DISSERTATIONS IN

FORESTRY AND NATURAL SCIENCES

EVGENY KARPOV

Efficient Speaker Recognition for Mobile Devices

PUBLICATIONS OF THE UNIVERSITY OF EASTERN FINLAND
Dissertations in Forestry and Natural Sciences

UNIVERSITY OF EASTERN FINLAND

EVGENY KARPOV

Efficient Speaker Recognition for Mobile Devices

Publications of the University of Eastern Finland
Dissertations in Forestry and Natural Sciences
No 52
Academic Dissertation

To be presented by permission of the Faculty of Science and Forestry for public examination in Louhela auditorium of the Science Park, at the University of Eastern Finland, Joensuu, on November 19th, 2011,
at 12 o'clock noon.

School of Computing

Kopijyvä Oy
Joensuu, 2011
Editors: *Prof. Pertti Pasanen, Prof. Kai-Erik Peiponen, and Prof. Matti Vornanen.*

Distribution:
University of Eastern Finland Library / Sales of publications
P.O. Box 107, FI-80101 Joensuu, Finland
tel. +358-50-3058396
http://www.uef.fi/kirjasto

ISBN: 978-952-61-0590-1 (printed)
ISSNL: 1798-5668
ISSN: 1798-5668
ISBN: 978-952-61-0591-8 (pdf)
ISSNL: 1798-5668
ISSN: 1798-5676

Author's address: University of Eastern Finland
School of Computing
P.O.Box 111
80101 JOENSUU
FINLAND
email: ekarpov@student.uef.fi

Supervisors: Prof. Pasi Fränti
University of Eastern Finland
School of Computing
P.O.Box 111
80101 JOENSUU
FINLAND
email: pasi.franti@uef.fi

Dr. Tomi Kinnunen
University of Eastern Finland
School of Computing
P.O.Box 111
80101 JOENSUU
FINLAND
email: tomi.kinnunen@uef.fi

Reviewers: Prof. Paavo Alku
Aalto University
Department of Signal Processing and Acoustics
P.O. Box 13000
FI-00076 Aalto, Finland
email: paavo.alku@aalto.fi

Prof. Yuri N. Matveev
The National Research University of Information Technologies
Department of Speech Information Systems
49 Kronverkskiy pr.,St. Petersburg, Russia, 197101
email: matveev@mail.ifmo.ru

Opponent: Prof. Zheng-Hua Tan
Multimedia Information and Signal Processing (MISP)
Department of Electronic Systems, Aalborg University
Niels Jernes Vej 12, 9220 Aalborg, Denmark
email: zt@es.aau.dk

ABSTRACT

Speaker recognition has been an active topic of research for several decades already. Successful application utilizing this technology can already be found from the market. However, reliable recognition system requires fast and expensive hardware to operate in reasonable time. In this work, we concentrate our efforts to squeeze a complex recognition system into low-cost and low-resource hardware usually found on a typical mobile phone.

This work has been a long journey for the author who has been working during this time in several closely related domains such as speaker recognition, speaker verification and speech recognition. While these are different applications, they share the same underlying components. A synergy of these areas has found its place in this work as well.

The research in this thesis mainly addresses the limitations that are found in typical mobile phone implementation. Firstly, we propose several *speaker pruning* methods which are able to significantly speed up the speaker matching step. Secondly, we limit the number of computations by effectively removing redundant data from speech input utilizing *voice activity detection* and feature vector *quantization techniques*.

Finally, we look into mobile phone specific design issues, most notably the absence of floating point unit and analyze algorithm conversion results to run on a fixed point processor with as little degradation in accuracy as possible. We also present efficient *model quantization* techniques that have been used in the speech recognition domain but can be easily applied to speaker verification as well.

Keywords: Speaker recognition, speaker verification, computational complexity, voice activity detection, speaker pruning, quantization, fixed-point implementation

Acknowledgements

The work presented in this thesis has started in the Computer Science Department of the University of Joensuu, Finland, in the year 2003. In 2005 the author has moved to work for Nokia Research Center (NRC), Tampere, and research work has continued there. At the end of 2008 the author has left NRC and continued this work on his own time. During that time also the University name has changed after the merge to the University of Eastern Finland.

There have been a lot of changes throughout this work and the biggest challenge was to keep going forward. For that I owe my deepest thanks to my supervisor Professor Pasi Fränti. Without his guidance and pushing me forward I believe I would not ever finish this work. Special thanks are also due to my other supervisor Dr. Tomi Kinnunen, whose endless support and experience have helped me to advance in this area. Tomi is a real research geek who showed me what real dedication and hard work can achieve. Pasi and Tomi have also co-authored many of the publications in this thesis.

I am grateful to all people who have worked with me in the in the University of Joensuu and Nokia Research Center. I hope we had great time together. Special thanks are due to Juhani Saastamoinen, PUMS project manager, and Dr. Imre Kiss from Nokia Research Center. I am thankful to Professor Paavo Alku and Yuri Matveev, the reviewers of the thesis, for their constructive feedback and comments.

Finally, I would like to thank my wife Elena and my son Danila for their support and understanding during the time I had to be absent from them while working on this thesis. I would also like to thank my parents Tatjana and Alexander for their long lasting support throughout my life.

Tampere, Sunday 30th of October 2011 – Evgeny Karpov

List of original publications

Method optimizations

P1. T. Kinnunen, E. Karpov and P. Fränti, "A speaker pruning algorithm for real-time speaker identification", *Proceedings of the International Conference on Audio- and Video-Based Biometric Person Authentication* (AVBPA'03), Guildford, UK, Springer Lecture Notes in Computer Science vol. 2688, 639-646, June 2003.

P2. T. Kinnunen, E. Karpov and P. Fränti, "Efficient online cohort selection method for speaker verification", *Proceedings of the International Conference on Spoken Language Processing*, (ICSLP'04), Jeju Island, South Korea, Vol. 3, 2401-2405, October 2004.

P3. T. Kinnunen, E. Karpov and P. Fränti, "Real-time speaker identification and verification", *IEEE Transactions on Audio, Speech and Language Processing*, 14 (1), 277-288, January 2006.

P4. E. Karpov, T. Kinnunen and P. Fränti, "Symmetric distortion measure for speaker recognition", *Proceedings of the International Conference Speech and Computer* (SPECOM'2004), St. Petersburg, Russia, 366-370, September 2004.

Mobile device specific optimizations

P5. J. Saastamoinen, E. Karpov, V. Hautamäki and P. Fränti, "Accuracy of MFCC based speaker recognition in Series 60 device", *EURASIP Journal of Applied Signal Processing* 17 (2005), 2816-2827.

P6. E. Karpov, I. Kiss, J. Leppanen, J. Olsen, D. Oria, S. Sivadas, J. Tian, "Short message dictation on Symbian series 60 mobile phones", *In Proceedings of MobileHCI 2006 Workshop on Speech in Mobile and Pervasive Environments*., Espoo, Finland, 2006.

P7. E. Karpov, Z. Nasibov, T. Kinnunen, P. Fränti "Combining voice activity detection algorithms by decision fusion", *Proceedings of International Conference Speech and Computer* (SPECOM'2011), Kazan, Russia, 278-283, September 2011.

Table of contents

1 Introduction

Speech signal conveys many different types of information, including the words and language being spoken, speaker specific characteristics and emotions, background noise and transmission channel properties to name a few. The main task of *speaker recognition* is to extract and characterize speaker specific properties while minimizing the effect of the other information and later to recognize the user's identity based on a given speech sample.

Because of the physical properties of the vocal tract, larynx and other voice production organs, no two human voices sound identical. Voice as a biometric modality has its own advantages and disadvantages compared to traditional methods such as fingerprint or face recognition. For humans, speech is a natural way to communicate and therefore easy to use. In telephone-based systems voice is the only available biometric. Another advantage is that equipment required to collect speech sample is usually cheap and often readily available (like in telephone systems).

Speaker recognition is a broad area that also includes *speaker verification, speaker identification, speaker segmentation* and *speaker indexing*. In this thesis we focus on speaker verification and identification systems. Speaker verification system targets to verify whether a given speech sample belongs to a claimed user or not. Whereas the speaker identification tries to find the most probable user from the set of speaker models enrolled to the system which has produced an unknown speech sample.

1.1 MOTIVATION

Speaker recognition techniques have been studied for several decades and successful applications utilizing this technology are already available on the market. These include, for example, Nuance Verifier [Nuance], speaker verification products from Loquendo [Loquen] and voice authentication solutions from Bayometric [Bayom]. However, the majority of the existing solutions are focused on desktop environment or call centers where computationally demanding algorithms are running on efficient but expensive hardware. In mobile device environment, on the other hand, available computational resources are normally very limited and require special optimizations of existing algorithms. One modern approach to overcome resource limitations is to split the speaker recognition system into two parts, where data is only collected on the mobile device and sent over the network to the server which then runs the actual recognition. Most known examples of this approach are speech input on Google Android phones [ASDK] and Nuance Dragon dictation for Apple iPhone [Dragon]. This method, however, is not always preferable as it requires persistent network connection which might be unavailable or expensive for the actual user. In this thesis we focus on speaker recognition systems that run entirely on mobile or other resource limited device that does not require additional infrastructure to operate.

The speaker recognition topic has not received as much attention compared to automatic speech recognition (ASR) which has more potential applications. As a consequence, many techniques in speaker recognition have been borrowed from ASR even though the goals in these two tasks are quite opposite. In speech recognition, one attempts to eliminate speaker specific characteristics whereas the speech content is usually not important in speaker recognition. Good overview tutorials about speaker recognition can be found in [Bimbot04, Camp97, Furui97, Kinn10].

2.1 THESIS STRUCTURE

The rest of the thesis is organized as follows. In Chapter 2, we give general introduction into the area of speaker recognition. We discuss methods and algorithms that have been most successful in modern systems. In Chapter 3, we discuss optimization strategies and provide algorithms to improve computational performance of speaker recognition systems. The main focus of these optimizations is to allow speaker recognition applications to run on mobile or embedded devices. In Chapter 4, we give summary of the original publications of this thesis, and summarize their main results in Chapter 5. Finally, we draw conclusions and outline future research directions in Chapter 6. The original publications are attached at the end of this thesis.

2 *Speaker recognition*

There are two operational modes involved in a typical speaker recognition system: *enrollment* (Fig. 2.1) and *recognition* that, depending on a system type, can be *identification* or *verification* (Fig. 2.2). In the enrollment mode a system is trained to recognize a particular speaker. Based on a provided speech sample, a speaker model is created and stored in a speaker database.

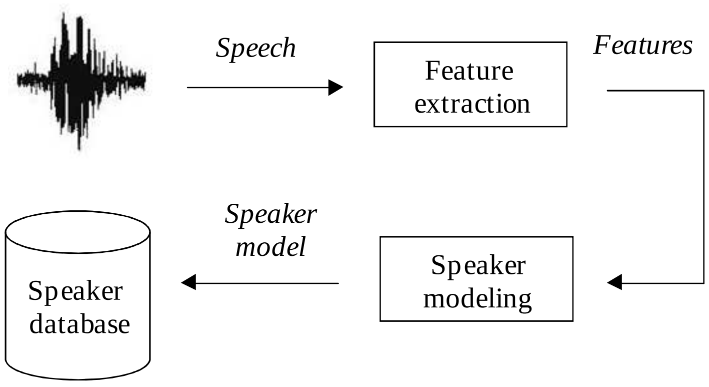

Figure 2.1 Enrollment mode

This model is later used to match a provided unknown speech sample and decide on its identity.

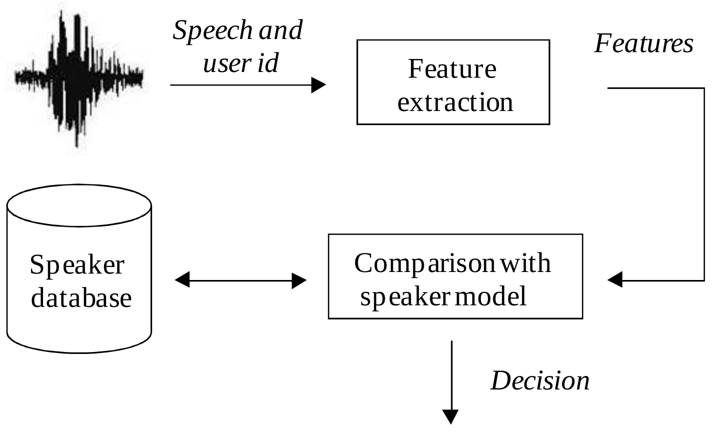

Figure 2.2 Verification mode

A typical recognition system is composed of two main components: *feature extraction* and *modeling*. The feature extraction component or *front-end* is responsible for transforming the raw audio data into a compact form so that speaker specific characteristics are emphasized and redundant or non-relevant information is removed. The modeling component or *back-end* aims at modeling unique speaker characteristics so that they can be stored in a system database and later used in recognition mode to judge incoming user claims.

To normalize the effect of non-speaker characteristics in a speaker verification system, a technique called *impostor modeling* is introduced [Reyn02]. There are two dominant approaches used to represent impostor model. The first is known as *cohort background set*, which is a collection of the other speaker models. The second approach known as the *universal background model* (UBM) is a large single speaker-independent model that has been trained on a large amount of data from different speakers [Reyn00].

2.1 FEATURE EXTRACTION

Feature extraction, or *speech parameterization,* is an important part of a speaker or speech recognition system that aims to convert the continuous speech pressure signal into a series of reasonably compressed vectors. Here, *features* are speaker specific characteristics that are present in a speech signal. The goal of the feature extraction step is to extract them from the signal while minimizing the effect of other less meaningful information such as background noise and the message itself. Ideally, features should have the following properties [Wolf72]:

- High between-speaker variability and low intra-speaker variability
- Easy to measure
- Stable over time
- Occur naturally and frequently in speech
- Change little from one speaking environment to another
- Not be susceptible to mimicry

Human speech production is driven by the excitation of the vocal folds due to the air flow expelled from the lungs. The produced sound is then modified by the properties of the vocal tract (oral cavity, nasal cavity and pharynx). From the *source-filter* theory of speech production [Deller00] we know that the resonance characteristics of the vocal tract can be estimated from the short-term spectral shape of the speech signal. Even though there are no exclusive speaker identity features, they are encoded via resonances (formants) and pitch harmonics [Deller00, Reyn02].

Ideally, speaker recognition systems should operate in different acoustic environments and transmission channels so that enrollment might be done at IT service desk and recognition over telephone network. However, since the spectrum is affected by environment and channel, feature normalization techniques are required for compensating the undesired effects. Usually this is achieved by different linear channel compensation techniques like short or long term cepstral mean subtraction [Reyn94, Reyn02].

A lot of research has been done in the speech parameterization area for speech recognition systems resulting in many different algorithms. However, little has been done for finding the best representation of precisely speaker specific characteristics that minimizes the effect of commonalities present in speech (e.g. same word pronounced by different users will have many characteristics in common). Even worse, speech recognition methods are, in general, designed to minimize inter-speaker variability and thus removing speaker specific information. Yet, many of these methods have been successfully utilized also in speaker recognition by using different normalization methods like background modeling.

We categorize different speech parameterization methods into three broad categories (1) *short-term features*, (2) *prosodic features* and (3) *high-level features*. We review these in the following sub-sections.

2.1.1 Short-terms features

The speech signal, in general, is seen as a quasi-stationary or slowly varying signal [Deller00]. In other words, speech signal is assumed to be stationary over relatively short intervals. This idea has motivated a series of methods that share the same main principle: the signal is divided into short segments (typically 20-30 ms) that usually overlap by about 30%. These segments are called *frames* and a set of coefficients calculated from a single frame forms a *feature vector*.

To avoid undesired effects due to splitting continuous signal into short segments, each frame is usually first preprocessed. A common step is to apply *window function* with the purpose to minimize effects of abrupt changes at the frame ends and to suppress the sidelobe leakage that results from the convolution of the signal spectrum [Deller00]. The most popular selection for window function is Hamming function. In addition, each frame might be *pre-emphasized* to boost higher frequency components which intensity would be otherwise low due to the downward sloping spectrum of the glottal voice source [Deller00].

Most popular methods for short-term feature extraction include *mel-frequency cepstral coefficients* (MFCCs) [Davis80, Deller00], *linear prediction cepstral coefficients* (LPCCs) [Camp97, Makh75] and *perceptual linear prediction* (PLP) *cepstral coefficients* [Herm90]. A thorough evaluation of these methods from a recognition performance point of view is available in [Reyn94].

MFCCs are by far the most popular features used both in speech and speaker recognition. This is due to their well defined theoretical background and good practical performance. Mel-frequency warping of the spectrum gives emphasis on low frequencies that are more important for speech perception by humans [Deller00]. MFCC feature extraction technique (Fig 2.3) consists of the following steps. First, the signal is windowed. Its spectrum is computed using Fourier transform (FFT). The spectrum is then warped on Mel-scale by averaging out FFT spectral magnitudes equi-spaced on the Mel-scale. In terms of linear frequency scale, this means that the lower frequencies are processed with filters having narrower bandwidths to give higher spectral resolution to these frequencies. Final coefficients are computed by taking inverse Fourier transform.

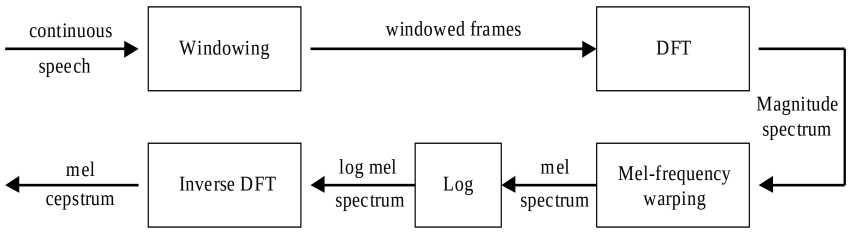

Figure 2.3 Computing MFCCs

Usually MFCCs are computed from the FFT spectrum but this is not always the case. The FFT spectrum is subject to various degradations, such as additive noise and fundamental frequency variations. Replacing FFT with alternative spectrum estimation may help to tackle those issues [Saeidi10].

Each MFFC vector is extracted independently from the other short-term frames and, consequently, information on their ordering is lost, meaning that feature trajectories are not taken into account. A common technique to capture some contextual information is to include estimates of the first and second order time derivatives – the *delta* and *delta-delta* features – to the cepstral feature vector. The delta coefficients are usually computed via linear regression:

$$d_t = \frac{\sum_{k=1}^{K} k \cdot \left(f_{t+k} - f_{t-k}\right)}{2\sum_{k=1}^{K} k^2}, \tag{2.1}$$

where f and d correspond to the static and delta (dynamic) coefficients respectively, K is the number of surrounding frames and t is feature vector for which the delta coefficients are being computed for. Delta-delta (acceleration) coefficients are computed in the same way but over the delta (first derivative) coefficients. The derivatives are estimated over a window of frames surrounding current frame (typically 7 frames for delta and 5 for delta-delta). Delta coefficients are normally appended to the end of the feature vector itself [Furui81, Huang01].

2.1.2 Prosodic features

In linguistics, *prosody* refers to various features of the speaker like speaking rhythm, stress, intonation patterns, emotional state of the speaker and other elements of the language that may not be encoded by grammar. Prosodic features are also referred to as *suprasegmental* features as they do not correspond to single phoneme but rather span over long periods of speech such as syllables, words and phrases. Even though modeling these features for speaker recognition systems is a challenging task, recent studies indicate that prosody features improve speaker verification system performance [Kockm11].

By far the most important prosodic feature is the *fundamental frequency* (also called F0) which is defined as the rate of vibration of the vocal folds during voiced speech segments [Hess83]. F0 has been used in speaker recognition system already in 1972 [Atal72]. The fundamental frequency value depends on the mass and size of the vocal folds [Titze94] and therefore it contains information that is expected to be independent of the speech content. Therefore, combing it with spectral features should improve overall system accuracy. For example, it has been found in [Kinn05] that using F0 related features alone shows poor recognition accuracy but when used in addition to spectral features recognition accuracy is improved, especially in noisy conditions.

The advantage of F0 is that it can be reliably extracted even from noisy speech [Hess83, Iwano04]. A comparison of F0 estimation methods can be found in [Chev01]. However, as F0 is a one-dimensional feature, it is not expected to be very discriminative in a speaker recognition system. These aspects have been studied in [Kinn05].

Other prosodic features that have been used for speaker recognition systems include *duration* features (pause statistics, phone duration), *energy* features (like energy distribution) and *speaking rate* among others. These features were extensively studied in [Shrib05] where it was found that F0 related features are still the best in terms of recognition accuracy.

2.1.3 High-level features

Human voice characteristics differ not only due to physical properties of the vocal tract but due to speaking style and lexicon as well. Listeners can distinguish between familiar people much better than between those they have not ever heard. This is due to certain *idiosyncrasies* present in speech that a human is able to catch.

The work on high-level features was initiated in [Dodd01] where the authors explored idiolectal differences by using N-gram language models for modeling co-occurrences of words and using this information as speaker specific characteristics. Another approach was studied in [Camp04] where the authors used frequency analysis of phone sequences to model speaker characteristics.

High-level features are not yet widely used in modern speaker recognition systems. However, with advances in speech recognition it is now possible to utilize efficient phone and word recognizers in speaker recognition area as well. An overview of recent advances in this area is available in [Shrib05, Kinn10].

2.1.4 Channel compensation

Modern speaker recognition systems strive to operate reliably across different acoustic conditions. There might be different equipment used at enrollment and recognition steps. In addition to background noise, transmission channel bandlimiting and spectral shaping greatly affect the system accuracy. Therefore, different *channel compensation* techniques are used for tackling those challenges. According to [Reyn94], short-term spectral features suffer from adverse acoustic conditions and thus perform poorly without channel compensation. Other feature types are expected to be less sensitive to channel properties.

From the signal processing theory we know that convolutive distortion in signal domain becomes additive in log-spectral domain. The simplest compensation technique is therefore to subtract the mean value of each feature over the entire speech sample. This technique is called *cepstral mean subtraction* (CMS) or *cepstral mean normalization* (CMN) [Atal74, Furui81]. In addition, the variances of the features can also be normalized by dividing each feature by its standard deviation. However, using mean value over the entire utterance is computationally not efficient as features are not available for processing before the entire utterance has been spoken. Channel characteristics may also change over the time of speaking. A *segmental* feature normalization approach was proposed in [Viikki98] where mean and variance of the features are updated over a sliding window usually of 3 to 5 seconds in duration.

Another important channel compensation technique, known as *RASTA filtering,* has been proposed in [Herm94]. The main idea of this method is to band-pass filter each feature in the cepstral domain and remove modulations that are out of typical speech signals. RASTA processing alone helps to improve system performance but it is not as efficient as other more advanced techniques [Reyn94]. However, its combinations with other methods have been extensively used.

Channel compensation is a very important step in any practical speaker recognition system and it is therefore still an active topic in research. There are many other promising methods found in the literature such as *feature warping* [Pele75], *feature mapping* [Reyn03] and different combinations [Burget07, Kinn10].

2.2 SPEAKER MODELING

Speaker modeling component or *back-end* is a part of the speaker recognition system that aims to create a parametric representation of the speaker's voice characteristics. This model is stored into the speaker database and later used to verify identity claims. Feature extraction component provides an input for modeling both in enrollment and recognition modes. During enrollment, a speaker model is trained from the input feature vectors, whereas in recognition, an input sample is matched against the stored speaker model(s). In a speaker verification system a decision is made by comparing the match score against a decision threshold. The match score is usually further normalized using other speakers' models to reduce the effects of speaker and environmental variations. In speaker identification system decision is made based on the match scores of all models. Usually, the speaker model with the best score is selected as the identification output.

Desirable attributes of a speaker modeling technique are [Reyn02]:

- Well-defined mathematical background that allows extensions and improvements
- Able to match new data not present at the enrollment step, i.e., does not over-fit to the enrollment data
- Practical representation both in storage size and computational performance

In speaker recognition research, modeling techniques are traditionally divided into *template models* and *stochastic models* [Camp97]. These two differ in the way pattern matching is carried out. In template models, it is assumed that feature vectors are inexact replicas of the template and, therefore, the training and test vectors are directly compared against each other by measuring the distance between them. Examples of these techniques are *dynamic time warping* (DWT) [Soong87] and *vector quantization* (VQ) [Furui81]. In stochastic models, in turn, speaker voice is assumed to be a probabilistic source with a fixed probability density function. Training vectors are used for estimating the parameters of this function. In the recognition step, the conditional probability or likelihood of the test vectors, given the model, is evaluated. The *Gaussian mixture model* (GMM) [Reyn95] and *hidden Markov model* (HMM) [Naik89] are the most well-known examples of stochastic models.

However, in recent years several new modeling techniques have evolved and started to get significant attention by the speaker recognition community. These are so called *discriminative models* that model the *boundaries* between speakers as opposed to *generative models* (template and stochastic models described above) that estimate the feature distribution within speakers [Kinn10]. Sound examples of discriminative modeling methods are *artificial neural networks* (ANNs) [Farrell94] and *support vector machines* (SVMs) [Camp06a].

In the following sub-sections we review the commonly used speaker modeling techniques, namely vector quantization and Gaussian mixture models. We will also pay special attention to match score normalization methods as they have become a crucial part of any modern speaker verification system and give short overview of emerging new modeling methods.

2.2.1 Vector quantization

Vector quantization (VQ) modeling was first introduced as a data compression technique [Ger91] and later used in speaker recognition as well [Burton87, Furui91]. Speaker model in VQ is created by partitioning the training data set into a finite, usually a predefined number of non-overlapping regions that are represented by their mean vectors or *centroids*. A set of such centroids is called a *codebook,* which represents a model of the training data. Partitioning process is called *clustering* and is performed by minimizing the average distortion between centroids and training vectors over the whole training data. This process is schematically represented in Figure 2.4.

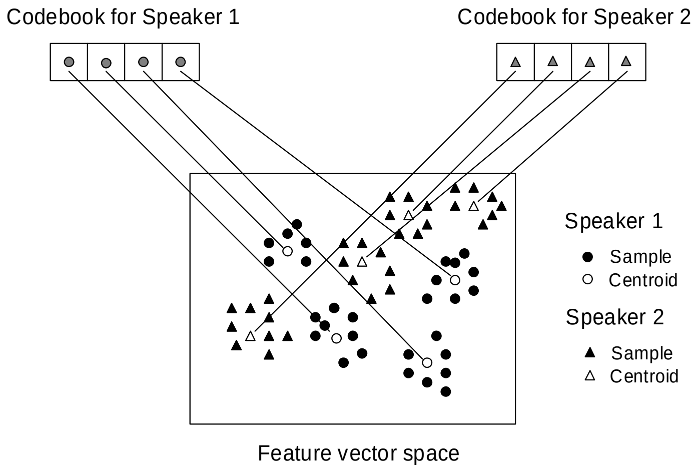

Figure 2.4 Vector quantization of two speakers

Several algorithms exist for codebook generation. The following is a list of several example methods [Kinn11]:

- *Generalized Lloyd algorithm (GLA)* [Linde80]
- *Self-organizing maps (SOM)* [Nasr88]
- *Pairwise nearest neighbor (PNN)* [Equitz94]
- *Iterative splitting technique (SPLIT)* [Fränti97]

- *Randomized local search (RLS)* [Fränti00]

In a recent study [Kinn11], the authors compare these algorithms for codebook generation in VQ-based speaker recognition. According to the authors, the choice of the method is not as important as the codebook size. Theoretically, it is possible to use all the training vectors as a model directly without any clustering but it is not efficient from a computational point of view and leads to over fitting for the training data.

Matching in VQ is done by combining minimum distances from test vectors to the codebook centroids. There are several techniques for computing the match score, with *mean square error* (MSE) being the most popular. It is computed as the sum of the squared distances between the vector and nearest centroid divided by the number of test vectors,

$$MSE(X,C) = \frac{1}{N}\sum_{i=1}^{N} \min_j (d(\mathbf{x}_i, \mathbf{c}_j)^2) \quad , \quad (2.2)$$

where X is a set of N extracted feature vectors, C is a speaker codebook, $\boldsymbol{x}_i$ are the feature vectors of the test utterance, $\boldsymbol{c}_j$ are codebook centroids and d is a vector distance function. Even though MSE is a very common method to compute the match score for VQ-based system, a search for a better metric is still an ongoing topic in speaker recognition [Hanilci11].

Although vector quantization is a relatively simple and lightweight technique that is well suited for practical applications on embedded devices it also provides competitive accuracy compared to other techniques [**P3**]. VQ is a natural choice while selecting a modeling method for application that is supposed to run on a device with limited hardware capabilities. We will discuss optimization strategies for VQ in more detail in Chapter 3.

2.2.2 Gaussian mixture model

The most popular modeling method in text-independent speaker recognition is *Gaussian mixture model* (GMM) [Reyn95]. While more advanced likelihood estimation methods like the *hidden Markov model* (HMM) have also been used they have not proved significant improvement over GMM [Reyn02, Bimbot04] since that HMM cannot be easily applied to text-independent speaker recognition. In fact, GMM can be considered to be single state hidden Markov Model. It can also be seen as an improved version of the VQ model with overlapping clusters [Kinn10].

In GMM, the speaker model consists of a finite mixture of multivariate Gaussian components. The model characteristics are defined by its *Gaussian mixture density* function which is a weighted sum of component densities,

$$p(\mathbf{x}) = \sum_{i=1}^{M} p_i \cdot b_i(\mathbf{x}), \tag{2.3}$$

where M is the number of Gaussian components, $\boldsymbol{x}$ is a multi-dimensional feature vector, $b_i(\boldsymbol{x})$ are the components densities and p_i are the mixture weights or prior probabilities. To ensure that the mixture is a proper density mixture, the weights should sum up to unity. Each component density function is given by the following equation:

$$b_i(\mathbf{x}) = \frac{1}{(2\cdot\pi)^{\frac{N}{2}} \cdot \left|\Sigma_i\right|^{\frac{1}{2}}} \cdot \exp\left\{-\frac{1}{2}\cdot(\mathbf{x}-\boldsymbol{\mu}_i)' \cdot \Sigma_i^{-1} \cdot (\mathbf{x}-\boldsymbol{\mu}_i)\right\}, \tag{2.4}$$

where N is the dimensionality of the feature vector $\boldsymbol{x}$, $\boldsymbol{\mu}_i$ is the mean vector and Σ_i is the covariance matrix for *i-th* component [Reyn95].

Each GMM speaker model is parameterized by the mean vectors, covariance matrices and mixture weights from all the component densities. Estimation of these parameters is not possible in closed form, and is computationally demanding if full covariance matrices are used. However, complexity is greatly reduced if one uses diagonal covariance matrices instead. One popular algorithm for estimating the GMM parameters is *expectation-maximization* (EM) [Bishop06]. In EM, an initial solution is required which is iteratively improved. Iterations are stopped when there is no significant improvement in the model likelihood for training data. Initial solution can be selected, for example, by clustering the training data set. Another modeling approach is to train one large universal GMM model from large amount of speech and estimate individual speaker models from it by *maximum a posteriori* (MAP) adaptation technique [Kinn10, Reyn00].

Feature vectors are assumed to be independent and match score in GMM is therefore computed simply by multiplying the individual vector likelihoods. To avoid numerical problems with very low likelihoods, usually a log-likelihood is used in practical implementations that result in the match score being a sum of the log-likelihoods of individual vectors.

Gaussian mixture modeling as such is a computationally demanding task. It involves several costly operations like square roots, exponents and logarithms. Many of these operations can be pre-computed at the enrollment step and using GMM model quantization computational load can therefore be greatly reduced [**P6**]. We will return to GMM optimization methods in Chapter 3.

2.2.3 Recent advances

In the previous subsections we have presented two major modeling techniques that dominate in speaker recognition systems. However, in recent years there has been significant progress in this area and several modern methods have evolved. The practical properties of these methods from mobile device environment point of view are yet to be seen in future but, for completeness, we present a short overview of the most promising techniques.

Speaker modeling using the EM algorithm in GMM-based systems requires significant amount of training data which is not always available. To tackle this data insufficiency problem, several authors have proposed training single large model from a development set and later adapting speaker specific models from it using maximum *a posteriori* (MAP) adaptation [Kinn10, Reyn00]. For simplicity, only the mean vectors are adapted. Stacking these adapted mean vectors together leads to so-called GMM *supervectors* [Camp06a]. Such vectors can also arise, for example, from polynomially expended vectors [Camp06b].

Several new speaker modeling methods have evolved based on the supervector concept. Using *support vector machines* (SVM) back-end to classify these supervectors has proven to be an effective speaker recognition method [Camp02, Camp06b]. The main idea of SVM is to perform a mapping from an input space to SVM space where linear classification techniques can be applied. Such matching function is a key design element of any SVM-based system [Camp06b]. For more details of SVM system based on supervector concept, we refer to [Camp06b].

Another promising technique that has been recently proposed is *joint actor analysis* (JFA) [Kenny07, Matr07]. JFA takes advantage of the correlations between Gaussians during speaker modeling to decompose the speaker model into three components: a speaker and session-independent component, a speaker-dependent component and a session-dependent component [Kenny07, Matr07]. However, as the original authors state, this method is computationally demanding and requires a well-balanced training set recorded under a variety of channel conditions [Kenny07]. The idea of JFA has later been extended in [Dehak09] where the authors proposed a novel method that, instead of modeling between-speaker and within-speaker variability in a high dimensional supervectors space, finds a low-dimensional supervector subspace that represents both the channel and speaker variabilities [Dehak09]. Accordingly, this space has been named *total variability* space [Dehak09], which is also known as *i-vector* method [Sen10] and f*ront-end factor analysis* [Dehak11].

2.2.4 Match score normalization

The main task of speaker verification system is to make a decision whether the speaker is the person he or she claims to be based on a given speech sample. In simple cases, a match score can be tested against a predefined threshold. However, such an approach is not reliable in practice since speaker modeling methods do not produce probabilities but, rather, a biased match score that depends on various conditions, such as channel, environment and speaking style. To tackle this problem, *match score normalization* has been introduced [Auck00, Reyn02].

In modern systems, the most common method for making the decision is to compute the likelihood that the input speech sample has been produced by the claimed speaker and compare it with the likelihood that it has *not* been produced by that speaker (so-called *impostor score*). In other words, given the claimed speaker identity, S, and input speech sample, Y, the verification task is a statistical *hypothesis test* between:

H_0: Y originates from the hypothesized speaker S

and

H_1: Y does **not** originate from the hypothesized speaker S.

Assuming that the likelihood functions for both hypotheses are known, the optimal decision in Bayes sense is a likelihood ratio test:

$$\frac{p(Y|H_0)}{p(Y|H_1)} \begin{cases} \geq \theta & accept\ H_0 \\ < \theta & reject\ H_0 \end{cases} \tag{2.5}$$

where $p(Y|H_i)$, i=0,1, are the likelihood functions for the two hypotheses H_i evaluated on speech segment Y, and θ is the decision threshold [Reyn00].

Estimating the *null hypothesis* likelihood $p(Y|H_0)$ is usually straightforward and is based on the speech sample match score against the claimant's model. However, estimating the *alternative hypothesis* likelihood $p(Y|H_1)$ is significantly harder [**P2**]. There are two dominant approaches in speaker verification, *world* or *universal background model* (UBM) and *cohort set normalization* [Auck00, Reyn02].

The world model approach uses a single speaker independent model trained from a large amount of speech data from a variety of speakers. The idea of this method is to model all the possible speakers and speaking contexts of the "*world*" and therefore it represents a general speech model. Match score normalization in this method is accomplished by a likelihood ratio test between claimant and world models likelihood's.

Cohort set normalization or modeling, on the other hand, uses a collection of other speakers, either enrolled to the system or coming from some other set, to estimate alternative hypothesis likelihood. Individual scores from cohort models are obtained and combined usually by averaging or selecting the maximum.

There is no preferred method in the speaker verification community as both methods have performed well in different studies [Bimbot00, Reyn02, **P2**]. The advantage of world model approach is that it is simpler as only one model has to be trained and scored [Reyn02]. However, the cohort approach provides a possibility for individual selection of impostor models for any enrolled speaker and therefore decreases the false acceptance rate making the overall system more secure [**P2**].

3 *Optimization techniques for mobile devices*

By *mobile device* in this work we refer to a generic pocket size handheld device that is battery powered. The hardware design for such a device involves many different factors with power consumption, component size and price being the most important. These limitations lead to significantly less powerful hardware that is available for speaker recognition system designer. On the other hand, speaker recognition, as any other pattern recognition technique, requires a lot of complex mathematical computations that are very demanding for the system resources. The challenge for the system designer here is how to reduce the amount of computations and required memory size while retaining recognition accuracy and usability on acceptable levels.

Before doing any optimizations, the so called *"80-20 rule"* (also known as the *Pareto principle*) has to be considered. The rule states that 80% of device resource like CPU time or memory is used by 20% of the system. While not being exactly true, it stresses the importance of finding the most time consuming places inside the system – the *bottlenecks* - and spend the most effort on optimizing them. These places can be reliably found only while running benchmarking tests on the target hardware. The well known author of algorithm design books Donald Knuth has stated: *"We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil. Yet we should not pass up our opportunities in that critical 3%. A good programmer will not be lulled into complacency by such reasoning, he will be wise to look carefully at the critical code; but only after that code has been identified"* [Knuth74].

From the author's personal experience, speaker recognition system front-end (feature extraction), if implemented right, can be executed in 3 to 4 times real-time, on average, in mobile device. In other words, 1 second of speech can be processed in 0.3-0.2 seconds. This is because the front-end utilizes techniques that have been developed for decades in digital signal processing community. Speaker model matching, on the other hand, is a much less studied problem and therefore requires more attention for seeking the bottlenecks. Performance of speaker enrollment into speaker recognition system is usually not that important either, as it can be done once and delays are normally more tolerated there. Speaker model adaption can also be done in the background and therefore it does not require a very efficient execution.

The best optimization is, in fact, *no computation at all*. While sounding absurd there are many places in a speaker recognition system where this can be achieved. Time consuming operations can be analyzed in real-time and decision can be made whether they have to be executed or not. Non speech segments removal in the front-end is the most obvious example of such strategy. Removing useless data at an early stage saves a significant amount of computations later. As a variation of this method, the relevance of input data can be analyzed in speaker model components to prune out the majority of them early and complete final precise computations for only a few [**P1**, **P6**]. Sometimes there are operations in algorithms that do not change much or have only a few possible results during the execution. Such places should be analyzed and replaced by pre-computed results.

One novel approach that is becoming more and more popular in modern systems is to split operations into two groups from which one runs on the device itself (e.g., front-end) and the other is executed on a remote server (e.g., back-end) that is connected to the device over a network. This approach has its own advantages and disadvantages that are beyond of the topic of this thesis. An example system based on such an approach has been reported in [Chow10].

Even though there are many limitations imposed by the mobile device, speaker recognition systems that are running entirely on an embedded device are starting to appear [Tyd07, Rao10, Roy11]. In the rest of this chapter, we will first discuss generic strategies to attack mobile device limitations, including the absence of a floating point unit. We also give a few guidelines on how to efficiently implement algorithms with focus on low-resource devices. After that we review methods for optimizing different parts of a typical speaker recognition system paying more attention to algorithm design rather than to its implementation.

3.1 MOBILE DEVICE LIMITATIONS

While designing a speaker recognition system for a mobile device many factors have to be addressed. At the forefront are certainly the device's limited hardware resources like CPU speed and a significantly lower amount of available memory both for algorithm execution and model storage. But there are also other less frequently considered limitations like absence of a floating point co-processor or signal pre-processing by device audio recording hardware. The mobile device is assumed to be operated in much more noisier and varying acoustic conditions compared to traditional desktop computer systems. Any mobile device is truly mobile only when it is powered by a battery. Therefore, its life time has to be preserved as much as possible for convenient usage, and as a result, all unnecessary computations have to be minimized. Development environment for mobile devices might be clumsy and programming languages may have limitations that have to be addressed during algorithm design or adaptation. In the following sub-sections, we will discuss generic strategies for designing algorithms focusing on mobile device applications.

3.1.1 Optimizing for device hardware

Any theoretically efficient algorithm can be ruined by an inefficient implementation. The same also concerns algorithms that are designed without hardware limitations in mind. Certain operations like memory access may look constant when the method is theoretically analyzed but, in reality, its execution time may vary significantly depending on different conditions and details of the target hardware design. In this sub-section, we give an overview of the most common issues that may be faced during algorithm optimization for mobile devices.

Most modern hardware designs for mobile devices have a *layered* memory structure in regards to access speed. The fastest memory is the most expensive and therefore there is less of such memory available. Processor register is the fastest memory type and the only type that a CPU can access to perform any operation. When there is not enough space in the registers, data is saved to slower memory that in turn will save its data to even slower memory when full and so on. This so called *memory pyramid* is represented in Figure 3.1.

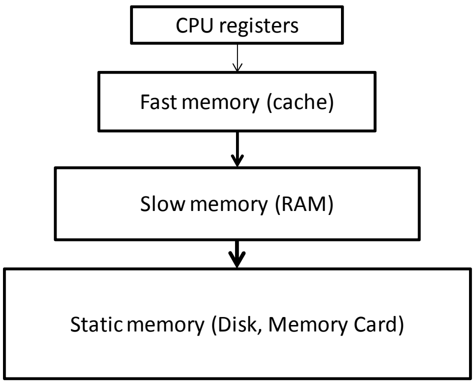

Figure 3.1 Memory pyramid

When some data is needed by CPU it has to be loaded to registers first. Loading process works in the opposite way so that data from one layer can be loaded only to the higher layer. For example, when some data is needed that is stored on disk memory it has to be loaded first to RAM, then to memory cache and only after that it can be loaded to processor registers. For efficiency reasons, data is never loaded from one layer to another as single byte but in data blocks called *cache lines* whose size varies depending on architecture but is usually smaller for higher levels [Henn06, Hoxey96].

This complicated process is transparent to the program running on a mobile device. However, if these issues are not taken into account while designing and implementing an algorithm, its performance may not be as expected. The main design principle to address these concerns is to utilize so called *data locality* or *locality of reference* principle when all data is processed in small chunks that are stored in a contiguous buffer. By designing algorithms in this way data transfer between memory layers will be minimized. The same concerns the application binary as it is loaded to memory in the same way. The less branches and variables there are in the algorithm implementation the faster it will work in regard of memory access speed.

Data structures used in the algorithm design should also be reconsidered from the memory access performance point of view. Linear data structures utilize a better data locality principle, and therefore, algorithms may be modified to use them. The data model format should be designed so that it may be loaded as one data block and used immediately without parsing it first. This may allow storing data models into *read only memory (ROM)* so those models will not be required to be loaded to the main memory at all.

Most hardware designs of mobile devices already contain dedicated units for signal processing and coding. As speaker recognition includes many signal processing algorithms these hardware modules can be utilized to share computation load of signal processing operations. Moreover, speech coding parameters can be directly utilized in speaker recognition. Work in this direction has been reported for speech recognition in [Huer98] and speaker recognition in [Quat00].

An ideal algorithm would be just a set of lookup tables combined with decision logic. In practice this is not always possible but such ideas have been applied to at least HMM-based speech recognition [**P6**, Vasila00], and they can be applied to speaker recognition as well. We have presented here only a few main principles of optimization directions, and recommend the studies of [Henn06, Hoxey96] for an interested reader.

3.1.2 Fixed point arithmetic

Even nowadays, many modern mobile devices still do not have a *floating point unit* (FPU) included due to their higher price and power consumption because of the larger silicon area required. This is in strong contrast with traditional desktop computer environment where FPUs are integrated by all major manufactures. In mobile devices, floating point is often emulated by software libraries or operating system. But without hardware support for floating point operations, this executes significantly slower and therefore, converting algorithms to run in fixed point is highly desired.

Fixed point number representation has a fixed number of bits reserved for the integer and fractional parts of a number. The value is basically an integer that is scaled by a predefined factor. For example, 12.345 may be represented as integer 12345 with scaling factor 1/1000. The scaling factor is fixed and does not change during computation, in contrast to floating point representations where *radix point* (also called decimal point) position may vary (hence the name - floating) depending on the value it holds. The main advantage of floating point representation over fixed point is the much wider range of values it can hold. For example, if 5 digits were reserved for the integer part and 2 for the fractional part, such number can hold values like 12345.12 or 0.01 but not any bigger or more accurate. Floating point representation, on the other hand, allows storing for example values like 1234567 or 0.12345. Therefore, it is important to analyze the data range before converting the algorithm to fixed point. If too few bits are allocated for the integer parts, the algorithm may overflow, but on the other hand, if too few bits allocated for the fractional part it may lose precision.

Arithmetic operations in fixed point are slightly different from the conventional integer operations. While addition and subtraction can be done in the normal way, care should be taken that numbers have the same scale. Multiplication and division are more complicated. When two fixed point integers are multiplied the resulting scale doubles and it has to be scaled back to the original precision. Particular care should be taken that multiplication does not overflow as there is only a fixed amount of bits for storing the integer part but number of bits for the fractional part in the result will be twice as big. To give an example, let us multiply 1.23 by 4.56 (stored as 123 and 456, respectively, with scale factor 1/100). The result will be an integer value 56088 with scale factor 1/10000, or 5.6088. To get back to the original precision, we need to scale it back to 5.60 with scale factor 1/100. To avoid overflow, a value can be scaled down to a lower fractional part before multiplication but this would result in loss of precision. Division works the opposite way as scale factor in result will be subtraction of dividend and divisor scale factors. This also means that if the dividend and divisor have the same scaling factor, the fractional part will be lost completely in the result. To attack this problem, the scaling factor for the dividend should be increased before division so as to retain the precision of the fractional part of the result.

Algorithm conversion to fixed point arithmetic is not an easy task. In our work, we have done this for a speaker recognition system in [**P5**] where the most tedious part was implementation of the *fast Fourier transform* (FFT). FFT turned out to be the most critical part of the entire system regarding recognition accuracy whereas errors in the other components such as speaker matching had no significant effect on the result. From our experience, careful numerical analysis is a crucial part of successful fixed point algorithm implementation [**P5**]. For more discussion on fixed point arithmetic error analysis in speaker recognition we refer to [Tyd07].

3.2 OPTIMIZING FRONT-END

Most of the computation time in a typical speaker recognition system is spent in matching the incoming feature vectors to speaker models. With a careful implementation of the front-end, the matching step is significantly more computationally expensive than feature extraction [**P3,** Karpov03]. Standard spectral feature extraction methods are also well studied over the past decades and there is not much space for further speed improvement. Some optimizations can still be done in the front-end to improve the overall system performance. Reducing the number of feature vectors is a simple example. In this sub-section, we will review the two most common feature vector reduction techniques, *voice activity detection* and *quantization of input samples*. Time complexity analysis of the most typical feature extraction techniques and their experimental performance evaluations can be found in [Karpov03].

3.2.1 Voice activity detection

Speech signal does not always contain relevant information for speaker recognition system but also includes silent or noisy regions where the speaker is not saying anything. Such segments should be discarded from the input signal for improving overall system performance. Voice activity detection (VAD) is a technique that aims at partitioning a given audio sample to speech and non-speech segments. Occasionally, it is also called *silence removal,* but VAD may also remove non-silent noise regions. While the problem of detecting and removing non-useful audio segments is a relatively long studied task, it is still unsolved in adverse acoustic conditions, especially at very low signal-to-noise ratios (SNRs), and there is still room for new research [Furui05]. VAD plays an important role in signal processing applications, especially in telecommunications where it can save a considerable amount of traffic and energy [Kond69].

Simple methods for voice activity detection make decision based on the measured value of signal characteristics such as [Marks88]:

- relative energy level,
- first autocorrelation coefficient,
- first LPC linear predictor coefficient,
- first mel-frequency cepstrum coefficient,
- normalized prediction error.

Some methods use output from feature extraction algorithms [Haigh93, Martin01] and some operate on non-processed signal [Buril00]. More advanced methods involve modeling of background noise to distinguish whether signal frame contains speech or not. The model is updated real-time to reflect changing background noise characteristics.

One representative example of such approaches is the *long-term spectral divergence* (LTSD) method [Ram04]. It compares the long-term spectral envelope to the average noise spectrum. The decision logic is adapted to the measured noise energy and hangover scheme is activated in low signal-to-noise ratio regions. While LTSD works well in noisy conditions, its main drawback is the initialization of the algorithm which requires a sample of the background noise before it can start operating. If, for some reason, the noise model was initialized with signal that contains speech this method will perform poorly. Also hangover scheme is less important in speaker recognition because speech regions do not have to be contiguous as is required by speech recognition.

Many VAD algorithms can be found in telecommunication industry, including standards such as ITU G729B [ITU96], ETSI AMR option 1 and 2 [ETSI99], ETSI AFE [ETSI00] and emerging Silk codec [Silk09] used in the popular Skype communication program. We have compared these algorithms in [**P7**] and found that all of them suffer from adverse acoustic conditions. VAD is certainly not a solved problem and more research is required in this area before acceptable methods will be found.

3.2.2 Quantization of input samples

Speech signal is slowly varying so that the feature vectors extracted from adjacent frames are highly correlated. The idea of *input sample quantization* or *pre-quantization* (PQ) is to replace a group of feature vectors with only a few representatives that can be matched against speaker model, and in this way to reduce the computation time needed for matching. This process is schematically represented in Figure 3.2.

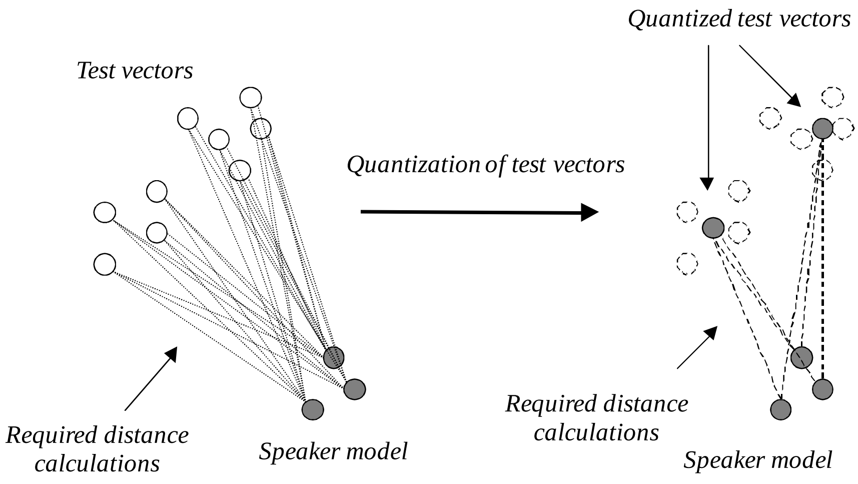
Quantized test vectors
Test vectors
Quantization of test vectors
Required distance calculations
Speaker model
Required distance calculations
Speaker model

Figure 3.2 Quantization of input samples

This idea has been originally introduced in [McLa99] where the authors reported a compression ratio of the input vectors at 20:1 without compromising system accuracy. In [**P3**] we proposed three novel quantization techniques and compared the results to the decimation method of [McLa99]. The compared variants in [**P3**] include the following methods:

- random subsampling,
- averaging,
- decimation,
- clustering-based PQ.

In *random subsampling* and *averaging*, we replace a consecutive sequence of feature vectors with only one randomly selected, or average vector of the sequence, respectively. In the *decimation* method, we simply take every M^{th} vector from the feature vector stream (here, M is a control parameter of the algorithm). In the *clustering-based* method, we partition the input sequence using K-means clustering algorithm. In [**P3**], we have evaluated different parameters for quantization on different data sets and found that all of them work quite well, while the clustering-based method performs the best. Quantization of input samples gives approximately 50% time reduction in the matching step with only minor (less than 1%) increase in error rate [**P3**].

Quantization of feature vectors has also been applied to speech recognition domain. In our system presented in [**P6**] we combined it with quantization of HMM model. In this approach, we can construct pre-computed tables and use them to calculate distances from feature vectors to the model mixtures just by a few table look-ups and additions. We will review this important optimization technique in more detail in the following sub-section.

3.3 OPTIMIZING BACK-END

Speaker recognition back-end is usually the most time consuming part of the recognizer. There are different speaker modeling techniques such as vector quantization and Gaussian mixture modeling and different methods for score normalization such as universal background modeling and cohort normalization. All these methods include a large number of distance computations to score the input feature vectors against speaker models. If the input is long and the speaker model has many parameters this will take a significant amount of time. Therefore, it is crucial to have optimized back-end for speaker recognition when targeted to run on a mobile device.

There have been a large number of methods proposed for speeding up back-end performance in speaker and speech recognition. The majority of them target either reducing or speeding up distance computations (scoring) between feature vectors and the acoustic or speaker models. As Gaussian mixture modeling (GMM) and vector quantization (VQ) are the most common modeling methods we will focus on them.

For GMM-based systems that utilize universal background model efficient scoring method was proposed in [Reyn00]. In the GMM-UBM approach, a single background model is trained first from a large amount of data from different speakers. Individual speaker models are later added by adapting the large background model. During the matching step, in the adapted GMMs, only those components are scored that are in the UBM top-list. This way one avoids most of the Gaussian density computations. Usually one scores only top-5 Gaussians, which leads to significant speed-ups without degradation in accuracy [Reyn00]. For GMM scoring there have also been various optimizations based on *hierarchical models* [Xiang03, Liu02, Sun03]. The authors in [Xiang03] reported a speed-up factor of 17:1 with a 5% relative increase in equal error rate. Similar approach was proposed in [Auck01] by using UBM-like *hash model,* which gained a speed-up factor of about 10:1 with a minor degradation in system accuracy.

An efficient scoring algorithm has been proposed in [Pellom98] with a reported speed-up factor of 6:1 relative to the baseline beam search. In [Tyd07] authors proposed GMM-based system targeted at embedded devices. Another efficient method for scoring Gaussian mixtures has been proposed in [Vasila00] that substantially reduces both memory consumption and computational load while retaining high recognition accuracy. Combined with feature quantization it leads to an efficient mixture scoring method [Vasila04]. Although this method was introduced for speech recognition, in principle it can also be applied to GMM-based speaker recognition as distance calculations are required in both tasks. We have used the same technique in [**P6**] and will review this method in more detail later.

Vector quantization optimization has received less attention than GMM mainly because GMM is already quite simple and efficient. However, we have found that VQ provides comparable performance, in terms of identification accuracy, to GMM [**P3**] and is even simpler to implement and runs efficiently in a mobile device. We have thoroughly investigated improvement areas for VQ based matching in speaker identification and verification [**P1**, **P2**, **P3**] and found several promising techniques such as *metric space indexing* [Chav01, Uhlm91] and *speaker pruning*.

In the following sections we will review the most promising methods that can significantly improve back-end performance in GMM and VQ based systems. We will also review the speaker pruning technique that was originally developed for the speaker identification task but was later adapted for the efficient online cohort model selection for score normalization [**P1**, **P2**] in speaker verification systems. For detailed complexity analysis for VQ and GMM matching we refer to [Karpov03].

3.3.1 Efficient search in vector space

The problem of finding the nearest vectors in metric space has been studied for decades [Chav01]. In general, these methods utilize the properties of metric space to build a search tree for a discrete set of vectors. This search tree allows finding the nearest vector quickly without computing all the distances. Surprisingly, such techniques have not been used in speaker recognition much, even though these techniques are mathematically proven to always find the nearest vector using less or, in worst case, the same amount of distance computations. In [**P3**], we applied *vantage point tree* [Uhlm91] for indexing speaker codebooks and found it to improve matching in VQ-based system without any degradation of accuracy.

Vantage point tree (VPT) or the "metric tree" was introduced in [Uhlm91] and later refined in [Yian93, Chiu94]. VPT is built recursively by taking one random vector as the root and computing median distance to all the rest. Those elements that have distance to root less than the median are grouped into the left sub-tree and those with larger distance into the right sub-tree. This process is repeated recursively on each sub-tree until all the elements have their place. Search for the nearest vector is started by computing distance to the root and, if it is smaller than the median, the search continues to the left sub-tree, otherwise to the right (if distance is equal we enter into both sub-trees). By traversing the tree in this way we report a set of vectors from which we select the one with smallest distance, which will be the nearest vector. Construction of VPT is illustrated in Figure 3.3.

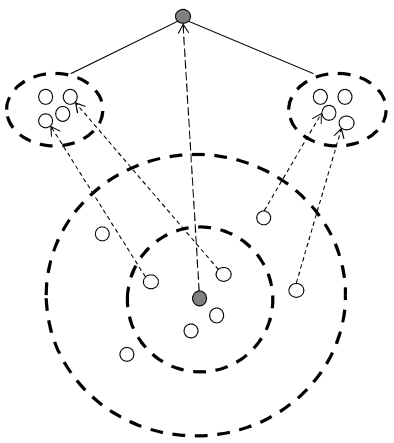
Figure 3.3 VPT construction

A serious drawback of such indexing methods is that they require a proper *metric* to construct such indexes, i.e., a distance function that satisfies the triangle inequality (which states that the sum of distances between vectors ***a*** and ***b*** and vectors ***a*** and ***c*** is always bigger or the same as the distance between vectors ***c*** and ***b***). As the likelihood scoring function in GMM is not a metric, metric space indexing cannot be applied to speed up its computation, and thus, this technique is limited only to VQ-based systems. We attempt to generalize this idea one level higher in [**P4**] by indexing the entire speaker space but did not find a good metric to measure the distance between speaker models that would allow us to construct such an index.

3.3.2 Quantized HMM

Since its introduction, GMM-based speaker recognition systems [Reyn95] have received significant attention and represent the mainstream modeling technique. However, in comparison to VQ, GMM is computationally more demanding. The majority of the load originates from the computation of Gaussian mixture densities. In the simplest case, when using diagonal covariance matrix mixtures [Vasila00]:

$$b(x) = \sum_{k=1}^{K} \omega_k \frac{1}{\prod_{i=1}^{N} \sqrt{2\pi\sigma_{ki}^2}} \exp\left(-\sum_{i=1}^{N} \frac{(x_i - \mu_{ki})^2}{2\sigma_{ki}^2} \right) \quad , \qquad (3.1)$$

where *K* is the number of Gaussian densities in a mixture, and *N* is the dimensionality of the feature vectors. In this formula, the mixture weights ω_k and Gaussian normalization factor values are known beforehand and can be pre-computed and stored with the model. The only time consuming part remaining is computing the Mahalanobis distance [Mah36], that is, the argument of the exponential term in (3.1).

To attack this problem, quantization of mean μ and variance σ has been proposed in [Vasila00] for all Gaussian components. The authors suggest using two global quantizers, one for means and one for variances, to effectively compress their values into two codebooks with 5 bits for mean index and 3 bits for variance index (per each feature vector component). This requires storing only *N* (where N is feature vector dimension) bytes plus the pre-computed weight and normalization factor for each Gaussian density. After the means and variances have been quantized, they can only have predefined values for each feature component and a look-up table can be constructed for every feature vector for fast Gaussian density computation [Vasila00]. The authors reported a speedup factor of 4:1 in [Vasila00] with only minor degradation in accuracy.

This idea has been further extended in [Vasila04] by quantizing the feature vector space as well. If features are also quantized the helper tables described above can be pre-computed for each Gaussian during model training. By doing so, Gaussian density computation can efficiently be implemented using simple table look-ups and summations with only a negligible effect on accuracy but significant savings both in storage and computational load. For a detailed description of the method, we refer to [Vasila04].

The method described in this section has been used in a speech recognition system in [Vasila04] and it has also shown to be efficient to run on a mobile device [**P6**]. Based on the fact that GMM can be seen as single state HMM, this method can also be applied to GMM-based speaker recognition as well.

3.3.3 Speaker pruning

Speaker pruning [**P1**] is a method to speed up the matching step when distance or likelihood computations are required for several speaker models. This is the case in open-set speaker identification or efficient real-time selection of cohort speakers for score normalization [**P2**].

The idea of pruning is to monitor how the match scores for each speaker model develop over time when more speech data is processed. When a model score saturates we can compare it with others and decide if we continue updating this score or prune it out from further computation and save resources. In [**P3**], we experimented with four pruning methods:

- Static pruning,
- Hierarchical pruning,
- Adaptive pruning,
- Confidence-based pruning.

These methods differ mainly in how and when the decision is made to prune speaker models from computation. In *static pruning* (SP), we use a fixed interval at which we sort the speaker models and prune out a fixed amount of those having the lowest scores. *Hierarchical pruning* (HP) uses two sets of models for each speaker, one for coarse speaker representation and another for more detailed modeling. Scores for the coarse model are computed first and speaker models with lowest scores are pruned out. Only a few detailed models are retained for the final decision. In *adaptive pruning* (AP), based on Gaussian assumption on the match score distribution, we prune models that are far from score distribution according to a pre-defined threshold. In *confidence-based pruning* (AP), the idea is similar to static method but, instead of pruning a fixed amount of models at each iteration, we prune only those models whose scores have been saturated with higher confidence. According to our experiments, the adaptive pruning method gives the best trade-off between accuracy and computational savings while the static method performed worst. In general, pruning algorithm can provide 2-5 times speed-up for match score computations. For detailed descriptions and results, we refer to [**P3**].

While the pruning method is most naturally suited for speaker identification from large speaker populations, it can also be used for certain tasks in speaker verification [**P2**]. We have conducted extensive experiments with different pruning methods and, in general, have found it to be a very efficient technique [Karpov03, **P1** and **P3**].

4 Summary of contributions

In this chapter we summarize results and contributions of the original publications [**P1**-**P7**] and their role in a general speaker recognition system. The majority of the contributions are focused on speeding up system performance and lowering resource usage. [**P1**] and [**P2**] contribute to methods for efficient scoring of speaker models, [**P5**] analyzes device limitation effects in feature extraction, and [**P3**] proposes a series of methods for a full speaker recognition system. Efficient matching methods are discussed in [**P4**] and [**P6**] and voice activity detection in [**P7**].

In the first paper [P1], we discovered that, in speaker identification, most of the speakers are correctly identified well before the full sample utterance is used. Based on this observation, we propose a novel speaker pruning algorithm. The main idea in this approach is to continuously monitor the difference between match scores for all stored models and prune out clear outsiders to avoid unnecessary computations. The challenge was to find a good heuristic to prune out "weak" models. In the paper, we introduce two variants, *static* and *adaptive* pruning, based on how the pruning threshold is selected. The static algorithm has a predefined parameter setup and works well in most conditions. The adaptive variant is slightly more complicated to implement but it is able to achieve better tradeoff between computing time and identification error rate.

Experiments on TIMIT corpus indicated that, using the best parameter choices for the adaptive algorithm achieves 0.46% error rate with just 24 seconds of speech sample whereas full matching took 230 seconds to reach 0.15% error rate. Even though TIMIT has only noise free speech samples we expect that especially the adaptive pruning algorithm can be tuned to cope well with adverse acoustic conditions as well.

In the second paper [P2], we propose a computationally efficient algorithm for adaptive selection of cohort speaker models for score normalization in speaker verification. Most speaker verification systems use a pre-defined set of cohort speakers for score normalization. Rather than using a fixed cohort, the proposed *pre-quantization based cohort search* (PQS) selects the most similar cohort models to the given test utterance. Computational load is reduced by quantizing the input vector sequence using the LBG clustering algorithm. Only the quantized data is used in scoring both the target speaker and the impostors.

In our experiments we compare the proposed PQS method to a static cohort selection that uses a pre-defined number of cohort speakers. Results on the NIST-1999 corpus show speed-up factors of 23:1 and 9:1 for the GMM- and VQ-based systems, respectively. Furthermore, the equal error rates (EERs) are slightly decreased from those of a full search. For GMM-based system with static cohort selection, EER is 7.51 %, while for the proposed method it is 7.37 %. VQ-based system equal error rate is the same for static and proposed variants.

In the third paper [P3], we combine and extend our previous work on speaker pruning **[P1]** and cohort score normalization **[P2]** with additional optimization techniques. We propose methods for optimizing speaker identification and verification systems where the input stream is processed in short frames that are first tested against a voice activity detector (VAD) to avoid matching non-speech parts. We then extract features from these frames which are further pre-quantized to a smaller representative sequence. Speaker models are indexed using search trees for efficient nearest neighbor search. To reduce the complexity of speaker matching, we propose several speaker pruning algorithms. We focus on optimizing vector quantization

(VQ) based speaker identification but also extend the same methods to Gaussian mixture modeling (GMM).

We extend the work of [**P1**] with two novel pruning algorithms: *hierarchical* and *confidence-based* pruning. The idea in hierarchical pruning is to use "coarse" and "detail" models to represent each speaker. Both are generated from the same training data, but the size of the coarse model is much smaller than the size of the detail model. Test vectors are first scored against the coarse models, and a number of speakers are pruned out. The match scores of the remaining speakers are then re-computed using the detail models. In the confidence-based pruning variant, only speakers whose match scores have stabilized are considered for pruning. If the match score is low but oscillates, the speaker can still change its rank and become selected later. Thus, we remove only speakers that have both stabilized and whose match score (average distortion) exceeds the pruning threshold.

TIMIT corpus was used for parameter tuning and the results were then validated using the NIST 1999 SRE corpus. Speaker model indexing using the vantage point tree (VPT) method improves matching time for models of size more than 32 vectors in VQ-based system. For codebooks of size 256 vectors it gives about 20 % improvement in identification time without any change in error rate. With the optimized VAD threshold, about 12% of the frames were classified as non-speech, and the identification time improved by about 10 % without degradation in accuracy. Pre-quantization of the feature vectors reduces the running time to about 50 % of the full search with only a minor degradation in the accuracy. Combination of VAD, feature vector pre-quantization and speaker pruning gives the best speed-up factor of 16:1 increasing identification error rate from 17.3 % to 18.2 %. For the GMM system, corresponding speed-up factor of 34:1 is obtained with increase of error rate from 16.9 % to 18.5 %. An equal error rate of 7% was reached in 0.84s on average when the length of test utterance is 30.4 s.

In the fourth paper [P4], we study novel *symmetrization* strategies for match score computation in speaker recognition tasks. Usually, the match score in VQ- and GMM-based systems is computed by measuring the distance or log-likelihood of each test utterance vector to the speaker model(s), i.e., *data-to-model* type of matching. But in speaker recognition, the fundamental question is whether two given utterances are produced by the same or different speaker. Therefore, the roles of the test utterance and speaker models should be, in principle, exchangeable. This study was also motivated by the success of vector space indexing in **[P3]**. If a proper distance measure in the *speaker model space* can be defined, an index can be constructed for fast search from very large speaker databases.

We study an alternative strategy for comparing two speech utterances represented by their corresponding codebooks. Computing the match score is mathematically formulated as measuring distance between two codebooks. We study four symmetric functions for a VQ-based speaker identification system based on traditional quantization distortion measure: minimum, maximum, sum and product. We study their theoretical properties with respect to the axioms for a proper distance measure and perform recognition experiments on NIST 1999 SRE corpus.

We found that sum and maximum metrics perform the best with error rates close to the baseline system. Unfortunately, error rates increase with codebook size. For codebook of size 64, the baseline error rate is 16.8% and corresponding sum and maximum rates are 16.9% and 16.8%, respectively. For codebook size of 256, the baseline system error rate is 14.9% while for sum and maximum distances they are 17.9% and 16.5%, respectively.

One reason for the unexpected behavior might be that the training and test segments are not of equal duration in our experiments. In particular, the test utterances are much shorter than the training utterances. The proposed approach may benefit in other situations where the same amount of data would be available both for training and testing.

In the fifth paper [P5], we focus on feature extraction algorithms used in speaker recognition. While MFCC feature extraction is fast enough on modern PCs with a floating point unit, they become a bottleneck of the whole system on mobile devices with fixed point arithmetics. We study all the steps in MFCC feature extraction, namely signal preprocessing, fast Fourier transform, filterbank and discrete cosine transform. We carefully analyze how these components can be ported to fixed point arithmetic and derive techniques to avoid information loss when a floating-point algorithm is replaced by a fixed-point version. We analyze the preservation of discrimination information with tests made on algorithms that are identical except for the different arithmetic used. The Fourier transform is found as the most critical component affecting overall system accuracy. Its efficiency is based on the layer structure. However, fixed point implementations introduce significant round-off errors. These errors accumulate in the repeatedly applied butterfly layers. We propose to reduce the representation accuracy in order to increase the amount of preserved signal information in Fourier transform.

In the experiments, we utilized two corpora, TIMIT and self-collected corpus containing dual recordings on mobile phone and desktop PC. We compare accuracy of conventional desktop PC based speaker identification system to the proposed fixed point arithmetic methods. The results indicate that we are able to mostly preserve accuracy also in fixed point variant, with a maximum degradation of 5% in identification rate.

We also analyzed the effect of signal pre-processing in an actual mobile device by utilizing the self-collected corpus. While both fixed and floating point variants give 100 % identification rate for data recorded in PC environment, the same system performs much worse on the data recorded on mobile device. Floating point system achieves 83% identification rate while the fixed point version gives only a 76% identification rate. This clearly indicates that signal pre-processing on the mobile device affects speaker specific information.

In the sixth paper [P6], we present an embedded speech recognition system for short message (SMS) dictation in US English. The system has been implemented on Nokia Series 60 mobile phones such as 6630, N70 and E60 using the Symbian operating system. In the paper, we describe the architecture and design of the system, and illustrate its performance both in off-line database simulations and in on-line usability tests. While this study does not address speaker recognition, it has a collection of best performing signal processing methods optimized for embedded systems over that might also be utilized in speaker recognition systems. Both speaker recognition and ASR methods share similar underlying front-end component, namely, extraction of MFCC features using similar signal processing operations. In both applications, these features are modeled using similar acoustic models involving Gaussian mixtures. Thus, despite the opposite goals in these two recognition tasks, good practices in feature extraction in either one are often useful in both tasks. In particular, robustness to environmental noise and transmission channel variability is required in both applications. We concentrate on practical aspects - execution time and used device memory - while preserving overall system accuracy. We reduce input data with long-term spectral divergence (LTSD) voice activity detector and optimize HMM-based acoustic modeling using model and input data quantization.

For the experiments we have used self-collected *Personal Communication* (PCOM) data that covers 12 topics representative of typical messaging communications, submitted by 23 users aged between 15 to 51 years. The text database was divided into disjoint training and test sets. The training set consisted of about two million words while the test set had approximately 4 thousand messages. Our system reaches 90.43 % word accuracy for density-tied biphones (2k) and 91.20 % word accuracy for biphones (5k). Even though our system vocabulary contains 23 thousand words, it is compact and efficient; it's Flash and RAM memory footprints are only 2 and 2.5 megabytes, respectively. After a short enrollment session, most native speakers can achieve a word accuracy of over 90% when dictating short messages in quiet or moderately noisy environments.

In the last paper [P7], we are in search of a robust voice activity detector (VAD) method. While VAD is a relatively well studied problem, acceptable solutions that work consistently across different acoustic conditions are yet to be found. Motivated by success of *classifier fusion* techniques in pattern recognition, we propose to use the *decision fusion* technique to combine decisions from multiple different VADs for improving speech/non-speech segmentation accuracy. We evaluate different combinations of four well known industrial VADs, namely G729, AMR, AFE and Silk, and compare it with standalone detection accuracy of respective methods. We combine VAD decisions by using two basic strategies: *majority voting* and *temporal context voting*. In majority voting, we classify each frame as majority of methods report while in temporal context voting we utilize a hangover scheme to correct possible erroneous frame decisions of the individual VADs.

For evaluations we used three datasets, NIST 2005 SRE, Bus stop and Lab. NIST 2005 SRE is a telephone-quality speaker recognition corpus, *Bus stop* contains data recorded from bus timetable search speech interface application and *Lab* is a very low signal-to-noise ratio recording recorded in our university facilities. First, we ran our tests on NIST 2005 to select best performing combinations and fusion techniques. These were then validated on the remaining two datasets. We found that combining G729, AMR2 and SILK using a context of 11 frames produces the best miss rate of 7.24% while having relatively high 77.4% false alarm rate. Combining G729, AMR1 and AMR2 with a simple majority voting (context size of 1) produces the smallest false alarm rate of 38.2% with reasonable 23.5% miss rate. However, only the latter combination generalizes on the other datasets, giving comparable or higher accuracy compared to the standard VADs. The best results were obtained on the most challenging Lab dataset, with low false alarm rate and comparable miss rate.

A summary of the main achievements is shown in Table 4.1

Table 4-1 Summary of the main results

	Contribution	Corpus	Results
[P1]	A novel speaker pruning algorithm: static and adaptive variants	TIMIT	10:1 speed-up in VQ system without affecting identification accuracy.
[P2]	Efficient algorithm for cohort normalization: pre-quantization based cohort search (PQS)	NIST-1999	23:1 speed-up in GMM system with improved EER from 7.51 % to 7.37 %.
[P3]	Extended speaker pruning algorithms: hierarchical and confidence-based pruning, pre-quantization of the feature vectors, efficient matching using vector space indexing, silence detection, and combinations of the above methods	TIMIT and NIST-1999	34:1 speed-up in GMM speaker identification system (with minor error rate degradation from 16.9 % to 18.5 %), 36:1 real-time factor in GMM speaker verification (EER = 7 %).
[P4]	Symmetric measures with comparable error rates based on traditional quantization distortion measure: minimum, maximum, sum and product.	NIST-1999	With minor degradation in error rate from baseline 14.9 % to 17.9 %
[P5]	Porting speaker recognition system to run on limited hardware that lacks floating point unit with special focus on preserving accuracy of feature extraction methods.	TIMIT and self collected corpus	Fixed-point implementation with 5% relative degradation in accuracy.
[P6]	Embedded speech recognition system for short message (SMS) dictation in US English with 23 thousand words in vocabulary	PCOM self-collected acoustic corpus	90.43% word accuracy for density-tied biphones (2k) and 91.20% word accuracy for biphones (5k) with less than 2 and 2.5 megabytes in Flash and RAM memory, respectively.
[P7]	Classifier fusion techniques to combine decisions from standalone voice activity detectors (VAD) for better detection accuracy	NIST2005 Bus stop and Lab	Miss and false alarm rates of 7.24 % and 38.3 % respectively

For the practical system running on a mobile device we propose to use quantized GMMs as a modeling technique as they produce more accurate error rates and most of the modern systems nowadays are based on GMM and therefore such a system will be easier to extend with new methods. However, for a really resource constrained devices we recommend the use of VQ as it can still perform reasonably good and is very efficient with vector space indexing enabled. For front-end MFCCs are still state of the art technique and there are efficient fixed point implementations available. We also propose to use simple energy based VAD to prune out silence vectors and feature pre-quantization to reduce the amount of input data. For speaker recognition systems we also propose to include speaker pruning as it significantly speeds-up performance. All the parameters of the algorithms have to be adjusted based on the real data from operational environment.

The contributions of the author of this thesis can be briefly summarized as follows. In [**P1-P4**] the author was involved in the design and implementation of the methods, planning and execution of evaluation tests and contributed to the text writing. Also in [**P4**] author was one of the key contributors. In [**P5**] the author contributed to method implementation concentrating on mobile device side and also helped with text writing. In [**P3**] author contributed to method design and implemented and evaluated all the proposed methods. In [**P6**] the author contributed to dictation system optimization on the mobile phone and helped with text writing. The user interface of the demonstrator was also implemented by the author. In [**P7**] author contributed to paper writing and algorithm design.

5 Conclusions

Today, speaker recognition systems are getting closer to much wider acceptance as the technology matures. Nevertheless, it is important to realize that in embedded systems domain many technical limitations may affect overall system performance. In this thesis we have studied the problems that practical speaker recognition system may face when ported on mobile device and proposed several methods to tackle the algorithmic and device limitations.

Any typical embedded system or mobile device has *a priori* significantly lower resources than average desktop computer can offer to the scientist and so many algorithms have to be re-considered from this angle. Performance of system components has to be analyzed with mobile device limitations in mind. Not only limited CPU and power considerations have to be taken into account but also device limited support for mathematical operations and typical signal disturbances that are common when the device is used in real life. All those limitations may render any well working system to completely useless on a mobile platform.

We have touched nearly all the parts of a typical speaker recognition system. For feature extraction, we have analyzed methods to reduce the computational load and numerical errors from porting algorithms to fixed point arithmetic. However, as feature extraction is not the main contributor to system performance, the majority of our efforts were targeted on pattern matching. We have proposed several novel methods for speeding up distance computations for vector quantization and Gaussian mixture models.

From the author's personal experience, even if a system is performing well in a simulation mode in close to real device conditions, the only way to discover its real capabilities is to implement the entire system on mobile or embedded device and verify its performance by hands-on experiments. Most of the optimization methods that have been presented in this thesis have been implemented into practical systems running entirely on a mobile device.

Despite the presented optimization methods there is still a need for further improvements. Speaker recognition is likely to continue for several decades still but focus is nowadays clearly shifting from conventional desktop PC based system to more optimized embedded system solutions. In author's personal view, the most challenging problem that is not solved yet is the environmental noise effect that may significantly reduce speaker recognition accuracy. Even though tackling environmental differences is very important for any practical speaker recognition system this topic has not received enough attention in this work and continues to be a direction for future research.

References

[Atal72] Atal, B. "Automatic speaker recognition based on pitch contours". *Journal of the Acoustic Society of America* 52, 6 (1972), 1687–1697.

[Atal74] Atal, B. "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification". *Journal of the Acoustic Society of America* 55, 6 (1974), 1304–1312.

[ASDK] Android speech input SDK: http://developer.android.com/resources/articles/speech-input.html, accessed on 19 Mar 2011.

[Auck00] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas. "Score normalization for text-independent speaker verification systems", *Digit. Sign. Proc.*, 10:42–54, 2000.

[Auck01] R. Auckenthaler and J. S. Mason, "Gaussian selection applied to text-independent speaker verification," in Proc. *Speaker Odyssey: The Speaker Recognition Workshop (Odyssey 2001)*, Crete, Greece, 2001, pp. 83–88.

[Bayom] Bayometric voice authentication solutions: http://www.bayometric.com/products/biometric-voice-authentication.htm, accessed on 19 Mar 2011.

[Bimbot00] F. Bimbot, M. Blomberg, L. Boves, D. Genoud, H.-P. Hutter, C. Jaboulet, J. Koolwaaij, J. Lindberg, and J.-B. Pierrot. "An overview of the CAVE project research activities in speaker verification", *Speech Communications*, 31:155–180, 2000.

[Bimbot04] Bimbot, F., Bonastre, J.-F., Fredouille, C., Gravier, G., Magrin-Chagnolleau, I., Meignier, S., Merlin, T., Ortega-Garcia, J., Petrovska-Delacretaz, D., and Reynolds, D. "A tutorial on text-independent speaker verification", *EURASIP Journal on Applied Signal Processing 2004*, 4 (2004), 430–451.

[Bishop06] Bishop, C. *Pattern Recognition and Machine Learning*. Springer Science+Business Media, LLC, New York, 2006.

[Burget07] Burget, L., Matˇejka, P., Schwarz, P., Glembek, O., and ˇCernock´y, J. "Analysis of feature extraction and channel compensation in a GMM speaker recognition system", *IEEE Trans. Audio, Speech and Language Processing 15, 7* (September 2007), 1979–1986.

[Buril00] D. Burileanu, L. Pascalin, C. Burileanu, M. Puchiu, "An Adaptive and Fast Speech Detection Algorithm", *Proc. TSD 2000 - Third International Workshop on Text, Speech and Dialogue*, Brno, Czech Republic, September 13-16, 2000.

[Burton87] Burton, D. "Text-dependent speaker verification using vector quantization source coding", *IEEE Trans. Acoustics, Speech, and Signal Processing 35*, 2 (February 1987), 133–143.

[Camp97] Campbell, J. "Speaker recognition: a tutorial". *Proceedings of the IEEE 85*, 9 (September 1997), 1437–1462.

[Camp02] W. M. Campbell, “”Generalized linear discriminant sequence kernels for speaker recognition», *in Proceedings of the International Conference on Acoustics Speech and Signal Processing*, 2002, pp. 161–164.

[Camp04] Campbell, W., Campbell, J., Reynolds, D., Jones, D., and Leek, T. "Phonetic speaker recognition with support vector machines". *In Advances in Neural Information Processing Systems* 16, S. Thrun, L. Saul, and B. Scholkopf, Eds. MIT Press, Cambridge, MA, 2004.

[Camp06a] Campbell, W., Sturim, D., and Reynolds, D. "Support vector machines using GMM supervectors for speaker verification", *IEEE Signal Processing Letters 13*, 5 (May 2006), 308–311.

[Camp06b] Campbell, W., Campbell, J., Reynolds, D., Singer, E., Torres-Carrasquillo, P., "Support vector machines for speaker and language recognition", *Comput. Speech Lang. 20* (2–3), 2006, 210–229.

[Chav01] E. Chavez, G. Nevarro, R. Bayeza-Yates, J. Marroquin, "Searching in Metric Spaces", *ACM Computing Surveys (CSUR)* September 2001 Vol. 33, pp. 273-321.

[Chev01] Cheveigne, A., and Kawahara, H. "Comparative evaluation of f0 estimation algorithms". In Proc. *7th European Conference on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, September 2001), pp. 2451–2454.

[Chiu94] Chiueh, T., "Content-based image indexing", In Proceedings of the *Twentieth Conference on Very Large Databases* (VLDB'94), 1994, pp. 582–593.

[Chow10] Chowdhury, M.F.R., Selouani, S.-A., O'Shaughnessy, D., "Text-independent distributed speaker identification and verification using GMM-UBM speaker models for mobile communications", In Proc. *10th Information Sciences Signal Processing and their Applications (ISSPA)*, 2010, pp. 57 - 60

[Davis80] Davis, S., Mermelstein, P. "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences". *IEEE Transactions on Acoustics, Speech, and Signal Processing,* 28(4):357–366, 1980

[Dehak09] N. Dehak, R. Dehak, P. Kenny, N. Brummer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification," *in INTERSPEECH*, Brighton, UK, Sept 2009.

[Dehak11] Dehak, N., Kenny, P., Dehak, R., Dumouchel, P and Ouellet, P. "Front-End Factor Analysis for Speaker Verification" *IEEE Transactions on Audio, Speech and Language Processing*, 19(4), pp. 788-798, May 2011.

[Deller00] J. R. Deller Jr., J. H. L. Hansen, and J. G. Proakis, *Discrete-Time Processing of Speech Signals*, 2nd ed. New York: IEEE Press, 2000.

[Dodd01] Doddington, G. "Speaker recognition based on idiolectal differences between speakers". In Proc. *7th European Conference on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, September 2001), pp. 2521–2524.

[Dragon] Dragon dictation for Applie iPhone: http://www.nuancemobilelife.com/apple/dictation.html, accessed on 19 Mar 2011.

[Equitz94] Equitz W.H.: "A new vector quantization clustering algorithm", *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 37(10): 1568-1575, October 1989.

[ETSI99] ETSI EN 301 708 Recommendation: *Voice Activity Detector (VAD) for Adaptive Multi-Rate (AMR) Speech Traffic Channels*, ETSI, Sophia Antipolis, Dec. 1999

[ETSI00] ETSI ES 202 050 Recommendation: *Speech processing, Transmission and Quality aspects (STQ);Distributed speech recognition; Advanced front-end feature extraction algorithm*; Compression algorithms, 2000

[Fränti97] Fränti P., Kaukoranta T., Nevalainen O.: "On the splitting method for vector quantization codebook generation", *Optical Engineering*, 36(11): 3043- 3051, November 1997.

[Fränti00] P. Fränti and J. Kivijärvi, "Randomized local search algorithm for the clustering problem", *Pattern Analysis and Applications*, 3 (4), 358-369, 2000

[Farrell94] Farrell, K., Mammone, R., and Assaleh, K. "Speaker recognition using neural networks and conventional classifiers", *IEEE Trans. on Speech and Audio Processing* 2, 1 (January 1994), 194–205.

[Furui81] Furui, S. "Cepstral analysis technique for automatic speaker verification", *IEEE Transactions on Acoustics, Speech and Signal Processing* 29, 2 (April 1981), 254–272.

[Furui91] S. Furui, "Vector-Quantization-Based Speech Recognition and Speaker Recognition Techniques", *IEEE Signals, Systems and Computers*, 1991, Vol. 2, pp. 954-958.

[Furui97] Furui, S. "Recent advances in speaker recognition", *Pattern Recognition Letters* 18, 9 (September 1997), 859–872.

[Furui05] Furui, S. "50 years of progress in speech and speaker recognition", *Proc. SPECOM 2005,* Patras, Greece, pp.1-9 (2005-10).

[Ger91] Gersho, A., and Gray, R. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, Boston, 1991.

[Haigh93] J.A. Haigh, J.S. Mason, "Robust Voice Activity Detection using Cepstral Features", *Computer, Communication, Control and Power Engineering*. Proceedings. TENCON '93, 1993 IEEE Region 10 Conference, Part: 30000 , 1993, Vol. 3, pp. 321-324

[Hanilci11] Hanilci, C., Ertas, F., "Comparison of the impact of some Minkowski metrics on VQ/GMM based speaker recognition", *Comput. Electr. Eng. 37*, 2011, 41–56

[Henn06] J. Hennessy, D. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 2006.

[Herm90] Hermansky, H. "Perceptual linear predictive (PLP) analysis of speech". *Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990

[Herm94] Hermansky, H., Morgan, N. "RASTA processing of speech", *IEEE Trans. on Speech and Audio Processing 2*, 4 (October 1994), 578–589.

[Hess83] W. Hess. *Pitch Determination of Speech Signals*. Springer-Verlag, Heidelberg, 1983.

[Hoxey96] S. Hoxey, F. Karim, B. Hay, H. Warren. *The PowerPC Compiler Writer's Guide*. Published for IBM by: Warthman Associates, 1996.

[Huang01] X. Huang, A. Acero, and H.-W. Hon. *Spoken Language Processing: a Guide to Theory, Algorithm, and System Development*. Prentice-Hall, New Jersey, 2001.

[Huer98] Huerta J.M., Stern R.M., "Speech Recognition from GSM Codec Parameters", *Proc. ICSLP-98*, 1998.

[ITU96] ITU-T Recommendation G.729-Annex B. (1996). *A silence compression scheme for G.729 optimized for terminals conforming to recommendation V.70.*

[Iwano04] K. Iwano, T. Asami, and S. Furui. "Noise-robust speaker verification using f0 features". In Proc. *Int. Conf. on Spoken Language Processing* (ICSLP 2004), volume 2, pages 1417–1420, 2004.

[Karpov03] E. Karpov, *Real-time speaker identification*. Master of Science thesis, 2003.

[Kenny07] Kenny, P., Boulianne, G., Ouellet, P., Dumouchel, P., "Joint Factor Analysis Versus Eigenchannels in Speaker Recognition". *IEEE Transactions on Audio, Speech, and Language Processing,* 15(4): 1435 - 1447, 2007

[Kockm11] Kockmann, M.; Ferrer, L.; Burget, L.; Shriberg, E.; Cernocky, J.; "Recent progress in prosodic speaker verification", *In Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2011)* (Prague, May 2011), pp. 4556 - 4559.

[Kond69] A.M. Kondoz, *Digital Speech: Coding for Low Bit Rate Communication Systems,* John Wiley & Sons, Ltd., 1969

[Kinn05] T. Kinnunen, R. Gonzalez-Hautamäki, "Long-Term F0 Modeling for Text-Independent Speaker Recognition", Proc. *Int. Conf. on Speech and Computer* (SPECOM'2005), pp. 567-570, Patras, Greece, October 2005

[Kinn10] T. Kinnunen and H. Li, "An Overview of Text-Independent Speaker Recognition: from Features to Supervectors", *Speech Communication* 52(1): 12--40, January 2010

[Kinn11] T. Kinnunen, I. Sidoroff, M. Tuononen and P. Fränti, "Comparison of clustering methods: a case study of text-independent speaker modeling", *Pattern Recognition Letters*, 32 (13), 1604-1617. October 2011

[Knuth74] Knuth, Donald: “Structured Programming with Goto Statements”, *Computing Surveys 6:4* (1974), 261–301.

[Linde80] Linde Y., Buzo A., Gray R.M.: “An algorithm for vector quantizer design”, *IEEE Trans. On Communications*, 28(1): 84-95, January 1980.

[Liu02] M. Liu, E. Chang, and B. Q. Dai, “Hierarchical gaussian mixture model for speaker verification,” in Proc. Int. *Conf. on Spoken Language Processing (ICSLP 2002)*, Denver, CO, 2002, pp. 1353–1356.

[Loquen] Loquendo identity verification solutions: http://www.loquendo.com/en/products/speaker-verification/, accessed on 19 March 2011.

[Mah36] Mahalanobis, P. C. "On the generalised distance in statistics". *Proceedings of the National Institute of Sciences of India* 2 (1): 49–55, 1936

[Matr07] D. Matrouf, N. Scheffer, B. Fauve, J.-F. Bonastre "A Straightforward and Efficient Implementation of the Factor Analysis Model for Speaker Verification", *in Proc. Interspeech*, 2007, pp. 1242-1245.

[McLa99] J. McLaughlin, D. A. Reynolds, and T. Gleason, “A study of computation speed-ups of the GMM-UBM speaker recognition system,” *in Proc. 6th European Conf. Speech Communication and Technology (Eurospeech 1999)*, Budapest, Hungary, 1999, pp. 1215–1218.

[Makh75] Makhoul, J. “Linear prediction: A tutorial review”. *Proc. IEEE*, 63:561–580, 1975.

[Marks88] J. Marks, “Real Time Speech Classification and Pitch Detection”, *IEEE Communications and Signal Processing*, 1988. Proceedings., COMSIG 88. Southern African Conference, pp. 1-6.

[Martin01] A. Martin, D. Charlet, L. Mauuary, "Robust Speech/Non-Speech Detection using LDA applied to MFCC", *IEEE Acoustics, Speech, and Signal Processing,* 2001 IEEE International Conference, Vol. 1, pp. 237-240.

[Naik89] Naik, J., Netsch, L., and Doddington, G. "Speaker verification over long distance telephone lines", *In Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1989)* (Glasgow, May 1989), pp. 524–527.

[Nasr88] Nasrabadi N.M., Feng Y.: "Vector quantization of images based upon the Kohonen self-organization feature maps", *Neural Networks,* 1: 518, 1988.

[Nuance] Nuance verifier voice authentication solution: http://www.nuance.com/ucmprod/groups/enterprise /@web/@enus/documents/collateral/nd_006284.pdf, accessed on 19 Mar 2011.

[Pele75] Pelecanos, J., and Sridharan, S. "Feature warping for robust speaker verification", *In Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)* (Crete, Greece, June 2001), pp. 213–218.

[Pellom98] B. L. Pellom and J. H. L. Hansen, "An efficient scoring algorithm for gaussian mixture model based speaker identification," IEEE Signal Process. Lett., vol. 5, no. 11, pp. 281–284, Nov. 1998.

[Quat00] T. Quatieri, R. Dunn, D. Reynolds, J. Campbell, and E. Singer, "Speaker recognition using G.729 speech codec parameters," *in Proc. ICASSP,* vol. II, 2000, pp. 1089–1092.

[Ram04] Javier Ramírez, José C. Segura, Carmen Benítez, Ángel de la Torre and Antonio Rubio, "Efficient voice activity detection algorithms using long-term speech information", *Speech Communication,* Volume 42, Issues 3-4, April 2004, pp 271-287.

[Rao10] Rao, K.S., Vuppala, A.K., Chakrabarti, S., Dutta, L., "Robust speaker recognition on mobile devices", *in*

Proc. Signal Processing and Communications (SPCOM), Bangalore, July 2010, pp. 1-5.

[Reyn94] Reynolds, D. A. "Experimental evaluation of features for robust speaker identification", *IEEE Transactions on Speech and Audio Processing,* 2(4):639–643, 1994.

[Reyn95] Reynolds, D. "Speaker identification and verification using Gaussian mixture speaker models", *Speech Communication 17* (August 1995), 91–108

[Reyn00] Reynolds, D., Quatieri, T., and Dunn, R. "Speaker verification using adapted gaussian mixture models", *Digital Signal Processing 10, 1 (January 2000),* 19–41.

[Reyn02] Reynolds, D. A. "An overview of automatic speaker recognition technology," in Proc. *Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2002),* Orlando, FL, 2002, pp. 4072–4075.

[Reyn03] Reynolds, D. "Channel robust speaker verification via feature mapping", *In Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (Hong Kong, China, April 2003), Vol. 2, pp. 53–56.

[Roy11] A. Roy, M. M. Doss, and S. Marcel. "Fast speaker verification on mobile phone data using boosted slice classifiers", in Proc. *IEEE IAPR International Joint Conference on Biometrics (IJCB),* Washington, October 2011.

[Saeidi10] R. Saeidi, J. Pohjalainen, T. Kinnunen, P. Alku, "Temporally Weighted Linear Prediction Features for Speaker Verification in Additive Noise", *Odyssey 2010: The Speaker and Language Recognition Workshop,* Brno, Czech Republic, pp. 40-46, June 2010

[Sen10] Senoussaoui, M., Kenny, P., Dehak, N., and Dumouchel, P., "An i-vector Extractor Suitable for Speaker Recognition with both Microphone and Telephone Speech", *in Proc Odyssey Speaker and Language Recognition Workshop,* Brno, Czech Republic, June 2010

[Shrib05] Shriberg, E., Ferrer, L., Kajarekar, S., Venkataraman, A., and Stolcke, A. "Modeling prosodic feature sequences for speaker recognition", *Speech Communication* 46, 3-4 (July 2005), 455–472.

[Silk09] Silk codec: http://tools.ietf.org/html/draft-vos-silk-00, July 6, 2009, accessed on 19 March 2011.

[Soong87] Soong, F., A.E., A. R., Juang, B.-H., and Rabiner, L. "A vector quantization approach to speaker recognition", *AT & T Technical Journal 66* (1987), 14–26.

[Sun03] B. Sun,W. Liu, and Q. Zhong, "Hierarchical speaker identification using speaker clustering," in Proc. Int. *Conf. Natural Language Processing and Knowledge Engineering 2003*, Beijing, China, 2003, pp. 299–304.

[Titze94] I. Titze. *Principles of Voice Production*. Prentice Hall, Englewood Cliffs, NJ, 1994.

[Tyd07] Tydlitat, B., Navratil, J., Pelecanos, J.W.; Ramaswamy, G.N., "Text-Independent Speaker Verification in Embedded Environments", in Proc. *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007*, pp. IV-293 - IV-296.

[Uhlm91] J. Uhlmann, "Satisfying general proximity/similarity queries with metric trees," *Inform. Process. Lett.*, Vol. 40, pp. 175–230, 1991.

[Vasila00] M. Vasilache: "Speech recognition using HMMs with quantized parameters", *INTERSPEECH 2000*: 441-444

[Vasila04] Vasilache M., Iso-Sipilä J. and Viikki O., "On a Practical Design of a Low Complexity Speech Recognition Engine", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, pp. 113-116, Montreal, Quebec, Canada, 2004.

[Viikki98] Viikki, O., and Laurila, K. "Cepstral domain segmental feature vector normalization for noise robust speech recognition". *Speech Communication* 25 (August 1998), 133–147.

[Vogt08] Vogt, R., Sridharan, S., 2008. "Explicit modeling of session variability for speaker verification", *Comput. Speech Lang.* 22 (1), 17–38.

[Wolf72] J. Wolf "Efficient acoustic parameters for speaker recognition". *Journal of the Acoustic Society of America* 51, 6 (Part 2) (1972), 2044–2056.

[Xiang03] B. Xiang and T. Berger, "Efficient text-independent speaker verification with structural gaussian mixture models and neural network," *IEEE Trans. Speech Audio Process.*, Vol. 11, No. 5, pp. 447–456, Sep. 2003.

[Yian93] Yianilos, P., "Data structures and algorithms for nearest neighbor search in general metric spaces", In Proceedings of the *Fourth ACM–SIAM Symposium on Discrete Algorithms* (SODA'93), 1993, pp. 311–321.