into account. In this thesis, only WER is considered to evaluate the speech recognition performance.

2.3 CMU-Sphinx toolkit

CMU-Sphinx is a famous toolkit for speech recognition purposes developed at Carnegie Mellon University. This toolkit includes number of speech recognizers i.e. Sphinx-3, Pocketsphinx, Sphinx-4 and acoustic model trainer i.e. Sphinxtrain. According to [36] the CMU-Sphinx toolkit has different packages and which one to use depends on the nature of the application.

- *Sphinxtrain*: Acoustic model training tool developed in C language.
- *Pocketsphinx*: Lightweight speech recognizer with all libraries written in C language.
- *Sphinxbase*: Provides libraries to Pocketsphinx and Sphinxtrain written in C language.
- *Sphinx-4*: It is modern speech recognizer build completely in Java.

Chapter 3

3 Small Vocabulary Automatic Speech Recognition System

The three main objectives of this thesis as discussed in Chapter 1 are, to create an acoustic model, modify the voice activity detection and create the garbage model for out-of-vocabulary words. These tasks are done for a company having off-line pocketsphinx speech recognition engine for the embedded devices i.e. smart phones and tablets. Since the speech recognition process needs to be done on a mobile device, so the vocabulary has to be limited. In other words, it can be said that the system is a small-vocabulary-speech-recognition (SVSR) system. Figure 6 shows the

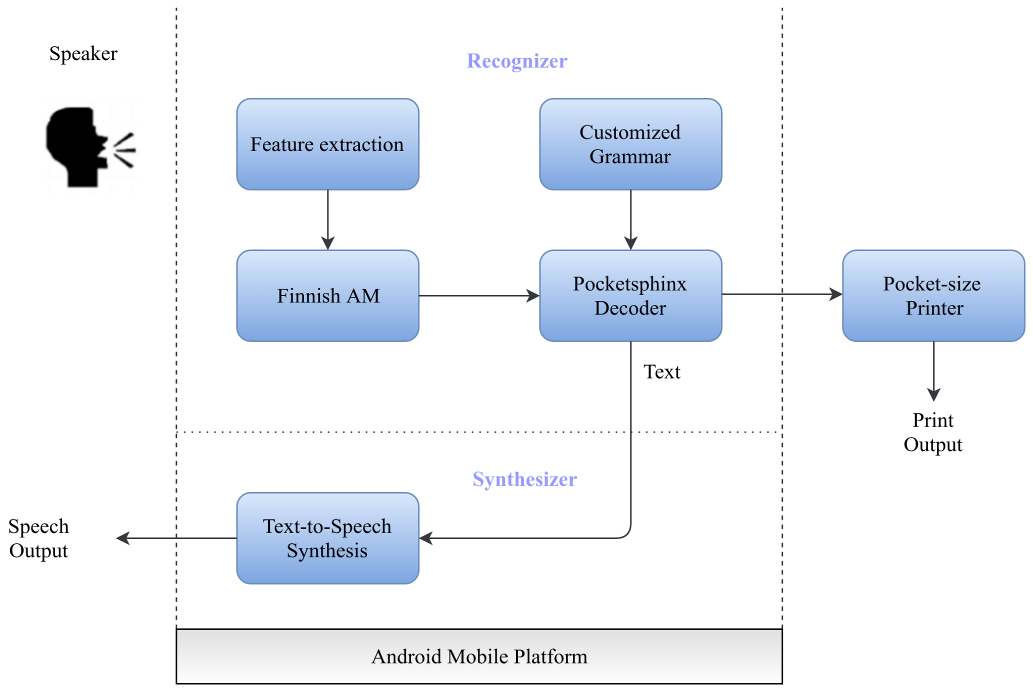

Figure 6: System architecture of TalknLabel

system architecture of the product *TalknLabel* of the company called Devoca having small vocabulary of 100 - 150 words including digits, names and special characters. The two essential parts of this product are *recognizer* and *synthesizer*. For the other products of the company, they are not using the speech synthesizer module but the recognition techniques will remain the same. *Pocketsphinx* speech engine has two recognition modes and it also has the ability to switches between them. One of the mode only operates for recognizing the special keywords its also called partial recognition. Second mode is known as full recognition mode and able to recognize all the words included in the grammar. Once the speech is recognized by the decoder, then the recognition is stopped and TTS module will play the audio to output the result. This is necessary as currently, there is no mechanism implemented to separate the TTS audio from recording audio. Once the TTS activity is over, the recognition process resumes from its last stage. As shown in the Figure 6 recognition engine is working on an android platform, so the good recognition speed is desired. This means that apart from the correctness of text result, the time taken by the speech recognizer to recognize and deliver the output should be acceptable by the end-user.

Chapter 4

4 Acoustic Model Training

4.1 Introduction

Acoustic model is a mathematical model which consists of the statistical properties of speech that create individual words. Acoustic models are trained with the help of efficient training algorithms and techniques which require diversified training data to create the statistical representations for each sound unit (i.e. *phoneme*) present in language and also to obtain the best solution for the model parameters. These statistical models are defined with the help of Hidden-Markov-Models (HMMs) [37] as explained in section 2.2.4 of Chapter 2. In order to train the noise-robust acoustic model for a particular language, large and dynamic speech corpus is required. Speech corpus consists of audio recordings of different speakers and their respective detailed transcriptions.

As mentioned in section 2.1 of Chapter 2, speech has a lot of variations in itself but to train the better acoustic model besides speech, the environmental conditions also need to be taken into account while recording the speech audio. The reason behind incorporating the environment variations is to keep the training data more closer to the real-world data for better speech recognition accuracy. CMUSphinx provides number of good-quality acoustic models and they can be downloaded from the CMUSphinx's subversion repository [38] for free. Most of the small and large vocabulary applications can use these already available acoustic models, but there are some applications where these models cannot be used if a specific model is required for the small vocabulary application to match the recognition conditions. New spoken language can also be an issue i.e. application demanding Finnish AM, can also leads to train the new acoustic model. An acoustic model can be trained easily by using the CMUSphinx's trainer i.e. *Sphinxtrain*, provided that the plenty of training data is available.

4.1.1 Model types

During the training of an acoustic model, mixture of Gaussians are used to estimate the score of each frame extracted from the speech signal. CMU-Sphinx supports three different types of parameters tying between Gaussian mixture models and consequently leads to three different types of the acoustic models namely Continuous model, Semi-continuous model and Phonetically tied mixture model (PTM) [39, 40, 41, 42]. Semi-continuous model works best when the amount of training data is limited. It can be considered as a special type of continuous model with tied Gaussian mixtures having reduced number of parameters and the computational complexity. The number of Gaussians in the continuous model are approximately $150 * 10^3$ and frame can be estimated by using the mixture of these Gaussians, whereas the number of Gaussians in semi-continuous models are just 700. Similarly, the PTM model uses $5 * 10^3$ Gaussians.

Acoustic model behaviour in terms of recognition accuracy and processing time can be explained with the above mentioned Gaussian mixtures. Continuous model is more accurate and precise than the other remaining models, since each senone in this model has it's own set of Gaussians [43]. Due to the large number of mixtures, the computational complexity to estimate the score for each frame is quite high in continuous model and for this very reason, this model cannot be used in mobile applications. On the other hand, semi-continuous models are very fast because of lesser Gaussian mixtures and light weight due to the data compression, which of-course affects its recognition accuracy. PTM model provides the golden middle between these two extremes which means it is considerably faster than continuous model and provides better recognition accuracy than semi-continuous model. PTM model can also be used in mobile applications.

4.1.2 Model format

Acoustic models build from *Sphinxtrain* is basically a folder having different files, which are listed below.

- *feat.params*: This file defines the different parameters for static and dynamic features extraction. It also provide the information about the model type along with other information.
- *mdef*: This is a model definition file and its function is to provide a particular numerical identity to each state of all HMMs, which are going to be trained.
- *means*: This file defines the means of all Gaussians in an ascii format.
- *variances*: This file defines the variances of all Gaussians in an ascii format.
- *noisedict*: This is a dictionary file having all kinds of noises or filler words which are there in the speech corpora.

- *mixture_weights*: This file contains the weights provided to every Gaussian mixture for all the senones present in the model.
- *transition_matrices*: This file contains the information related to each HMM state transition topologies and probabilities in model.
- *sendump*: This file contains the compressed and quantized Gaussian mixture weights for all the senones present in the model. This file can be used if *mixture_weights* file is not available.

4.2 Speech database

The Devocan database is comprised of Finnish audio recordings and their transcriptions. The provided data was collected from different speakers including both males and females during the data collection campaign conducted by the **Devoca Oy**. Native speakers as well as non-native Finnish speakers participated in that campaign. During the data collection process more than 150 Finnish speakers took the part and each speaker had to read the provided text for almost 10 minutes. Each utterance had been stored with the sampling rate of 16 KHz, having Bit depth of 16 bits along with the Microsoft waveform file format (***MS WAV***).

4.2.1 Database division

Database is divided into three datasets namely “Training”, “Evaluation” and “Testing” datasets. The training dataset is comprised of recordings obtained from **143** speakers. Table 2 explicitly shows the composition of training dataset.

Table 2: Training Dataset

Gender	**Native Speakers**	**Non-Native Speakers**
Male	77	8
Female	50	8

The training dataset consists of approximately 24 hours of speech data. The evaluation dataset contains the recordings obtained from **10** speakers, whose speech data is not included in the training dataset. This data contains almost 90 minutes of speech. This dataset is prepared to evaluate the performance of the Finnish acoustic model built from the training dataset. In other words, this data helps to understand that how better is the acoustic model working for the speakers not present in the training data. In total, there are **577** utterances in the evaluation set containing 6289 words. Table 3 shows the statistics of evaluation set in terms of male and female speakers.

Another dataset called testing set is created by selecting 5 utterances from each speaker included in the training database. These selected recordings are not included

Table 3: Evaluation Dataset

Gender	**Native Speakers**	**Non-Native Speakers**
Male	4	1
Female	3	2

in the training database. So, the testing dataset contains **828** utterances having 10100 words. This set contains 2 hours of speech.

4.2.2 Aligning speech data

The actual speech data and their respective transcriptions provided by Devoca was in the raw form and cannot be used directly to build and evaluate the acoustic model. So, the large speech recordings were cut into thousands of small recordings by using **Sphinx-4 Aligner** and bash scripting. It is a known fact that during the recordings some variations may arise in the speech data in the form of coughing, fumbling, repetition of words and non-words. These factors are not handled in the transcriptions initially, which cause the difference between recorded speech data and the original transcription. These factors can generate the errors during the acoustic model training process. In order to remove such erroneous points from the transcriptions, Sphinx-4 aligner is used which specifies the time (both starting and ending time) for each words from the transcriptions against the speech audio. It also has the ability to insert or remove a word in the transcribed output. The aligned output of some random utterance by Sphinx-4 aligner is shown below.

```
   seuraava [17380:18030]
   kahdeksantoista [18660:19500]
   tuote [19510:20050]
-  soita
+  toista [20090:20650]
```

The *start* and *end* time of a word can be seen in the square bracket separated by colon. The **-** sign shows that this word **soita** is existing in the transcript but not present in the speech file. Similarly, the + sign shows that the word **toista** is not in the transcript but present in the speech file and its start and end time is also provided by the aligner.

Once the aligned output is obtained, now the bash scripting can be done to split the long audio recording into many small audio clips, where each audio clip can have atleast 5 words. Besides splitting the long audio, bash script can also generate the correct transcription files and user ID files for small audio clips based on the aligner output.

4.2.3 Transcriptions and FileIds

Transcription file is just a simple text file with contains the transcription of each audio file included in a dataset. Each dataset has its own transcription file. The format of transcription file is given below.

```
<s> yksi nolla kaksi kolme </s> (utt1)
<s> viisitoista kaksitoista kolmetoista </s> (utt2)
```

It can seen that each line starts with <s>and ends with </s>, with the utterance file name in parenthesis. File name should be provided without any extension and folder path. Similarly, Fileids is also a text file which contains the folder path of the recordings. The format of Filedis is given below.

```
speaker_1/utt1
speaker_2/utt2
```

4.3 Training

As discussed in section 4.1, the core purpose for training an acoustic model is to find the most optimal parameters for HMMs. Figure 7 from [44] shows the overall generic flow for training an acoustic model. The training process for an acoustic model essentially means that four important parameters namely, transition probabilities, emission probabilities, initial probabilities and total number of HMM states have to be computed. These parameters are represented by the set (A, B, Π, Q) [45]. Q represents the total number of HMM states in the model which is set by hand in the beginning of the process and remains unchanged during this process.

The first phase in training is the estimation of features (mfc) file from the available speech recordings and their transcriptions. Then, as the next step is based on the initially assumed probabilities, the hidden Markov models are initialized to start the procedure. The actual estimation is focused on the parameter set $\Lambda = (A, B)$ [45]. The most commonly used method for calculating these parameters is maximum likelihood estimation (MLE). This method consists of maximization of the likelihood function $L(\Lambda|X)$ as follows;

$$\Lambda' = \arg\max_{\Lambda} L(\Lambda|X) = \arg\max_{\Lambda} \sum_{Y\in\Theta} \prod_{t=1}^{m} b_{y(t)}(\mathbf{x}_t) a_{y(t),y(t+1)} \tag{4.1}$$

Here X represents the training corpora as a sequence of observation vectors such that $X = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_m)$, Λ and Λ' the initial and updated parameter sets respectively, and Θ the set of all allowed HMM paths in the acoustic model. Y is the sequence of states for all HMM states associated with the set Θ.

In essence, the training corpora X symbolize the actual transcription utterances for the training speech data. The parameter set Λ is roughly initialized in the beginning of the training process and the re-estimation of this set takes place in such a manner as to exhibit consistency with the maximization solution presented in the equation 4.1. Using Baum-Welch algorithm, the transition probabilities are updated in the maximization step as:

$$\hat{a}_{i,j} = \frac{\sum_r \frac{1}{L_r} \sum_{t=1}^{M_r-1} \alpha_i^r(t) a_{i,j} b_j(\mathbf{x}_{t+1}^r) \beta_j^r(t+1)}{\sum_r \frac{1}{L_r} \sum_{t=1}^{T_r-1} \alpha_i^r(t) a_{i,j} \beta_j^r(t)} \tag{4.2}$$

whereas $a_{i,j}$ and b_j are initializing parameters before re-estimation and are used in forward-backward process in BW algorithm. For any given HMM state, α and β represent the forward and backward probabilities respectively. Then the next step of the process is to compute the emission probabilities for a particular GMM which consist of estimating the set $(\omega, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ [45]. Using BW algorithm for maximum likelihood estimation yields:

$$\omega_{q,i} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_{q,i}^r(t)}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_q^r(t)} \tag{4.3}$$

$$\boldsymbol{\mu}'_{q,i} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{q,i}^r(t) \mathbf{x}_t^r}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_{q,i}^r(t)} \tag{4.4}$$

$$\boldsymbol{\Sigma}'_{q,i} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_{q,i}^r(t) (\mathbf{x}_t^r - \boldsymbol{\mu}'_{q,i})(\mathbf{x}_t^r - \boldsymbol{\mu}'_{q,i})^T}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} \gamma_{q,i}^r(t)} \tag{4.5}$$

Here, R is the total number of observation sequences and t is the time of the respective observation sequence, hence all the paths start from $t = 0$ and end at $t = R$. The state q belongs to the set Q which defines the number of all the states and i is the instantaneous component of the Gaussian mixture. $\gamma_{q,i}^r(t)$ is the state occupancy of state q for the ith component of Gaussian mixture at time t for rth observation sequence. These parameters (A, B) are repeatedly re-estimated until the function $L(\Lambda|X)$ reaches the point where it converges to its local maximum and are maximized as governed by equation 4.1.

Training of an acoustic model by using Sphinxtrain is quite an easy task provided that all the data is available. But the procedures actually happening behind the training process are very complex. Several modules are executed during the training course [46]. Module **000** and **00** compute the feature files for all the audio files present in the training data. They also check the duplication of words in the dictionary and make sure that all the words present in the transcription file have

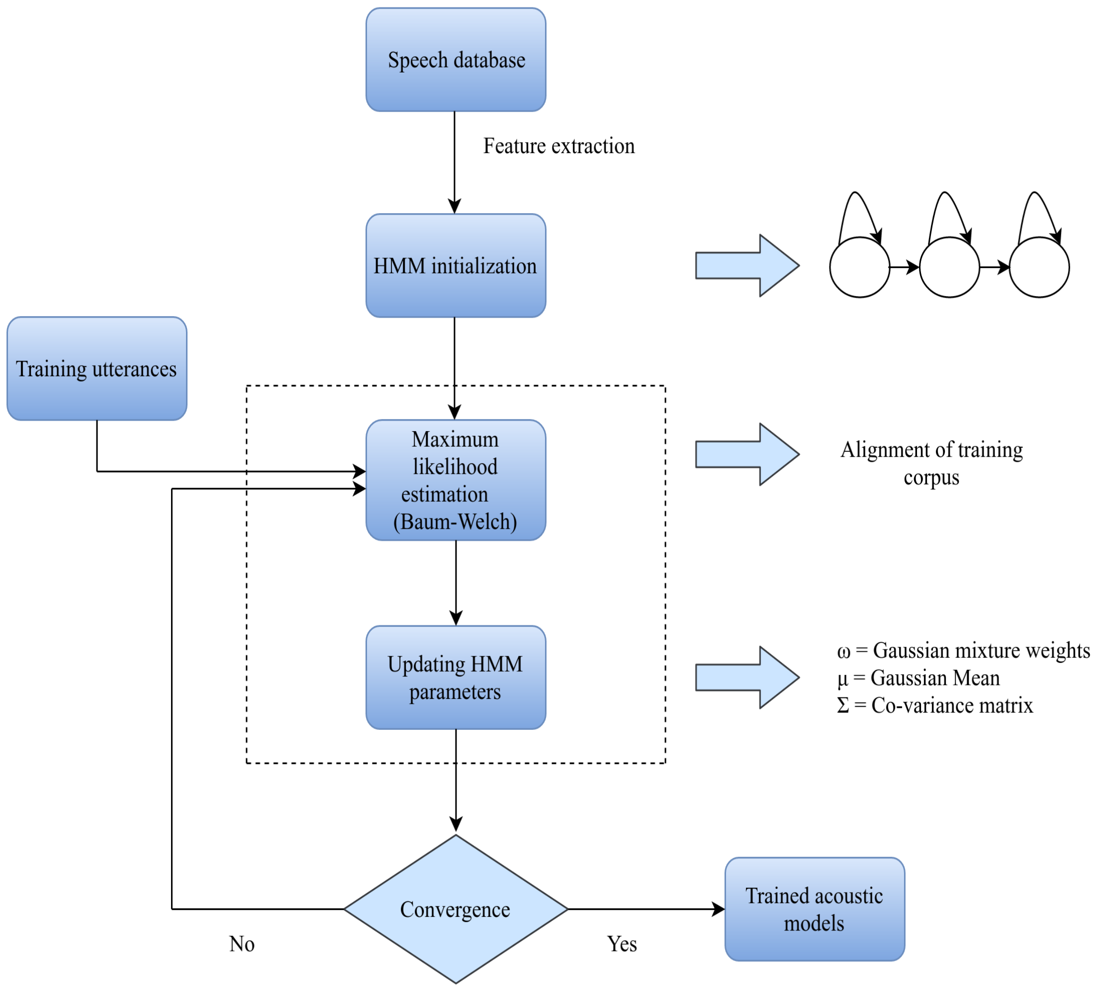

Figure 7: Block diagram of AM Training

phonetic representation in the dictionary. Module **05.vector_quantize** generates the common model for acoustic features without differentiating the phones. In module **10.falign_ci_hmm** and **11.force_align**, forced alignment algorithms and techniques are applied to train the context independent models. The next module is **20.ci_hmm** which train the context independent models i.e. all single phones are trained without reference to their context. In module **30.cd_hmm_untied**, models are trained for each triphone. This means that the phones are now trained with reference to their context. The next modules are **40.buildtrees** and **45.prunetree**, where decision trees are used to decode the HMM states of all triphones and to cluster the triphones with similar states. After the creation of decision trees, these are pruned to create the global acoustic units known as senones. In the last module **50.cd_hmm_tied** senone or tied state models are trained to generate the final version of the acoustic model.

4.4 Experiments and Results

Two different types of acoustic models are trained from the Devocan database namely PTM and Semi-continuous model. As discussed in 4.2.1, evaluation and testing datasets are used to evaluate the accuracy of the two newly trained acoustic models. Table 4 shows the word error rates (WERs) of these datasets, when tested with the new models having 3k senones. So, by looking at the **WERs** it can be said that PTM model is having better recognition accuracy than semi-continuous model. The table is showing the averaged WERs of all speakers from the evaluation and testing datasets.

Table 4: WER of PTM & Semi-Cont. Models

Datasets	**PTM Model (WER)**	**Semi-Continuous Model (WER)**
Evaluation data	4.34%	4.87%
Testing data	4.3%	4.9%

It is interesting to see the recognition accuracy (in terms of WERs) of the acoustic model for males and females separately. Table 5 and 6 shows such results.

Table 5: WER for Male speakers

Evaluation data	**PTM Model (WER)**	**Semi-Continuous Model (WER)**
5 male speakers	4.55%	5.27%

Table 6: WER for Female speakers

Evaluation data	**PTM Model (WER)**	**Semi-Continuous Model (WER)**
5 female speakers	4.14%	4.51%

The WER for female dataset is better than male evaluation dataset. The reason for that is the speech data of females included in the evaluation dataset is recorded in slightly less noisy environment than that of males. Figure 8 shows the WERs for male and female speakers individually. From the bar graph it can be seen that the two male speakers have WER roughly 11% which explains the high background noises. Therefore, the averaged WER for male speakers is higher than female speakers.

The new acoustic models i.e. PTM and Semi-Continuous model, work good when evaluated against the test datasets. However, when checked in the real-word testing PTM model is quite slow in terms of giving the output text as compared to the semi-continuous model. Since these acoustic models are used in the mobile

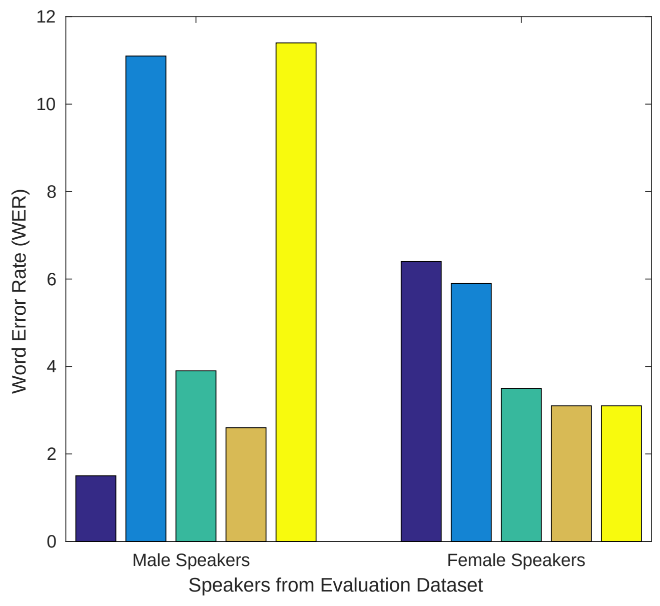

Figure 8: Word Error Rate of Speakers

applications, where the memory resources are limited, so the PTM model is not a good option as it contains 5000 Gaussians which add extra computational complexity during the decoding process. So due to the recognition speed issue, semi-continuous model is considered for the Devoca's mobile applications namely *Talk'n'Label* and *Talk'n'Pick*.

4.4.1 Model comparison

The word error rates for the new semi-continuous model are almost as low as PTM's, when evaluated against the different sets of test data. There is also a need to compare the performance of this new acoustic model with the Devoca's old acoustic model in terms of recognition accuracy. The old acoustic model of the Devoca is semi-continuous model with 4k senones. Table 7 shows the results, which are obtained by using the same test datasets with the old acoustic model. It shows that the recognition accuracy of old acoustic model is not good as compared to the newly trained semi-continuous model.

Table 7: WER using Old acoustic model

Datasets	**Old Semi-Continuous Model (WER)**
Evaluation data	10.94%
Testing data	13%

Chapter 5

5 Voice Activity Detection

5.1 Introduction

Voice activity detection (VAD) is an essential step in various speech related applications like speech coding, speaker recognition and automatic speech recognition which can significantly improve the performance of these applications. In simple words, it is defined as the problem of distinguishing speech part from the input signal which can consists of noise, speech and non-speech segments. The non-speech segments consist of coughs, sneezes and breathe sounds. The basic functionality of VAD is to remove the signal segments which contain nothing but noise and non-speech and this activity of removing signal is known as *frame dropping* in the context of automatic speech recognition [45].

The classification of speech/non-speech becomes an issue when the background noise of the working environment increases. Many VAD algorithms are proposed and implemented so far by the researchers to detect speech from a noisy signal. Figure 9 shows the block diagram of a VAD system. Firstly, the speech signal is divided in to sequence of frames where each frame having duration of 20 to 40 ms. Secondly, the noise reduction mechanism is applied on the frames to minimize the noise and improve the signal to noise ratio (SNR) of the signal. Thirdly, the useful features are extracted from the frames and then related to a threshold limit and finally the VAD decision is calculated. If the value of extracted features exceed from the estimated threshold value than VAD output is 1 otherwise 0.

The core modules of any VAD system are *'feature extraction'* module and *'VAD decision'* module which can also be seen in the Figure 9. There are different feature extraction techniques such as Mel-Frequency Cepstral Co-efficient (MFCC), Power Normalized Cepstral Co-efficient (PNCC), Perceptual Linear Prediction (PLP) and each of them have associated benefits and drawbacks. Similarly multiple methods for speech/non-speech decision are available such as energy based classifier and neural

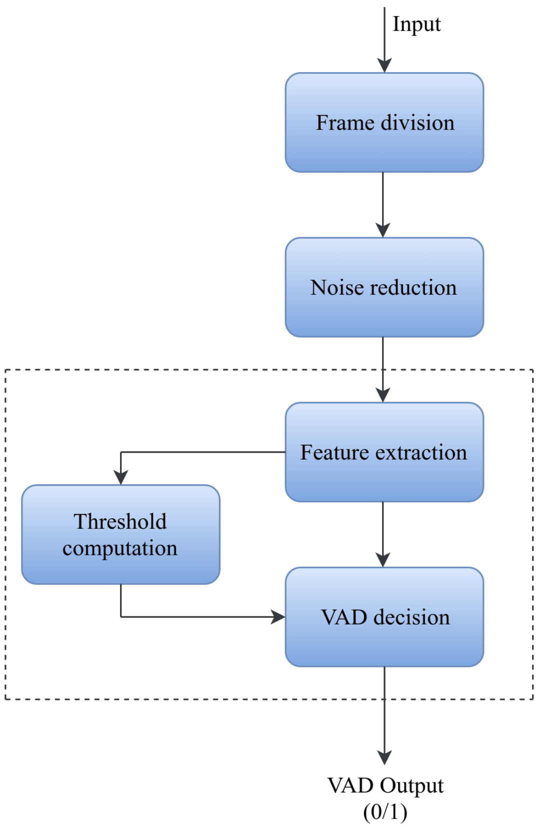

Figure 9: Block diagram of a VAD System

network classifier. In the following sections of this chapter, these two modules will be discussed in more detail.

5.2 Current implementation in Pocketsphinx

The current voice activity detection module of Pocketsphinx is implemented by combining different algorithms together. There is always a need to carefully tune the VAD parameters in order to achieve good performance and operate reliably in different noisy environments.

Figure 10 shows the nearest possible VAD implementation in *Pocketsphinx / Sphinxbase*. In Pocketsphinx, **MFCC** features are extracted from the input frames. Once the computation of features are done, than these feature vectors are fed to the noise estimation module which removes the silences present in the signal and also minimizes the noise. After the noise estimation step, the VAD decision is made for the feature frame by computing the overall signal-to-noise ratio and if the computed value is greater than the pre-setted VAD threshold than the frame contains the speech content. Similarly, if the value is lesser than the VAD threshold, it is treated as non-speech frame. The last step is the VAD hanging-over mechanism, which is implemented to increase the positive alarms and reduce the false alarms.

The noise estimation module implemented for Pocketsphinx's VAD is important and adopted from [47, 48]. Figure 10 also shows the components present in the noise estimation module. In this module, firstly, the "medium-time power" is calculated having the duration of 50 - 120 ms along with short-time Fourier analysis to Figure-out the parameters indicating environmental de-gradation. This approach provides the ability to respond to the speech signals changing rapidly. Secondly, the noise is minimized from the frame. The frame having background noise contains the high frequency spectrum, which can be suppressed by passing the frame through low pass filter. Thirdly, the temporal masking technique is applied on the frame, which is very useful for the reverberant environments. Lastly, the spectral weight-smoothing is performed, to remove the spectrum non-linearities.

The VAD hangover module keeps the track of VAD state and is also responsible to make the state's transition from speech to silence and silence to speech. The state transition from silence to speech will only occur if 10 speech frames are detected continuously. Similarly, for the transition of speech to silence state, the continuous 50 non-speech frames are required. These values for the state transitions can be tuned depending on the environmental conditions.

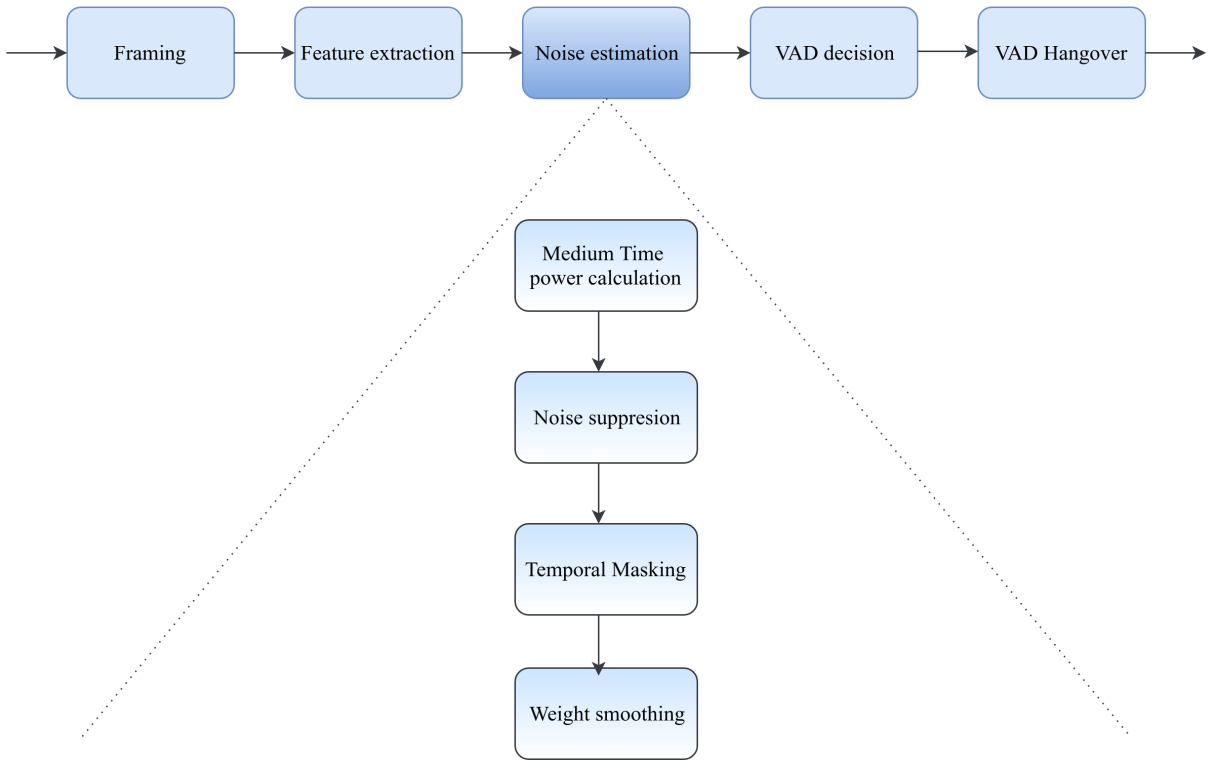

Figure 10: VAD implementation in Pocketsphinx

5.3 New feature extraction techniques

As discussed earlier, the feature extraction is an important part of VAD system and different extraction techniques are available which can be used with the Pocketsphinx's VAD classifier to obtain the new VAD algorithm. In order to improve the voice activity detection of Pocketsphinx, two new features are extracted besides MFCC, namely Gammatone Frequency Cepstral Coefficients (GFCC) [49, 3] and Gabor features [50, 3]. These two features are tested with the existing VAD classifier of the Pocketsphinx as well as with the newly implemented classifier. Following subsections will explain these new features in detail.

5.3.1 Gammatone frequency cepstral coefficients (GFCC) features

GFCC is a FFT based feature extraction technique. These features are similar to MFCC, but employing Gammatone-filterbanks and equivalent rectangular bandwidth (ERB) scale instead of Mel-filterbanks. GFCC features are also referred as auditory features, since gammatone filterbanks represents the nearest possible filter response of human ear cochlea [51]. The procedure for the extraction of simple GFCC features is summarised as follows.

1. **Pre-emphasis and Windowing:** Just like MFCC, pre-emphasis is also performed on the input speech signal for smoothing the speech spectrum as discussed in section 2.2.3 of Chapter 2. The transfer function of the pre-emphasis filter is given by the equation 2.4 where the default value of a_{pre} is -0.97. After the implementation of pre-emphasis, windowing is performed by dividing the speech signal into the sequences of frames. The frame size of 25ms and the frame shift of 10ms is considered for the GFCC features. Than each frame is passed through the Hamming window as discussed in equation 2.5 to reduce the spectral irregularities present at the frame edges.

2. **Discrete fourier transform:** The next step after windowing is to take the spectral response of each frame by taking the fast Fourier transform (FFT). As a result of FFT, a 512 point FFT spectrum of each frame is obtained.

3. **Gammatone filter-banks:** The next step is to pass the spectral response of each frame through the Gammatone filter-banks which consists of a series of bandpass filters. Equation 5.1 describes the impulse response of the gammatone filterbanks [51].

   $$g(i,t) = t^{k-1}e^{-2\pi Bt} * cos(2\pi f_i t) \tag{5.1}$$

   Here, t represents the time, k is the order of the filter and B is the rectangular bandwidth which increases with the increase in center frequency f. In other words, the center frequencies are distributed on the equivalent rectangular

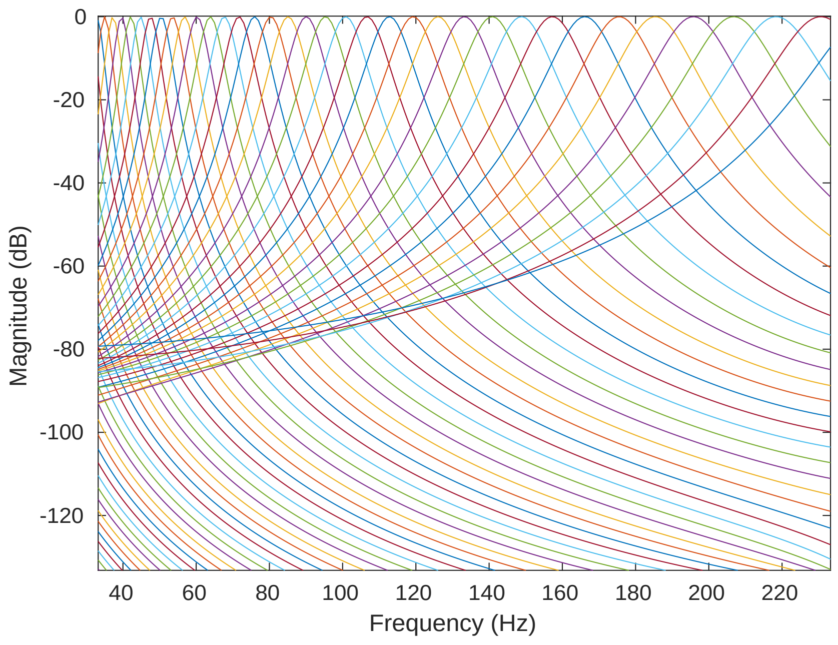

Figure 11: Frequency response of Gammatone filterbanks

bandwidth (ERB) scale equally. The range of center frequencies is typically from 50 Hz to 8000 Hz and can be defined by the equation 5.2 [52, 53, 54].

$$f_c = -228.7 + (f_{max} + 228.7)e^{-\frac{wn}{9.26}} \tag{5.2}$$

where, f_{max} is the maximum frequency and w presents the overlapping spacing. The frequency response of the Gammatone filter-banks can be seen in the Figure 11. The benefit of using Gammatone filter-banks with ERB scale is that the finer resolution can be achieved at the lower frequencies where most of the speech is concentrated.

4. **Cubic root compression and Discrete cosine transform:** The next step is the non-linear rectification and the loudness-compression of the magnitude of filter's output which is performed by taking the cubic root operation [50]. After the compression, discrete cosine transform is applied to the frame to decorrelate the components of the features which are correlated due to the neighbouring filter channels. The coefficients obtain as the result of DCT are known as Gammatone frequency cepstral coefficients (GFCC).

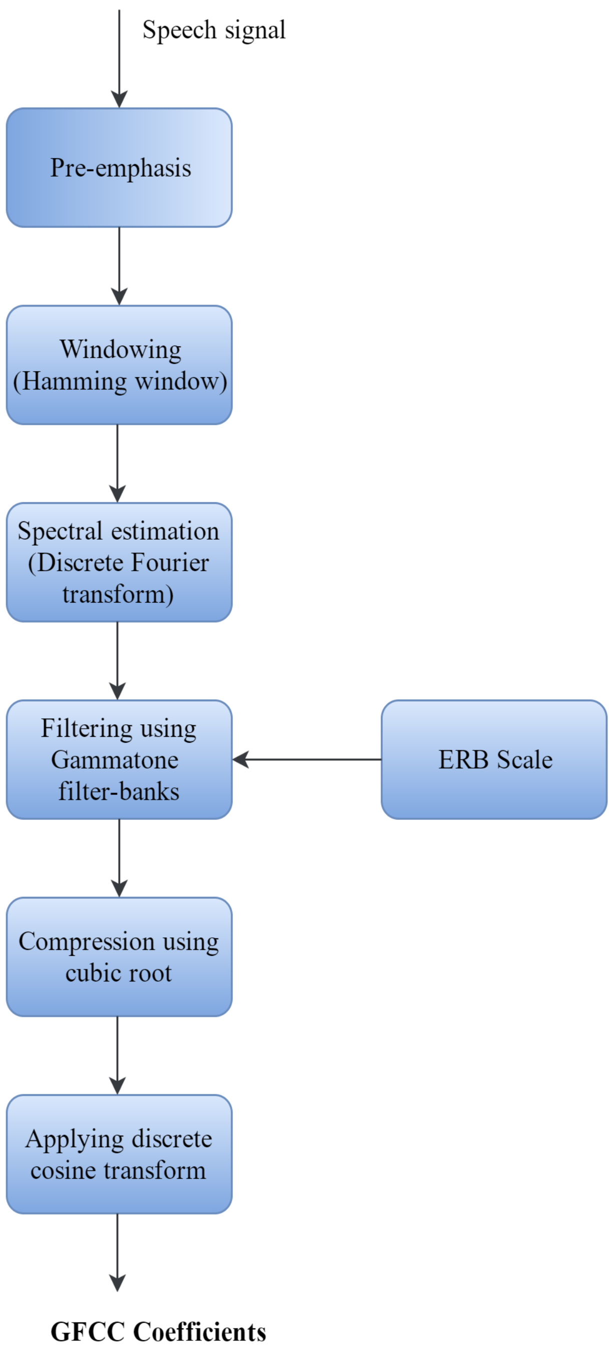
Speech signal
Pre-emphasis
Windowing (Hamming window)
Spectral estimation (Discrete Fourier transform)
Filtering using Gammatone filter-banks
ERB Scale
Compression using cubic root
Applying discrete cosine transform
GFCC Coefficients

Figure 12: Flowchart for extracting GFCC features

The Figure 12 shows the block diagram of the GFCC features as explained above.

5.3.2 Gabor features

Apart from GFCC, another new feature called *Gabor* feature is also extracted from the speech signal. Gabor features extract spectro-temporal cue from the input speech signals to increase the robustness of speech recognition. These new features are obtained by passing the log-Mel spectrogram from the number of two dimensional Gabor filters organized in a filter-bank [55]. Gabor filters simultaneously perform the spectral and temporal processing of the spectrogram.

Initially, just like other feature extraction techniques, the pre-emphasis and windowing operations are applied on the input speech signal and then the logarithmic Mel spectrogram is obtained by passing each frame through the 23 channelled Mel filter-bank and performing the logarithmic compression. After this, the 2D-Gabor filters organized in a filter-bank are convolved with the log-Mel spectrogram to extract the spectro-temporal patterns. Finally, the representative channels are selected by the critical sampling of filtered log-Mel spectrogram. The purpose of selecting these representative channels is to restrict the systematic correlation of feature dimensions [56]. These channels are combined and yield the Gabor features. The most important component of Gabor feature extraction technique is the 2D-Gabor filter banks which are discussed bellow in more detail.

Gabor filters

Figure 13 shows the 2D-Gabor filter bank having 14 filters arranged by spectro-temporal modulation frequencies. The values of temporal modulation frequencies (ω_n) are 0 and 0.6193 Hz, whereas the range of spectral modulation frequencies (ω_k) is considered between 0.1841 and 1.5708 $\frac{cycles}{Mel-band}$. Each 2D Gabor filter is just the product of complex sinusoidal carrier function having modulation frequencies ω_n and ω_k and the Hanning envelope function [57, 55].

$$g(n,k) = s(n,k) * h(n,k) \tag{5.3}$$

The complex sinusoid carrier function and the Hanning envelope function are described below by the equation 5.4 and 5.5.

$$s(n,k) = exp(i\omega_n(n-n_0) + i\omega_k(k-k_0)) \tag{5.4}$$

$$h(i,t) = 0.5 - 0.5cos(\frac{2\pi(n-n_0)}{\omega_n+1})cos(\frac{2\pi(k-k_0)}{\omega_k+1}) \tag{5.5}$$

The DC removal technique is also applied on the filters that are placed near to the edges of the spectrogram. The reason behind this is these filters do not occur

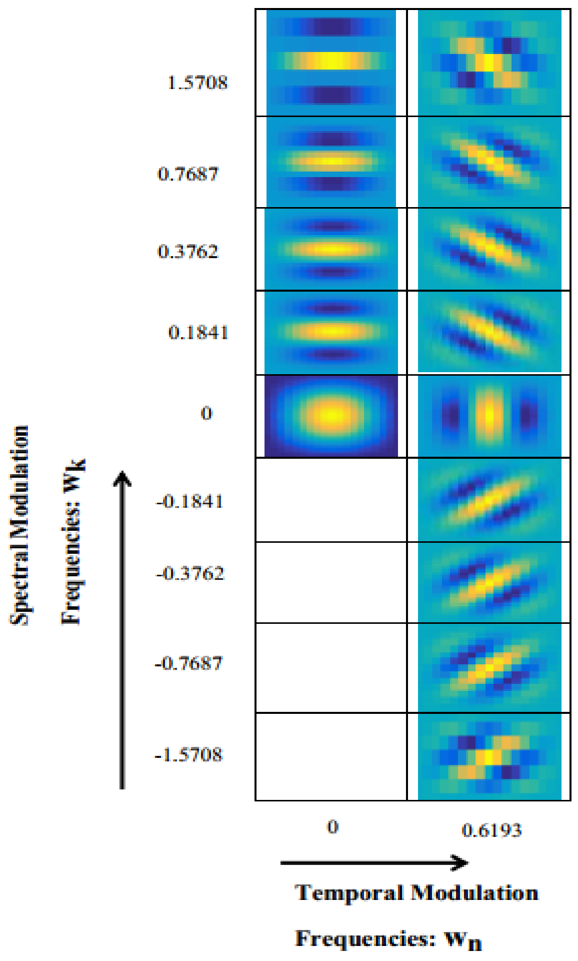

Figure 13: The real components of 2D-Gabor filter bank: 14 filters

completely within the boundaries of the spectrogram. So it is done by subtracting the normalized filter envelope function from the filter's function.

Representative frequency channels

One known issue is that the output of 2D-Gabor filter banks i.e. *filtered log-Mel spectrogram* cannot be used directly due to the high dimensionality and the presence of correlation between the output of adjacent filters. The filtered features have dimensions 23 $(frequency-channels)$ x 14 $(Gaborfilters)$. The selection of representative frequency channels is done in such a way that all the 23 frequency channels are selected for the filters with lowest spectral magnitude and for the filters with high magnitude only the central frequency channel is selected. This selection scheme of the features is quite handy as it decorrelates the feature vectors and the filtered output is reduced to only 110 dimensions which are considered as the **Gabor features**.

5.4 New VAD classifier

In this thesis, a pre-trained three-layered i.e. input layer, hidden layer and output layer, *MLP classifier* from [3] is used. This MLP classifier is trained with the labelled speech/non-speech training data. The duration of speech data to train this MLP classifier is only 30 hours. This three-layer MLP is trained on the combined input stream of GFCC and Gabor features. For the classification of input signal as speech or non-speech, the newly extracted features explained in the previous section of this chapter are fed to the MLP classifier. The dimension of GFCC feature vectors is 64 (filters) per frame and for the Gabor features it has 110 dimensions. So the linear combination of these two feature streams will provide the combined feature vectors with 174 dimensions. After combining the features, normalization is performed and the normalized feature vector is provided to the MLP classifier for the VAD decision.

This three-layer neural network works in reverse-direction and update the weights and find the error accordingly. The combined features are applied to the input layer and weights are calculated and updated there. The error vector is calculated at the output layer. If there is any error found, then it is back-propagated to the input layer. All these steps are performed iteratively and at the end VAD decision is made for the extracted features. Figure 14 shows the structure of the new VAD classifier used to evaluate the newly extracted features.

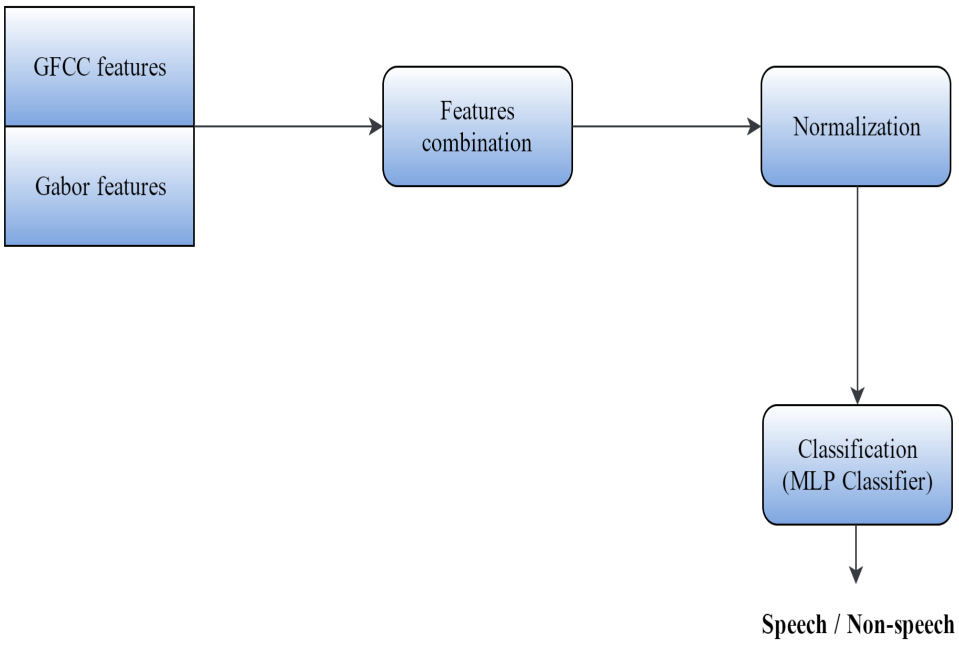

Figure 14: New VAD Classifier

5.5 Experiments and Results

In this section, the results will be discussed which are obtained by experimenting the new feature extraction techniques with the current VAD and the newly implemented VAD classifier in Pocketsphinx. The default feature extraction process implemented in the Pocketsphinx is extracting MFCC features. The results obtained by using MFCC features for the speech recognition are already discussed in the section 4.4 of Chapter 4. Experiments are conducted in such a way that the new features are used individually and together with the current VAD classifier. Three different datasets i.e. Dataset 1, 2 and 3, are used to evaluate the performance of these features. Dataset 1 consists of the evaluation data and Dataset 2 contains the testing data as discussed in Chapter 4. The dataset 1 comprises of the speech of 10 speakers i.e. 5 male and 5 female speakers. The total duration of speech in dataset 1 is 90 minutes. The dataset 2 contains data from the speakers having speech in the acoustic model. The total duration of speech in dataset 2 is 2 hours. The dataset 1 and 2 contains the Finnish speech and the level of background noise is high. For dataset 3 an exception is made as it contains English speech, recorded from 5 different English native speakers. The duration of speech in this set is 50 minutes. The level of background noise is also kept low as compared to the other datasets. Table 8 shows the results of new features and just for the reference and comparison purposes, MFCC results are also reported.

Table 8: % WER by using existing VAD

Features	**Dataset 1**	**Dataset 2**	**Dataset 3**
MFCC	4.87	5.28	6.56
GFCC	4.57	4.85	6.35
Gabor	4.70	5.05	6.50
MFCC + GFCC + Gabor	4.55	4.82	6.34

The results (WERs) obtained from the GFCC, Gabor and combined features are better than the conventional MFCC features. The combination of three features i.e. MFCC, GFCC and Gabor, when used with the existing VAD of the Pocketsphinx is producing the best results. The results obtained from the new VAD classifier are listed below in Table 9. Since, the three-layer MLP classifier is trained by using the combined features i.e. GFCC and Gabor, so only the combination of these new features is used with it.

Table 9: % WER by using new VAD classifier

Combined features	**Dataset 1**	**Dataset 2**	**Dataset 3**
GFCC + Gabor	4.78	5.07	5.02

For dataset 1 and 2, results of the new VAD classifier are not better than the old one, but for dataset 3, the WER is reduced. The main reason behind the improved result is the similarity of evaluation data with the MLP training data. So the new

VAD classifier works better when the speech data matches more with the training data of MLP classifier. After obtaining all these results, the decision has to be made for the best feature extraction technique and VAD classifier. The decision criteria is simply the recognition accuracy and the overall speech recognition speed. The recognition accuracy is already listed in Tables 8 and 9. The recognition speed indicates, how much of the CPU time is taken to complete the speech recognition process. For the extraction of Gabor features, 2D convolution is required for each filter from the filter-bank with log-Mel spectrogram. For this reason, the computational complexity increases and CPU takes more time for processing. Since the speech recognition engine is used on the Android platform, so the recognition speed is also of same importance as that of recognition accuracy. GFCC features stand on these limitations and provide the better performance than Gabor and combined features. The accuracy of the GFCC is much better than that of MFCC. So for these two reasons, GFCC features are used with the existing VAD classifier. Table 10 shows the averaged false recognitions which occurred when GFCC and MFCC features are used with the current VAD classifier. The false recognitions show the number of falsely recognized words which the recognizer incorrectly substituted against the correct word.

Table 10: Averaged False alarms for evaluation dataset

Features	**False Recognition**
GFCC	21.2
MFCC	29.8

Figure 15 shows the individual statistics of the speakers in terms of false recognitions present in the evaulation data. The VAD output of three randomly selected utterances from the evaluation data is also shown in the Figures 16, 17 and 18. In these figures, the reference utterance can be seen along with the two different outputs of speech recognizer i.e. one with the MFCC and other one with GFCC. For the comparison of these two features, the log Mel spectrogram is obtained from the speech signal. The yellow parts of the spectrogram are indicating the speech regions and the rest are non-speech. Also for the better visibility, the positive VAD ouput i.e. 1, is scaled to 10. It can been that GFCC is providing better results than MFCC.

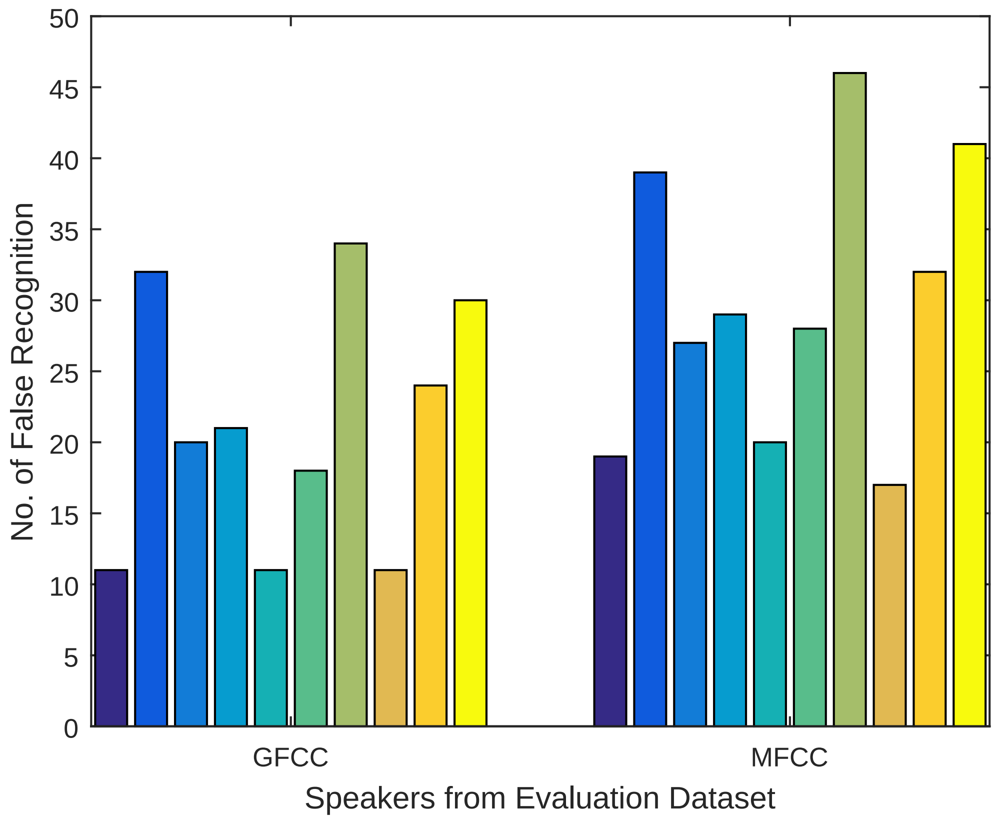

Figure 15: Comparison between GFCC and MFCC in terms of False Recognition

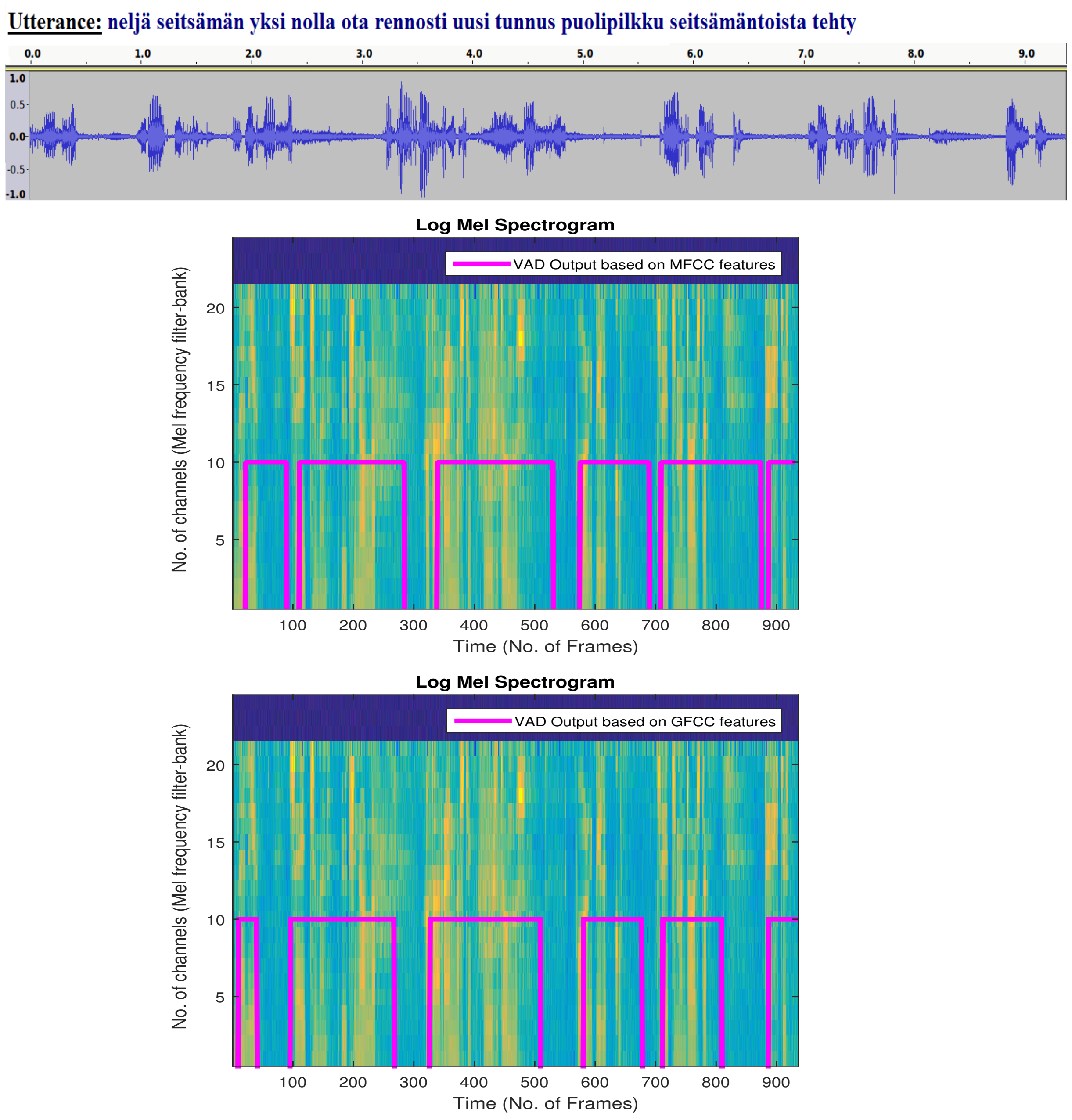

Figure 16: VAD output for utterance 1

The output of recognizer, when **MFCC** is used with current VAD classifier:

```
neljä seitsämän yksi nolla ota rennosti plus tunnus puolipilkku seit-
sämäntoista tehty
```

The output of recognizer, when **GFCC** is used with current VAD classifier:

```
neljä seitsämän yksi nolla ota rennosti uusi tunnus puolipilkku seit-
sämäntoista tehty
```

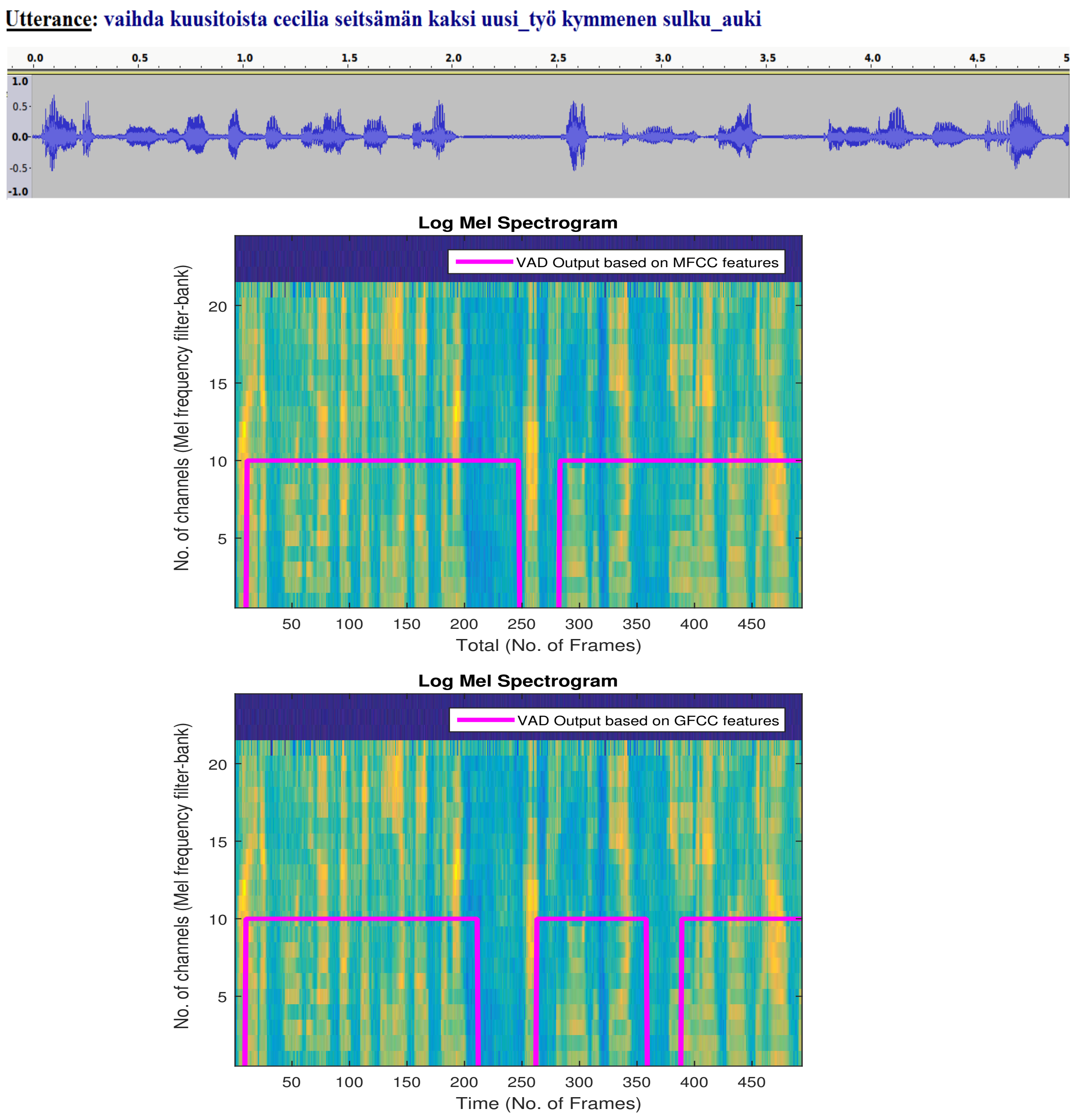

Figure 17: VAD output for utterance 2

The output of recognizer, when **MFCC** is used with current VAD classifier:

```
vaihda kuusitoista cecilia seitsämän yksi uusi_työ kymmenen sulku_auki
```

The output of recognizer, when **GFCC** is used with current VAD classifier:

```
vaihda kuusitoista cecilia seitsämän kaksi uusi_työ kymmenen sulku_auki
```

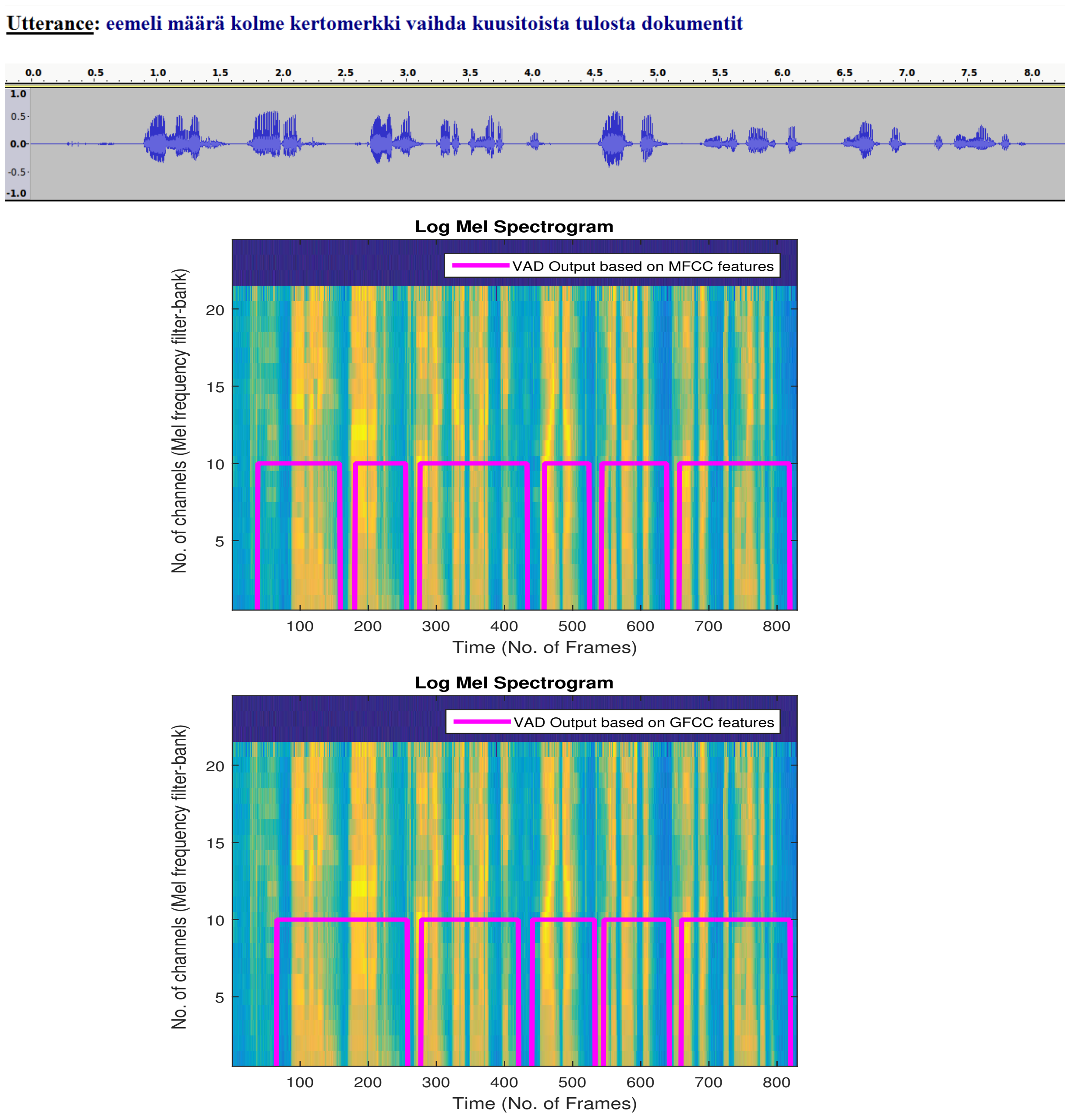

Figure 18: VAD output for utterance 3

The output of recognizer, when **MFCC** is used with current VAD classifier:

```
eemeli määrä työ kolme kertomerkki vaihda kuusitoista tulosta doku-
mentit
```

The output of recognizer, when **GFCC** is used with current VAD classifier:

```
eemeli määrä kolme kertomerkki vaihda kuusitoista tulosta dokumen-
tit
```

Chapter 6

6 Garbage Modelling for Out-Of-Vocabulary Words

6.1 Introduction

One of the most important part of any speech recognition application is the detection of words and sometimes utterances which are out of the vocabulary. The OOV words can also contain distorted and non-words. Distorted words are the mispronounced words and non-words have no meaningful definition. The presence of such words or sounds in the input speech signal can have adverse effect on the performance of any speech recognition engine. Most of the ASR applications have limited and customized vocabularies which are used on the embedded platforms such as mobiles and tablets. Such applications continuously remain in the listening mode and keep processing the speech data. This is also a known fact that the speakers can speak many words which are unknown to the ASR engine, while using the speech recognition applications on mobile devices. These unknown words can really decrease the recognition performance by extending the errors into the adjacent words and thus increasing the overall WER.

The OOV words can be encountered when it comes to both, the small and large vocabulary ASR applications, as the vocabulary of any language is not fixed and is constantly changing. However, the magnitude of this problem can vary and according to [58] it depends on the vocabulary size and the similarity between the training corpus and the testing corpus. If the amount of words in a particular vocabulary is more than 60 thousands and the mismatch between the training and testing corpora is negligible than the WER due to the OOV words could be just a fraction of a percent. Similarly, for the scenario where the number of words in a vocabulary is just a few hundreds or thousands and the mismatch between the training and evaluation corpora is big, than the WER due to the OOV words could be more than 50%. It is also interesting to know, what usually happens when any OOV word is spoken to the speech recognition engine. The recognizer will take that speech signal and hypothesize a similar sounding word from the vocabulary. One similar example can be seen below.

```
Utterance fed to the recognizer:      yksi kaksi kolme soita


Utterance hypothesized by recognizer: yksi kaksi kolmetoista
```

Here the OOV word is *soita* and the recognizer gave the similar sounding output **kolmetoista** in place of **kolme soita**. In this utterance, though the OOV word is only one, but the WER is 50%, and can be explained with the help of three factors as mentioned in [58]. The first factor is the miss-recognition of word soita since it is from outside the vocabulary. The second factor is the miss-recognition of the neighbouring word i.e. kolme. The third factor is hard recognition of out-of-domain utterances, which is more observable when the application is using large vocabulary continuous speech recognition system.

Various attempts have been made earlier to tackle the problem of OOV words. Some of these approaches are vocabulary growth, confidence measure and filler models. The vocabulary size matters when it comes to the detection of OOV words. If the vocabulary size is increased by adding more domain related words, than WER introduced due to the OOV words can be decreased but not eliminated. The major drawbacks for this approach is the persistence of OOV words (i.e. still unknown words can be detected after the growth of vocabulary) and the computational complexity to search the in-vocabulary word increases, which make the overall recognition quite a slow process.

The other approach is the measurement of the confidence score for the recognized word by using posterior probabilities associated to the hypothesized words [59, 60]. Some threshold can be decided and if the confidence score of a word is lesser than the threshold, it can be rejected. The last approach is most commonly used to handle OOV words and called filler models [61]. These filler models can be used in parallel to the language models and can absorb the OOV words.

In the next section, a new technique is discussed for which the base is provided by the above mentioned approaches, to tackle the problem of handling OOV words. The garbage model is implemented in the Pocketsphinx, useful for the small vocabulary systems. Experiments and results shall also be presented in the later section.

6.2 Garbage modelling technique

In this section, an important technique called *generic word model* or *phone loop model*, is discussed for creating the garbage model for OOV words [58, 62, 63]. This technique employs parallel branch for recognizing the OOV words in addition to the standard word-recognizer branch. This additional branch uses subword network to recognize the subword sequence for an OOV word. The subwords can be phones or syllables.

6.2.1 Generic word model

The two distinct characteristics of the generic word model are the lexicon and language model. The lexicon has two set of vocabularies. The first set contains the phonetic definition of in-vocabulary words and the second set consists of the OOV words. The size of the In-Vocabulary lexicon can range from hundred to several thousand. The size of the out-vocabulary lexicon depends on the efficiency and performance of the recognizer i.e. if the number of words are high, the efficiency of recognition search is degraded due to the additional complexity added to the search space.

Just like lexicon, language model also consists of two different language models. For the In-vocabulary words, the standard n-gram model is used and the second model is a subword language model which is only utilized when OOV word is encountered. The framework for the generic word model is shown in the Figure 19. Each search network i.e. In-vocabulary and out-vocabulary, is having its own lexicon and language model.

The In-vocabulary search network is the standard word based recognizer and the out-vocabulary search network is an addition to the system which recognizes the sequence of subwords. For In-vocabulary search network, the finite-state transducer (FST) representation of the search space for recognizing the best search path can be described as equation 6.1 [62]. Where P is the scored phonetic graph, L is the word

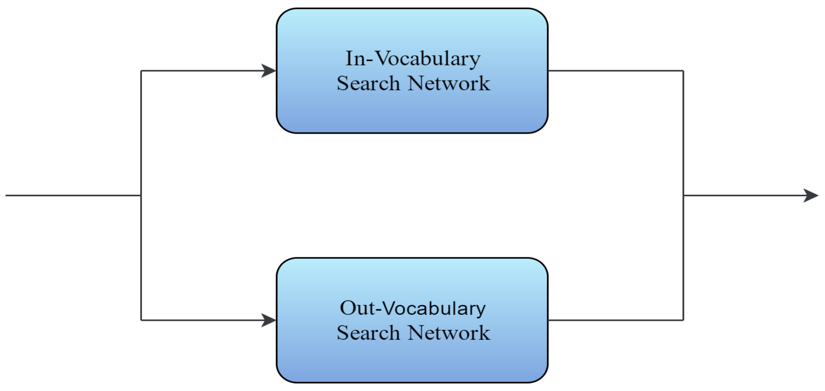

Figure 19: Framework of generic word model

lexicon and G is the language model.

$$S = P \ . \ L \ . \ G \tag{6.1}$$

For the out-vocabulary search network, a simple phonetic-based recognizer is considered. The vocabulary of this recognizer is made up of phones which can easily cover all possible unknown or OOV words. It also has an advantage for small size. Similarly, the search space for this network can be described by equation 6.2, where G_p represents the phone-level n-gram language model.

$$S = P \ . \ G_p \tag{6.2}$$

In order to facilitate the recognition of OOV words, these two search branches are combined in parallel, where they compete with each other for the best search path to recognize the words or subwords. The transition from In-vocabulary search network to the out-vocabulary search network for the best path is done on the basis of threshold called *OOV threshold*. The value of threshold is decided on the basis of probability assigned to OOV word by the standard recognizer. After obtaining the hypotheses from the two search networks, the search spaces are merged.

The FST representation of combined search networks is given by equation 6.3. The union sign i.e. $\cup$ represents that the choice can be made between the In-vocabulary and out-vocabulary search networks during the recognition. The $*$ represents the closure operation on FST, which means that the output result can consists of both search spaces or one of the search space. If no OOV word is present in the input signal, than the search space for In-vocabulary network is only considered and vice-versa.

$$S = P \ . \ (L \cup L_p \ . \ G_p)^* \ . \ G \tag{6.3}$$

Figure 20 shows the highlighted best search path for the utterance **`yksi kaksi kolme`** ***`tyhjä`***.

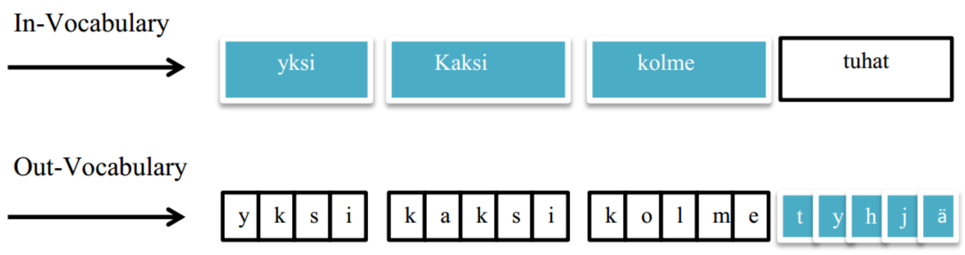

Figure 20: Best search path

In Pocketsphinx, the garbage model for OOV words is not implemented. But it has the functionality of performing phone-based recognition. So the changes have been made in Pocketsphinx to use the phone-based recognition in parallel with the word-based recognition by setting the proper threshold for the phoneme insertion penalty. After the introduction of garbage model in Pocketsphinx, the utterance containing any word outside the grammar will be considered garbage as a whole and return **UNKNOWN**.

6.3 Experiments and Results

In this section, the results will be discussed which are obtained by testing the newly implemented garbage model in Pocketsphinx. The datasets used to evaluate the performance of new acoustic models and the VAD module cannot be used directly, since it do not contain any OOV words. In order to use the evaluation dataset, some of the words present in the speech data are removed from the grammar and lexicon to make them unknown to the system. For this purpose, 100 random utterances were selected from the evaluation data and out of these selected utterances, 53 utterances consist of In-vocabulary words and the remaining 47 utterances consist of words from inside and outside the vocabulary. This subset of evaluation data is evaluated with different values of garbage phone insertion threshold. Table 11 shows the results obtained from the selected dataset.

Table 11: Results from the subset of evaluation data for evaluating garbage model

Garbage Phone Insertion Threshold	**Correct IV**	**False IV**	**Correct OOV**	**False OOV**
1e-50	45	20	27	8
1e-40	44	15	32	9
1e-30	37	7	40	16
1e-20	37	5	42	16
1e-10	32	3	44	21

It can be seen that, the correct recognition of OOV utterances is improving with the increase of insertion threshold for the garbage phones. Also with the increase of this threshold, most of the In-vocabulary utterances are recognized as OOV, which is not desired. So the appropriate value of threshold has to be selected by fulfilling two conditions. The first condition is the good rejection accuracy and second one is the overall good recognition accuracy. The threshold of $1e - 20$ seems to be that value. Figure 21 shows the rejection accuracy and overall recognition accuracy against the garbage phone insertion threshold. The rejection accuracy is just the percentage of utterances that are recognized correctly as out of vocabulary. The overall accuracy is also interesting which explains the percentage of utterances correctly classified as In-vocabulary and out-of-vocabulary. If the value of threshold is kept on increasing

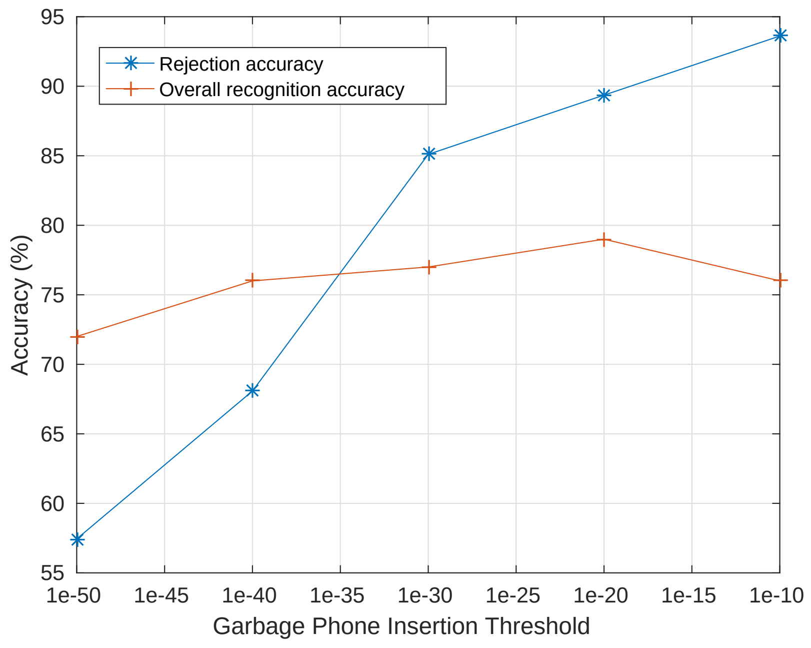

Figure 21: Rejection accuracy for the different values of garbage phone insertion threshold

than every utterance will be identified as out of vocabulary.

The live test was also conducted for measuring the accuracy of the garbage model implementation. Different speakers i.e. both native and non-native, participated in the experiment. Each speaker has to speak 93 different utterances. The composition of these utterances is made in such a way that 39 utterances consist of only In-vocabulary words, 45 utterances are composed of only OOV words and 9 utterances contain both In-vocabulary and OOV words. The value of phone insertion threshold is $1e-15$.

The Table 12 shows the results obtained from all speakers during the live test. 37 utterances out of 39 are correctly recognized as In-vocabulary and only 2 utterances are miss-recognized as OOV. The rejection of utterances containing only unknown words is also good. On average 37.5 utterances are recognized by the garbage model and 16.5 utterances are considered from inside the vocabulary.

Table 12: Results from the live test for evaluating garbage model

Correct IV	**False IV**	**Correct OOV**	**False OOV**
37 utterances	16.5 utterances	37.5 utterances	2 utterances

The Table 13 shows the overall accuracy of the recognizer along with the rejection accuracy by rejecting the OOV words. The rejection accuracy and overall accuracy can be improved by tuning the threshold for the garbage phone insertion penalty. The range of the threshold for better results should be kept between $1e-15$ and

$1e-30$. This value of threshold can vary for different environments depending on the background noises.

Table 13: Rejection accuracy

Rejection Acuracy	**Overall Accuracy**
70%	80%

For garbage modelling, the quality of detection can also be expressed in two parameters namely true positive alarms and false positive alarms. The true positive alarm (TPA) can be considered as a ratio of the total count of correctly recognized OOV words to the total number of OOV words. The false positive alarm (FPA) is the ratio of the count of wrongly hypothesized OOV words to the total count of In-vocabulary words. The relation between the TPA and FPA can express in the receiver operation characteristics (ROC) curve. Figure 22 shows such ROC curve for OOV detections obtained from the live test. If the obtained curve is closer to the diagonal, the false alarm rate would be higher which indicates the overall poor recognition accuracy and vice-versa.

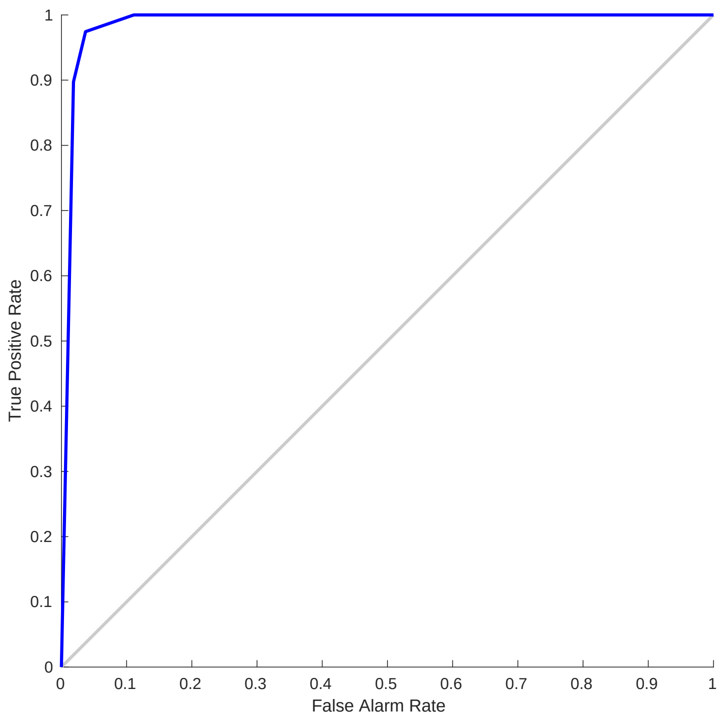

Figure 22: ROC curve for OOV words

Chapter 7

7 Conclusion

In this master thesis, three different tasks are implemented to increase the accuracy of a speech recognition engine. For the first task, a Finnish acoustic model is trained. The data required to train and evaluate the acoustic model is provided by Devoca Oy. Speech data is recorded from the warehouse environment of different companies, and the motive behind this approach is to incorporate more noise in the training data. Sphinxtrain from CMU Sphinx toolkit is used to train the acoustic model. Two different acoustic models i.e. Semi-continuous and PTM, are created. In terms of recognition accuracy, PTM model outperforms the semi-continuous model. Both AMs are evaluated on two different datasets and for each dataset the WER of PTM model is less than for the semi-continuous model. Besides the recognition accuracy, the recognizer's speed is also considered. When both new AMs are tested on the mobile platform, the PTM model appeared to be quite slow as compared to the semi-continuous model. So the trade-off has to make between the accuracy and speed and finally the semi-continuous is selected. In future, PTM model can be considered but require more experiments.

For the second task, new feature extraction techniques are considered and implemented for the speech activity detection in the Pocketsphinx. Currently, the MFCC features are extracted from the speech signal by default. The new features namely GFCC and Gabor features are extracted in addition to MFCC. These new features are fed to the current VAD classifier of the Pocketsphinx, individually and together. The results have shown that all the new features are more robust to noisy conditions, since the WER obtained by using the MFCC feature is more as compared to the other features. In addition to these experiments, GFCC and Gabor features are combined and fed to the three-layered MLP classifier, which classify the input signal into speech or non-speech by giving 0 or 1 respectively. This new classifier showed some good results, but for some datasets the current classifier worked better. The best result, in terms of low WER, is obtained by providing the combined features (GFCC, Gabor and MFCC) to the Pocketsphinx's VAD classifier. However, the Gabor features have introduced a lot of computational complexity, which slowed down the entire speech recognition process. So again, the decision is made to use the GFCC features with the VAD classifier of Pocketsphinx. In future, the current VAD classifier can be replaced by the deep neural network (DNN) classifier trained from the multiple features.

For the third task, a garbage model is implemented in the Pocketsphinx to handle the unknown or OOV words. The technique considered and implemented for this task is employing a new search network in parallel to the standard word-based search network. This new search network can recognize the OOV words by expanding them into their acoustical definition i.e. phoneme based recognition. During the decoding process, the recognizer can switch between the two search networks and the end result can be the combination of both search branches. The implementation of this generic word model produced best results and detected 70% of the OOV words. Garbage model also works equally good for the detection of non-words and rejects them as well. Of-course, this OOV words detection can be improved by tuning the threshold for phone insertion penalty. In future the Garbage model can be implemented along with the confidence measure in Pocketsphinx to increase the overall accuracy as discussed in [64] and also neural network based garbage system can be implemented to detect OOV words [65].

References

[1] W. Holmes, *Speech synthesis and recognition.* CRC press, 2001.

[2] 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.

[3] M. Van Segbroeck, A. Tsiartas, and S. Narayanan, "A robust frontend for vad: exploiting contextual, discriminative and spectral cues of human voice." in *INTERSPEECH*, 2013, pp. 704–708.

[4] Y. Kida and T. Kawahara, "Voice activity detection based on optimally weighted combination of multiple features." in *INTERSPEECH*, 2005, pp. 2621–2624.

[5] M. Fujimoto, K. Ishizuka, and T. Nakatani, "A voice activity detection based on the adaptive integration of multiple speech features and a signal decision scheme," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing.* IEEE, 2008, pp. 4441–4444.

[6] "CMU Sphinx, a project by Carnegie Mellon University," 2016, available at http://cmusphinx.sourceforge.net/. Accessed on 18.08.2016.

[7] M. Beutnagel, A. Conkie, and A. K. Syrdal, "Diphone synthesis using unit selection," in *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998.

[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] F. Jelinek, L. Bahl, and R. Mercer, "Design of a linguistic statistical decoder for the recognition of continuous speech," *IEEE Transactions on Information Theory*, vol. 21, no. 3, pp. 250–256, 1975.

[10] H. A. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach.* Springer Science & Business Media, 2012, vol. 247.

[11] 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.

[12] 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.

[13] M. Kawamoto, K. Matsuoka, and N. Ohnishi, "A method of blind separation for convolved non-stationary signals," *Neurocomputing*, vol. 22, no. 1, pp. 157–171, 1998.

[14] D. A. Kapilow, Y. Stylianou, and J. Schroeter, "Detection of non-stationarity in speech signals and its application to time-scaling." in *EUROSPEECH*, 1999.

[15] W. Han, C.-F. Chan, C.-S. Choy, and K.-P. Pun, "An efficient mfcc extraction method in speech recognition," in *2006 IEEE international symposium on circuits and systems*. IEEE, 2006, pp. 4–pp.

[16] 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.

[17] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[18] J. R. Deller Jr, J. G. Proakis, and J. H. Hansen, *Discrete time processing of speech signals*. Prentice Hall PTR, 1993.

[19] J. W. Picone, "Signal modeling techniques in speech recognition," *Proceedings of the IEEE*, vol. 81, no. 9, pp. 1215–1247, 1993.

[20] "National Instruments, Understanding FFTs and Windowing," 2016, available at http://www.ni.com/white-paper/4844/en/. Accessed on 06.12.2016.

[21] A. D. Poularikas, *Handbook of formulas and tables for signal processing*. CRC Press, 1998, vol. 13.

[22] S. Molau, M. Pitz, R. Schluter, and H. Ney, "Computing mel-frequency cepstral coefficients on the power spectrum," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 73–76.

[23] X. Xiong, "Robust speech features and acoustic models for speech recognition," *Nanyang Technological University, PhD thesis*, 2009.

[24] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[25] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[26] A. Stolcke *et al.*, "Srilm-an extensible language modeling toolkit." in *Interspeech*, vol. 2002, 2002, p. 2002.

[27] A. Waibel, *Readings in speech recognition*. Morgan Kaufmann, 1990.

[28] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing.* MIT Press, 1999, vol. 999.

[29] 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.

[30] J. Pylkkönen *et al.*, "Towards efficient and robust automatic speech recognition: decoding techniques and discriminative training," 2013.

[31] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[32] S. J. Young, N. H. Russell, and J. H. S. Thornton, "Token passing: a simple conceptual model for connected speech recognition system," Cambridge University Engineering Department, Tech. Rep., 1989.

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

[34] A. C. Morris, V. Maier, and P. D. Green, "From wer and ril to mer and wil: improved evaluation measures for connected speech recognition." in *INTER-SPEECH*, 2004.

[35] K. Zechner and A. Waibel, "Minimizing word error rate in textual summaries of spoken language," in *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference.* Association for Computational Linguistics, 2000, pp. 186–193.

[36] "CMUSphinx, Overview of CMUSphinx toolkit," 2016, available at http://cmusphinx.sourceforge.net/wiki/tutorialoverview. Accessed on 07.12.2016.

[37] "Voxforge, What is an Acoustic Model," 2016, available at http://www.voxforge.org/home/docs/faq/faq/what-is-an-acoustic-model. Accessed on 16.12.2016.

[38] "Sourceforge, Acoustic and Language Models," 2016, available at https://sourceforge.net/projects/cmusphinx/files/Acoustic%20and%20Language%20Models/. Accessed on 17.12.2016.

[39] A. Lee, T. Kawahara, K. Takeda, and K. Shikano, "A new phonetic tied-mixture model for efficient decoding," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, vol. 3. IEEE, 2000, pp. 1269–1272.

[40] Y. Liu and P. Fung, "State-dependent phonetic tied mixtures with pronunciation modeling for spontaneous speech recognition," *IEEE Transactions on speech and audio processing*, vol. 12, no. 4, pp. 351–364, 2004.

[41] 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.

[42] X. Huang, K.-F. Lee, and H.-W. Hon, "On semi-continuous hidden markov modeling," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on.* IEEE, 1990, pp. 689–692.

[43] "CMUSphinx, Acoustic Model Types," 2016, available at http://cmusphinx.sourceforge.net/wiki/acousticmodeltypes. Accessed on 21.12.2016.

[44] "Yandex, ASR for mobile applications in Yandex," 2016, available at https://events.yandex.com/events/yac/2013/talks/83/. Accessed on 22.12.2016.

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

[46] "CMUSphinx, Training Acoustic Model for CMUSphinx," 2016, available at http://cmusphinx.sourceforge.net/wiki/tutorialam. Accessed on 28.12.2016.

[47] C. Kim and R. M. Stern, "Power-normalized cepstral coefficients (pncc) for robust speech recognition," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2012, pp. 4101–4104.

[48] C. Kim, "Signal processing for robust speech recognition motivated by auditory processing," Ph.D. dissertation, Johns Hopkins University, 2010.

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

[50] Y. Shao, Z. Jin, D. Wang, and S. Srinivasan, "An auditory-based feature for robust speech recognition," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing.* IEEE, 2009, pp. 4625–4628.

[51] K. A. Das, K. K. George, C. S. Kumar, S. Veni, and A. Panda, "Modified gammatone frequency cepstral coefficients to improve spoofing detection," in *Advances in Computing, Communications and Informatics (ICACCI), 2016 International Conference on.* IEEE, 2016, pp. 50–55.

[52] R. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice, "An efficient auditory filterbank based on the gammatone function," in *a meeting of the IOC Speech Group on Auditory Modelling at RSRE*, vol. 2, no. 7, 1987.

[53] W. H. Abdulla, "Auditory based feature vectors for speech recognition systems," *Advances in Communications and Software Technologies*, pp. 231–236, 2002.

[54] M. Slaney *et al.*, "An efficient implementation of the patterson-holdsworth auditory filter bank," *Apple Computer, Perception Group, Tech. Rep*, vol. 35, p. 8, 1993.

[55] I. Missaoui and Z. Lachiri, "Physiologically motivated feature extraction for robust automatic speech recognition," *International Journal of Advanced Computer Science & Applications*, vol. 1, no. 7, pp. 297–301, 2016.

[56] M. R. Schädler and B. Kollmeier, "Normalization of spectro-temporal gabor filter bank features for improved robust automatic speech recognition systems." in *INTERSPEECH*, 2012, pp. 1812–1815.

[57] M. R. Schädler, B. T. Meyer, and B. Kollmeier, "Spectro-temporal modulation subspace-spanning filter bank features for robust automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 131, no. 5, pp. 4134–4151, 2012.

[58] I. Bazzi, "Modelling out-of-vocabulary words for robust speech recognition," Ph.D. dissertation, Massachusetts Institute of Technology, 2002.

[59] T. J. Hazen, T. Burianek, J. Polifroni, and S. Seneff, "Recognition confidence scoring for use in speech understanding systems."

[60] S. O. Kamppari and T. J. Hazen, "Word and phone level acoustic confidence scoring," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, vol. 3. IEEE, 2000, pp. 1799–1802.

[61] A. S. Manos and V. W. Zue, "A segment-based wordspotter using phonetic filler models," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, vol. 2. IEEE, 1997, pp. 899–902.

[62] I. J. Glass, "Modeling out-of-vocabulary words for robust speech recognition," in *The Proceedings of the 6˜(th) International Conference on Spoken Language Processing (Volume I)*, 2000.

[63] J.-C. Junqua and J.-P. Haton, *Robustness in automatic speech recognition: Fundamentals and applications.* Springer Science & Business Media, 2012, vol. 341.

[64] T. J. Hazen and I. Bazzi, "A comparison and combination of methods for oov word detection and word confidence scoring," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 397–400.

[65] S. Kombrink, M. Hannemann, and L. Burget, "Out-of-vocabulary word detection and beyond," in *Detection and Identification of Rare Audiovisual Cues.* Springer, 2012, pp. 57–65.