UNIVERSITY OF JOENSUU
COMPUTER SCIENCE
DISSERTATIONS 12

Tomi H. Kinnunen

# Optimizing Spectral Feature Based Text-Independent Speaker Recognition

Academic dissertation

To be presented, with the permission of the Faculty of Science of the University of Joensuu, for public criticism in the Louhela Auditorium of the Science Park, Länsikatu 15, Joensuu, on June 19$^{\text{th}}$ 2005, at 13 o'clock.

UNIVERSITY OF JOENSUU
2005

Supervisor Professor Pasi Fränti
Department of Computer Science
University of Joensuu
Joensuu, FINLAND

Reviewers Professor Sadaoki Furui
Department of Computer Science, Furui Laboratory
Graduate School of Information Science and Engineering
Tokyo Institute of Technology, JAPAN

Professor Unto K. Laine
Laboratory of Acoustics and Audio Signal Processing
Helsinki University of Technology, FINLAND

Opponent Professor Anil K. Jain
Departments of Computer Science and Engineering
Michigan State University, USA

ISBN 952-458-693-2 (printed)
ISBN 952-458-694-0 (PDF)
ISSN 1238-6944 (printed)
ISSN 1795-7931 (PDF)
Computing Reviews (1998) Classification: I.2.7, I.5.1, I.5.4, I.5.3, G.1.6

Joensuun yliopistopaino

Joensuu 2005

**Optimizing Spectral Feature Based Text-Independent Speaker Recognition**

Tomi H. Kinnunen

Department of Computer Science
University of Joensuu
P.O.Box 111, FIN-80101 Joensuu, FINLAND
tomi.kinnunen@cs.joensuu.fi

University of Joensuu, Computer Science, Dissertations 12
Joensuu, 2005, 156 pages
ISBN 952-458-693-2 (printed), 952-458-694-0 (PDF)
ISSN 1238-6944 (printed), 1795-7931 (PDF)

## Abstract

AUTOMATIC speaker recognition has been an active research area for more than 30 years, and the technology has gradually matured to a state ready for real applications. In the early years, text-depended recognition was more studied but gradually the focus has moved towards text-independent recognition because their application field is much wider, including forensics, teleconferencing, and user interfaces in addition to security applications.

Text-independent speaker recognition is considerably more difficult problem compared to text-depended recognition because the recognition system must be prepared for an arbitrary input text. Commonly used acoustic features contain both linguistic and speaker information mixed in highly complex way over the frequency spectrum. The solution is to use either better features or better matching strategy, or a combination of the two. In this thesis, the subcomponents of text-independent speaker recognition are studied, and several improvements are proposed for achieving better accuracy and faster processing.

For feature extraction, a frame-adaptive filterbank that utilizes rough phonetic information is proposed. Pseudo-phoneme templates are found using unsupervised clustering, and frame labeling is performed via vector quantization, so there is no need for annotated training data. For speaker modeling, experimental comparison of five clustering algorithms is carried out, and the answer to the question of which clustering method should be used is given. For the combination of feature extraction and speaker modeling, multiparametric speaker profile approach is studied. In particular, combination strategies for different fullband spectral feature sets is addressed.

Speaker identification is computationally demanding due to the large number of comparisons. Several computational speedup methods are proposed, including prequantization of the test sequence and iterative model pruning, as well as their combination.

Finally, selection of the cohort models (background models, anti-models) is addressed. A large number of heuristic cohort selection methods have been proposed in literature, and there is controversy how the cohort models should be selected. Cohort selection is formulated as a combinatorial optimization problem, and genetic algorithm (GA) is used for optimizing the cohort sets for the desired security-convenience balance. The solution provided by the GA is used for establishing a lower bound to the error rate of an MFCC/GMM system, and the selected models are analyzed with an aim to enlighten the mystery of the cohort selection.

**Keywords:** Text-independent speaker recognition, vector quantization, spectral features, Gaussian mixture model, cohort modeling, classifier fusion, realtime recognition.

# Acknowledgements

CONGRATULATIONS! For one reason or another, you have opened my PhD thesis. You are currently holding a piece of work to which I have devoted quite a many hours of hard work. The work was carried out at the Department of Computer Science, University of Joensuu, Finland, during 2000-2005. In the two first years of my postgraduate studies, I was an assistant in the CS department, and since 2002, my funding has been covered by the Eastern Finland Graduate School in Computer Science and Engineering (ECSE).

My supervisor Professor Pasi Fränti deserves big thanks for helping me when I've needed help, for giving a large amount of constructive criticism, as well arranging nice *ex tempore* events. I am thankful to Professors Sadaoki Furui and Unto K. Laine, the reviewers of the thesis, for the helpful comments.

My colleagues Ismo Kärkkäinen and Ville Hautamäki deserve special thanks for their endless help in practical things. I also want to thank the other co-authors Evgeny and Teemu, the rest of the PUMS group, as well as other colleagues. Joensuu has been a pleasant place to work. Which reminds me that I must thank the pizza and kebab places of the town, for keeping me in good shape.

Gladly, life is not just work (well, except for the time of PhD studies maybe). The greatest thanks go to my lovely parents and my wonderful sister, who have shown understanding, love, and simply good company during my life. Many other persons would deserve thanks, hugs and smiles as well, but I might forget easily someone. And, in fact, I need to get this thesis for print in one hour. So my dear friends out there: remember that you are special, and forget me - NOT! :) You are the air that I am breathing when I am not working, sleeping or playing the guitar. See you soon, it's summer time!

Joensuu, Monday 30th of May 2005, 3 weeks before the defense. –*Tomi*

# List of original publications

**P1.** T. Kinnunen, T. Kilpeläinen, P. Fränti. Comparison of Clustering Algorithms in Speaker Identification, *Proc. IASTED Int. Conf. Signal Processing and Communications* (SPC 2000), pp. 222-227, Marbella, Spain, September 19-22, 2000.

**P2.** T. Kinnunen, Designing a Speaker-Discriminative Adaptive Filter Bank for Speaker Recognition, *Proc. 7th Int. Conf. on Spoken Language Processing* (ICSLP 2002), pp. 2325-2328, Denver, Colorado, USA, September 16-20, 2002.

**P3.** T. Kinnunen, V. Hautamäki, P. Fränti, On the Fusion of Dissimilarity-Based Classifiers for Speaker Identification, *Proc. 8th European Conf. on Speech Communication and Technology* (EUROSPEECH 2003), pp. 2641-2644, Geneva, Switzerland, September 1-4, 2003.

**P4.** T. Kinnunen, V. Hautamäki, P. Fränti, Fusion of Spectral Feature Sets for Accurate Speaker Identification, *Proc. 9th Int. Conf. Speech and Computer* (SPECOM 2004), pp. 361-365, St. Petersburg, Russia, September 20-22, 2004.

**P5.** T. Kinnunen, E. Karpov, P. Fränti, Real-Time Speaker Identification and Verification, Accepted for publication in *IEEE Trans. on Speech and Audio Processing.*

**P6.** T. Kinnunen, I. Kärkkäinen, P. Fränti, The Mystery of Cohort Selection, Report A-2005-1, Report series A, University of Joensuu, Department of Computer Science (ISBN 952-458-676-2, ISSN 0789-7316).

# Contents

# Chapter 1

# Introduction

SPEECH signal (see Fig. 1.1) can be considered as a carrier wave to which the talker codes linguistic and nonlinguistic information. The linguistic information refers to the message, and nonlinguistic information to everything else, including social factors (social class, dialect), affective factors (emotion, attitude), and the properties of the physical voice production appratus. In addition, the signal is transmitted over a communication channel to the listener/microphone which adds it's own characteristics. The different information are not coded in separate acoustic parameters such as different frequency bands, but instead they are mixed in a highly complex way.

In *speaker recognition*, one is interested in the speaker-specific information included in speech waves. In a larger context, speaker recognition belongs to the field of *biometric person authentication* [24, 176], which refers to authenticating persons based on their physical and/or learned characteristics. Biometrics has been appearing with increasing frequency in daily media during the past few years, and speaker recognition has also received some attention. For instance, in 12th November 2002, a voice on tape broadcast on Arabic television network referred to recent terrorist strikes which US officials believed to be connected to al-Qaeda network lead by the terrorist Osama bin Laden. The tape was sent for analysis for the IDIAP group in Lausanne, Switzerland, which concluded that the voice on the tape, with high probability, did *not* belong to bin Laden[1] .

Forensics is an area where speaker recognition is routinely applied. For instance, in Finland about 50 requests related to forensic audio research are sent each year to the National Bureau of Investigation, of which a considerable amount (30-60%) are related to speaker recognition [153]. Forensic voice samples are often from phone calls or from wiretapping and can contain huge amounts of data (consider continuous

[1] `http://news.bbc.co.uk/2/hi/middle_east/2526309.stm`

recording in wiretapping, for instance). Automatic speaker recognition could be used for locating given speaker(s) in a long recording, the task called *speaker tracking.*

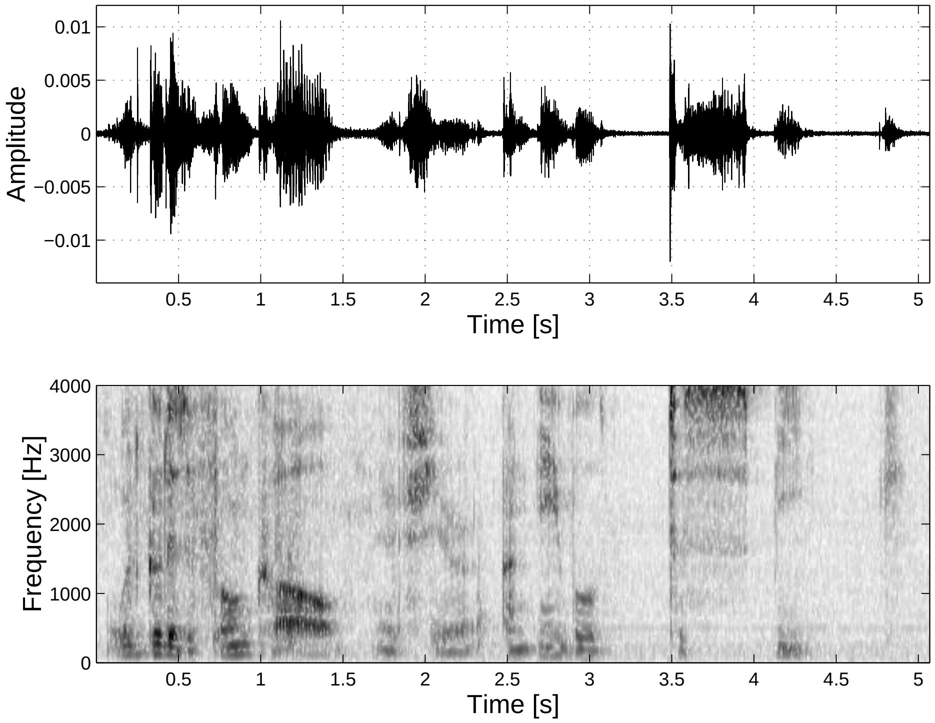


Figure 1.1: An example of speech signal: waveform (upper panel) and spectrogram (lower panel). Utterance "*What good is a phone call, if you are unable to speak?*" spoken by a male.

Recently, there has been increasing interest to apply automatic speaker recognition methodology to help decision making process in forensic speaker recognition [73, 174, 3, 154], which has traditionally been a task of a human operator having phonetic-linguistic background [189]. Increased accuracy of automatic speaker recognition systems has motivated to use them in parallel to support other analysis methods. One problem with this approach is that the results must be interpretable and quantifiable in the terms of accepted statistical protocols, which sets up more challenges to the system design. The main difference between commercial and forensic applications is that in the former case the system makes always a hard decision, whereas in the latter case, the system should output a degree of similarity, and the human operator is responsible for interpreting and quantifying the significance of the match.

For commercial applications, voice biometric has many desirable properties.

Firstly, speech is a natural way of communicating, and does not require special attention from the user. By combining speech and speaker recognition technologies, it is possible to give the identity claim via speech [85] ("I am **Tomi**, please verify me"). Secondly, speaking does not require physical contact with the sensor as contrast to fingerprints and palm prints, for instance. Thirdly, the sensor (microphone) is small, which makes speaker authentication systems attractive for mobile devices. For instance, it could be used as an alternative to the PIN number, or for continuous authentication so that if an unauthenticated person speaks to the phone, it locks itself.

Voice biometric could also be used as an additional person authentication method in e-commerce and bank transactions. PC microphones are cheap, and at home or office, the environmental acoustics is predictable so that in most practical cases noise or acoustic mismatch would not be a problem. Furthermore, as webcams have also become increasingly popular, combining voice and face recognition could be used for increasing the accuracy. In general, voice can be combined with arbitrary biometrics.

Speaker recognition and profiling has also potential to help solving other problems within speech technology. The most studied subproblem is *speech recognition*, which refers to transcribing spoken language into text. Often speech and speaker recognition are considered as separate fields, although from the technical side they share many similarities. For instance, similar acoustic features are used for both tasks with good success, which is somehow ironical considering the opposite nature of the task. This indicates that the same features contain both phonetic and speaker information, and it would be advantageous to combine the tasks [85, 20].

The main problems of speech are associated with the high variability of the signal due to (1) *speaker him/herself* (mental condition, health, long-term physiological changes), (2) *technical conditions* (environment acoustics, transmission line) and (3) *linguistic factors* (speech content, language, dialectal variations). These variabilities make it rather difficult to form a stable voice template over all different conditions. Due to the high intra-person variability of speech, a relatively large template is needed for modeling the variabilities.

## 1.1 Definitions

In automatic speaker recognition literature, speaker recognition is divided into *identification* and *verification* tasks [27, 67]. In the identification task, or *1:N matching*, an unknown speaker is compared against a database of $N$ known speakers, and the best matching speaker is returned as the recognition decision; "no one" decision is also possible in the task called *open set* identification problem.

The verification task, or *1:1 matching*, consists of making a decision whether a given voice sample is produced by a claimed speaker (the *claimant* or *target*). In general, identification task is much more difficult since a large number of speakers must be matched. The verification task, on the other hand, is less dependent on the population size.

Speaker recognition systems can be further classified into *text-dependent* and *text-independent* ones. In the former case, the utterance presented to the recognizer is fixed, or known beforehand. In the latter case, no assumptions about the text is made. Consequently, the system must model the general underlying properties of the speaker's vocal space so that matching of arbitrary texts is possible.

In text-dependent speaker verification, the pass phrase presented to the system can be *fixed*, or alternatively, it can vary from session to session. In the latter case, the system prompts the user to utter a particular phrase (*text prompting*). An advantage of text prompting is that impostor can hardly know the prompted phrase in advance, and playback of pre-recorded or synthesized speech becomes difficult. The recognition decision can be a combination of utterance verification ("did the speaker utter the prompted words?") and speaker verification ("is the voice of similar to the claimed person's voice?") [128, 188].

In general, text-dependent systems are more accurate, since the speaker is forced to speak under restricted linguistic constraints. From the methodological side, text-dependent recognition is a combination of speech recognition and text-independent speaker recognition.

## 1.2 Human Performance

In forensics, *auditory speaker recognition* might have some use. An *earwitness* refers to a person who heard the voice of the criminal during the crime. Although this protocol has been used in actual crime cases, it is somehow questionable because of the subjective nature. For instance, it has been observed that there are considerable differences in recognition accuracies between individuals [193, 189].

Human and computer performance in speaker recognition have been compared in [133, 193, 3]. Schmidt-Nielsen and Crystal [193] conducted a large-scale comparison in which nearly 50,000 listening judgments were performed by 65 listeners. The results were compared with the state-of-the-art computer algorithms. It was observed that humans perform better when the quality of the speech samples is degraded with background noise, crosstalk, channel mismatch, and other sources of noise. With matched acoustic conditions and clean speech, the performance of the best algorithms was observed to be comparable with the human listeners.

Similar results were recently obtained by Alexander *et al.* [3]. In their experiment, 90 subjects participated the aural recognition test. It was found out that in the matched conditions (GSM-GSM and PSTN-PSTN) the automatic speaker recognition system clearly outperformed human listeners (EER of 4 % vs. 16 %). However, in mismatched conditions (for instance, PSTN-GSM), human outperformed the automatic system. The subjects were also asked to describe what "features" they used in their recognition decisions. Pronounciation and accent were most popular, followed by timbre, intonation and speaking rate. It is noteworthy that the automatic system used only spectral cues (RASTA-PLP coefficients), but it still could outperform human in matched conditions. This suggests that human auditory system considers speaker features to some extent as irrelevant information or undesired noise.

## 1.3 Speaker Individuality

It is widely known that the main determinants of speaker sex are the formant frequencies and the fundamental frequency ($F_0$) [18]. Formant frequencies correspond to high-amplitude regions of the speech spectrum, and they correspond to one or more resonance frequencies of the vocal tract which are, in turn, related to the sizes of the various acoustic cavities. The overall vocal tract length (from glottis to lips) can be estimated from the formants rather accurately [147]. The $F_0$, on the other hand, depends on the size of the vibrating segments of the vocal folds, and therefore it is an acoustic correlate of the larynx size [189].

Studies in automatic speaker recognition have indicated the high frequencies to be important for speaker recognition [82, 21]. For instance, in [82] the spectrum was divided into upper and lower frequency regions, the cutoff frequency being a varied parameter. It was found out that regions 0-4 kHz and 4-10 kHz are equally important for speaker recognition. For high-quality speech, the low end of the spectrum (below 300 Hz) was found to be useful in [21].

Analysis of speaker variability of phonemes and phonetic classes has revealed some differences in discrimination properties of individual phonemes [52, 191, 204, 168, 16, 106]. The most extensive study is by Eatock and Mason [52], in which the authors studied a corpus of 125 speakers using hand-annotated speech sampled. They found out that the nasals and vowels performed the best and stop consonants the worst.

Intonation, timing, and other *suprasegmental features* are also speaker-specific, and they have been applied in automatic speaker recognition systems [12, 190, 30, 198, 124, 215, 19, 28, 62, 183, 171, 2]. These are affected by the speaker's attitude and they can be more easily impersonated compared to vocal tract features (see [11] for an imitation study). However, they have proven to be very robust against noise [30, 124, 100].

# Chapter 2

# Automatic Speaker Recognition

FROM the user's perspective, a speaker authentication system has two operational modes: *enrollment* and *recognition* modes. In the enrollment mode, the user provides his/her voice sample to the system along with his unique user ID. In the recognition mode, the user provides another voice sample, which the system compares with the previously stored sample and makes it's decision.

Depending on the application, the biometric authentication system might include several modalities, such as combination of speaker and face recognition [26]. In this case, the user provides a separate biometric sample for each modality, and in the recognition mode, the system combines the subdecisions of the different modalities. Multimodal person authentication is a research topic on its own, and will not be discussed here further.

## 2.1 Components of Speaker Recognizer

Identification and verification systems share the same components (see Fig. 2.1), and they will not be discussed separately. *Feature extractor* is common for enrollment and recognition modes. The feature extractor, or system *front-end*, transforms the raw audio stream into a more manageable format so that speaker-specific properties are emphasized and statistical redundancies suppressed. The result is a set of *feature vectors.*

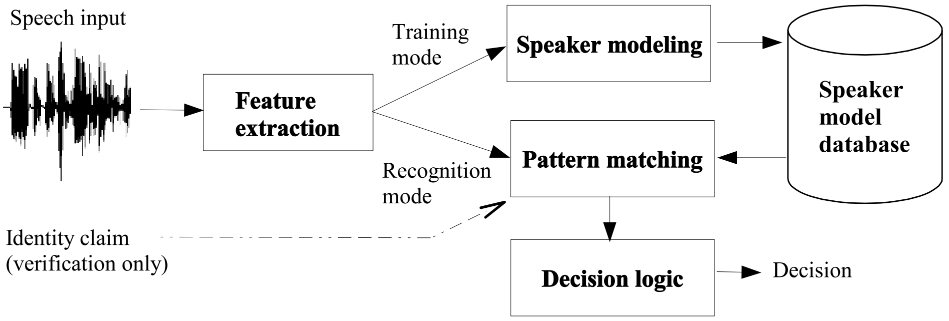


Figure 2.1: Components of an automatic speaker recognition system.

In the enrollment mode, the speaker's voice template is formed by statistical modeling of the features, and stored into speaker database. It depends on the features what type of model is most appropriate. For example, the Gaussian mixture model (GMM) [185, 184] has been established as a baseline model for spectral features to which other models and features are compared.

In the recognition mode, feature vectors extracted from the unknown person's utterance are compared with the stored models. The component responsible for this task is called *1:1 match engine*, as it compares one voice sample against one stored model. The match produce a single real number, which is a similarity or dissimilarity score. In current systems, the match score is normalized relative to some other models in order to make it more robust against mismatches between training and recognition conditions [127, 67, 92, 182, 184, 196][**P6**]. The rationale is that when there is an acoustic mismatch, it will affect equally all models, and making the score relative to other models should provide a more robust score.

The component that is essentially different for identification and verification is the *decision module*. It takes the match scores as input, and makes the final decision, possibly with a confidence value [72, 95]. In the identification task, the decision is the best matching speaker index, or "no one" in the case of open-set identification. In the verification task, decision is "accept" or "reject". In both cases, it is possible to have a refuse-to-decide option, for instance due to low SNR. In this case, the system might prompt the user to speak more.

## 2.2 Selection of Features

Feature extraction is necessary for several reasons. First, speech is a highly complex signal which carries several features mixed together [189]. In speaker recognition we are interested in the features that correlate with the physiological and behavioral characteristics of the speaker. Other information sources are considered as undesirable noise whose effect must be minimized. The second reason is a mathematical one, and relates to the phenomenon known as *curse of dimensionality* [25, 101, 102], which implies that the number of needed training vectors increases exponentially with the dimensionality. Furthermore, low-dimensional representations lead to computational and storage savings.

### 2.2.1 Criteria for Feature Selection

In [216, 189], desired properties for an ideal feature for speaker recognition are listed. The ideal feature should

- have large between-speaker and small within-speaker variability
- be difficult to impersonate/mimic
- not be affected by the speaker's health or long-term variations in voice
- occur frequently and naturally in speech
- be robust against noises and distortions

It is unlikely that a single feature would fulfill all the listed requirements. Fortunately, due to the complexity of speech signals, a large number of complementary features can be extracted and combined to improve accuracy. For instance, short-term spectral features are highly discriminative and, in general, they can be reliably measured from short segments (1-5 seconds) [151], but will be easily corrupted when transmitted over a noisy channel. In contrast, $F_0$ statistics are robust against technical mismatches but require rather long speech segments and are not as discriminative. Formant frequencies are also rather noise robust, and formant ratios, relating to the relative sizes of resonant cavities, are expected to be something that is not easily under the speaker's voluntary control. The selection of features depends largely on the application (co-operative/non co-operative speakers, desired security/convenience balance, database size, amount of environmental noise).

### 2.2.2 Types of Features

A vast number of features have been proposed for speaker recognition. We divide them into the following classes:

- Spectral features
- Dynamic features

- Source features
- Suprasegmental features
- High-level features

Table 2.1 shows examples from each class. *Spectral features* are descriptors of the short-term speech spectrum, and they reflect more or less the physical characteristics of the vocal tract. *Dynamic features* relate to time evolution of spectral (and other) features. *Source features* refer to the features of the glottal voice source. *Suprasegmental* features span over several segments. Finally, *high-level features* refer to symbolic type of information, such as characteristic word usage.

Table 2.1: Examples of features for speaker recognition.

| Feature type | Examples |
|---|---|
| Spectral features | MFCC, LPCC, LSF<br>Long-term average spectrum (LTAS)<br>Formant frequencies and bandwidths |
| Dynamic features | Delta features<br>Modulation frequencies<br>Vector autoregressive coefficients |
| Source features | $F_0$ mean<br>Glottal pulse shape |
| Suprasegmental features | $F_0$ contours<br>Intensity contours<br>Microprosody |
| High-level features | Idiosyncratic word usage<br>Pronounciation |

An alternative classification of features could be *phonetic-computational* dichotomy. Phonetic features are based on the acoustic-phonetic knowledge and they often have a direct physical meaning (such as vibration frequency of vocal folds or resonances of the vocal tract). In contrast, by computational features we refer to features that aim at finding good presentation in the terms of small correlations and/or high discrimination between speakers. These do not necessarily have any physical meaning, but for automatic recognition this does not matter.

### 2.2.3 Dimension Reduction by Feature Mapping

By *feature mapping* we refer to any function producing a linear or nonlinear combination of the original features. Well-known linear feature mapping methods include *principal component analysis* (PCA) [50], *independent component analysis* (ICA) [97] and *linear discriminant analysis* (LDA) [65, 50]. An example of a nonlinear method is the multilayer perceptron (MLP) [25].

PCA finds the directions of largest variances and can be used for eliminating (linear) correlations between the features. ICA goes further by aiming at finding statistically independent components. LDA utilizes class labels and finds the directions, on which the linear separability is maximized.

ICA can be used when it can be assumed that the observed vector is a linear mixture of some underlying sources. This is the basic assumption in the source-filter theory of speech production [57], in which the spectra of the observed signal is assumed to be a product of the spectra of excitation source, vocal tract filter and lip radiation. In cepstral domain, these are additive, which motivated the authors of [104] to apply ICA on the cepstral features. ICA-derived basis function have also been proposed as an alternative to discrete Fourier transform in feature extraction [103].

MLP can be used as a feature extractor when trained for autoassociation task. This means that the desired output vector is the same as the input vector, and the network is trained to learn the reconstruction mapping through nonlinear hidden layer(s) having a small number of neurons. In this way, the high-dimensional input space is represented using a small number of hidden units performing nonlinear PCA. Neural networks can also be used as an integrated feature extractor and speaker model [86].

### 2.2.4 Dimension Reduction by Feature Selection

An alternative to feature mapping is *feature selection* [102], which was introduced to speaker recognition in 1970s [39, 191]. The difference with feature mapping is that in feature selection, the selected features are a subset, and not a combination, of the original features. The subset is selected to maximize a separability criterion, see [27] for a detailed discussion.

In addition to the optimization criterion, the search algorithm needs to be specified, and for this several methods exist. Naive selection takes the individually best-performing features. Better approaches include bottom-up and top-down search algorithms, dynamic programming, and genetic algorithms. For a general overview and comparison, refer to [102], and for comparison in speaker recognition, see [32].

In [32], it is noted that the feature selection can be considered as a special case of *weighting* the features in the matching phase with binary weights $\{0, 1\}$ (0=feature is not selected, 1=feature is selected). Thus, a natural extension is to consider weights from a continuous set. The authors applied a genetic algorithm for optimizing the weights, and there was only a minor improvement over the feature selection.

In an interesting approach presented in [166] and later applied in [46, 40], personal features are selected for each speaker. This allows efficient exploitation of features that might be bad speaker discriminators on average, but discriminative for a certain individual.

## 2.3 The Matching Problem

Given a previously stored speaker model $\mathcal{R}$ and test vectors $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ extracted from the unknown person's sample, the task is to define a *match score* $s(\mathcal{X}, \mathcal{R}) \in \mathbb{R}$ indicating the similarity of $\mathcal{X}$ and $\mathcal{R}$. Depending on the type of the model, match score can be a likelihood, membership value, dissimilarity value, and so on.

The intrinsic complexity of speech signal makes the speaker matching problem difficult. Speech signal contains both linguistic and nonlinguistic information, which are mixed in a nonlinear way, and it is nontrivial to extract features that would be free of all other information except speaker characteristics. For example, HMM and GMM modeling of MFCC coefficients have been successfully applied in speech recognition [178], speaker recognition [181], emotion recognition [123, 126], and even in language recognition [217]. The fact that the same features give reasonable results in so diverse tasks suggests that MFCCs contain several information sources. Thus, small distance between reference and test vectors does not necessarily indicate that the vectors are produced by the same person, but they might be from different speakers pronouncing different phoneme.

Another point that deserves attention is that statistical pattern recognition literature [50, 65, 101] deals mostly the problem of classifying *single* vectors, for which the methodology is well-understood. However, in speaker recognition, we rather have a sequence of vectors $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ extracted from short-time frames around the rate of 100 vectors/sec, and we need a joint decision for the whole vector sequence presenting a complete utterance. Frames cannot be concatenated into a single vector because utterances vary in their length, and so would the dimensionality also vary. Even if one managed to equalize all the utterances to a fixed dimensionality, one would have the problem of text dependence (arbitrary order of concatenation).

Thus, it is not obvious how the traditional classification methods for the single vector case can be generalized to the problem that we call *sequence classification*. One can argue that making certain assumptions, this is a well-defined problem. For instance, it is common to assume mutual independence of the test vectors so that the joint likelihood of the test sequence $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ given the model $\mathcal{R}$ can be factorized as follows:

$$p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T | \mathcal{R}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t | \mathcal{R}). \tag{2.1}$$

However, the independence assumption does *not* hold in general, but the feature vectors have strong temporal correlations. An alternative strategy is to classify each test vector separately using traditional single-vector methods, and to combine the individual vector votes [163].

A compromise between the whole sequence classification and individual vector voting is to divide $\mathcal{X}$ into *temporal blocks* of fixed length (say $K$ vectors) [64], and classify them independently. More advanced methods include segmentation of the utterance into variable-length segments corresponding to linguistically or statistically meaningful units, which is discussed in the next section.

## 2.4 Segmentation as Preprocessing

The phonetic information (text content) is considered as the most severe inferring information to speaker recognition, and a number of approaches have been proposed for separating these two strands [76, 52, 157, 17, 138, 155, 55, 85, 1, 162, 84, 20, 145, 77][**P2**]. In text-dependent recognition, separation of phonetic and speaker information is embedded into the recognizer which performs nonlinear alignment of the reference and test utterances using Hidden Markov models (HMM) or dynamic time warping (DTW). In text-independent recognition, this kind of "stretching/shrinking" is not possible since comparable phonemes in two recordings are in arbitrary positions.

Therefore, a segmenter can be considered in text-independent case as a *preprocessor* that segments the signal. If the segmentation produces also the transcription, the segments of the same type can be compared [7, 84]. The segmentation is based on some linguistically relevant division such as phonemes/phoneme groups [157, 17, 55, 84, 167, 78], broad phonetic categories [76, 106], phoneme-like data-driven units [172][**P2**], unvoiced/voiced segments [187, 8], pitch classes [54], prosodic patterns [1], and steady/transient spectral regions [134].

In general, the segmentation/alignment and the actual matching can, and probably should be, based on independent features and models because phonetic and speaker information are, at least in theory, independent of each other. In [76], smooth spectrum features derived from a 3rd order LPC model were used for broad phonetic segmentation. In [155] the authors use principal component analysis to project the feature vectors into "phonetic" and "speaker" subspaces, corresponding to lower- and higher order principal components, respectively.

The model and features for segmentation can be speaker-independent or speaker-dependent, and these have been compared for text-depended case in [63, 33]. The results in both studies [63, 33] indicate that speaker-dependent segmentation is more accurate. However, speaker-independent segmentation needs to be done only once which makes it computationally more efficient. In [167], speaker-dependent scoring is made faster using a two-stage approach. In the first stage, a GMM speaker recognizer and speaker-independent speech recognizer are used in parallel. The GMM produces an $N$-best list of speakers, and for them speaker-dependent refined segmentation and scoring is carried out. Similar approaches, with an aim to jointly improve speech and speaker recognition performance, has been proposed in [85, 20]. The formulation was done as finding the word sequence $W$ and speaker $S$ to maximize their joint probability $p(W, S|\mathcal{X})$.

In speaker recognition, text content of the utterances is not of interest, and one could replace the symbols by an arbitrary alphabet; it only matters that the segmentation is consistent across different utterances. Annotated data is not needed for training, but phoneme-like units can be found by unsupervised methods [77][**P2**].

## 2.5 Types of Models

Campbell [27] divides speaker models into *template models* and *stochastic models.* In the former case, the model is nonparametric, and pattern matching deterministic; it is assumed that the test sample is an imperfect replica of the reference template and a dissimilarity measure between them needs to be defined. In the stochastic case, it is assumed that the feature vectors are sampled from a fixed but an unknown distribution. The parameters of the unknown distribution are estimated from the training samples, and the match score is typically based on the conditional probability (likelihood) of the observed test vectors $\mathcal{X}$ given the reference model $\mathcal{R}$, $p(\mathcal{X}|\mathcal{R})$. It is also possible to estimate the parameters of the test distribution parameters, and to compare the model parameters [23].

Models can be also divided according to training method into *unsupervised* and *supervised* (or *discriminative*) approaches [179]. In the former case, the target model is trained using his/her training data only, whereas in the latter case, the data from other classes is taken into account so that the models are directly optimized to discriminate between speakers. This is usually done using an independent tuning set matched against the models, and the models are adjusted so that the tuning set samples are classified as accurately as possible. Using another validation set, overfitting can be avoided. Unsupervised training is typical for statistical models like GMM [185] and VQ [200], and supervised training is common for neural networks [58, 86] and kernel classifiers [29, 213]. For a survey of various approaches, see [179].

A compromise between unsupervised and supervised approaches is to use a unsupervised model training and discriminative *matching* [63, 209, 141]. In this approach, non-discriminating parts of the input signal contribute less to match score. For this, likelihood ratio [63, 141], competitive model ranking [141], and Jensen difference [209] have been used.

## 2.6 Template Models

The simplest template model is *no model at all* [93, 47]. In other words, the features extracted in the training phase serve as the template for the speaker. Although this represents the largest amount of information, it can lead to excessive matching times and to overfitting. For this reason, it is common to reduce the number of test vectors by clustering such as $K$-means [129]. Even simpler approach is to represent speaker by a single mean vector [139].

In the following, the test template is denoted as $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ and the reference template as $\mathcal{R} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_K\}$. Theory of *vector quantization* (VQ) [69] can be applied in template matching. The *average quantization distortion* of $\mathcal{X}$, using $\mathcal{R}$ as the quantizer is defined as

$$D_Q(\mathcal{X}, \mathcal{R}) = \frac{1}{T} \sum_{t=1}^{T} \min_{1 \leq k \leq K} d(\boldsymbol{x}_t, \boldsymbol{r}_k), \tag{2.2}$$

where $d(\cdot, \cdot)$ is a distance measure for vectors, e.g. the Euclidean distance or some measure tailored for certain type of features (see [178]). In [36], nearest neighbor distance is replaced by the minimum distance to the projection between all vector pairs, and improvement was obtained, especially for small template sizes. Soft quantization (or *fuzzy VQ*) has also been used [208, 207].

For the vector distance $d(\cdot, \cdot)$, weighted distance measures of the following form are commonly used:

$$d_W^2(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} - \boldsymbol{y})' \boldsymbol{W} (\boldsymbol{x} - \boldsymbol{y}), \tag{2.3}$$

in which $\boldsymbol{W}$ is a weighting matrix used for variance normalization or emphasizing discriminative features. Euclidean distance is a special case when $\boldsymbol{W}$ is an identity matrix. The *Mahalanobis* distance [50] is obtained from (2.3) when $\boldsymbol{W}$ is the inverse covariance matrix. The covariance matrix can be same for all speakers or it can be speaker-depended. In [180], the covariance matrix is partition-depended. Diagonal covariance matrices are typically used because of numerical reasons.

### 2.6.1 Properties of $D_Q$

The dissimilarity measure (2.2) is intuitively reasonable: for each test vector, the nearest template vector is found and the minimum distances are summed. Thus, if most of the test vectors are close to reference vectors, the distance will be small, indicating high similarity. It is easy to show that $D_Q(\mathcal{X}, \mathcal{R}) = 0$ if and only if $\mathcal{X} \subseteq \mathcal{R}$, given that $d$ is a distance function [107]. However, $D_Q$ is not symmetric because in general $D_Q(\mathcal{X}, \mathcal{R}) \neq D_Q(\mathcal{R}, \mathcal{X})$, which arises a question what should be quantized with which one?

Symmetrization of (2.2) was recently proposed in [107] by computing the asymmetric measures $D_Q(\mathcal{X}, \mathcal{R})$ and $D_Q(\mathcal{R}, \mathcal{X})$, and combining them using sum, max, min and product operators. The maximum and sum are the most attractive ones since they define a distance function. However, according to the experiments in [107], neither one could beat out the nonsymmetric measure (2.2), which arises suspicion whether symmetrization is needed after all.

Our answer is conditional. In principle, the measure should be symmetric by intuition. However, due to imperfections in the measurement process, features are not free from context, but they contain mixed information about the speaker, text, and other factors. In text-independent recognition, the asymmetry might be advantageous because of mismatched texts. However, there is experimental evidence in favor of symmetrization. Bimbot *et al.* [23] studied symmetrization procedures for monogaussian speaker modeling, and in the case of limited data for either modeling or matching, symmetrization was found to be useful. In [107], rather long training and test segments were used, which might explain the difference. The symmetrization deserves more attention.

### 2.6.2 Alternative Measures

Higgins *et al.* [93] have proposed the following dissimilarity measure:

$$\begin{aligned} D_H(\mathcal{X}, \mathcal{R}) &= \frac{1}{T}\sum_{t=1}^{T} \min_{1 \leq k \leq K} d(\boldsymbol{x}_t, \boldsymbol{r}_k)^2 + \frac{1}{K}\sum_{k=1}^{K} \min_{1 \leq t \leq T} d(\boldsymbol{x}_t, \boldsymbol{r}_k)^2 \\ &\quad - \frac{1}{T}\sum_{t=1}^{T} \min_{1 \leq k \leq K, k \neq t} d(\boldsymbol{x}_t, \boldsymbol{x}_k)^2 - \frac{1}{K}\sum_{k=1}^{K} \min_{1 \leq t \leq T, t \neq k} d(\boldsymbol{r}_t, \boldsymbol{r}_k)^2, \quad (2.4) \end{aligned}$$

in which $d^2$ is the squared Euclidean distance. They also show that, under certain assumptions, the expected value of $D_H$ is proportional to the divergence between the continuous probability distributions. Divergence is the total average information for discriminating one class from another, and can be considered as a "distance" between two probability distributions [41]. The first two sum terms in (2.4) correspond to cross-entropies and the last two terms to self-entropies.

Several other heuristic distance and similarity measures have been proposed [143, 91, 10, 111, 116]. Matsui and Furui [143] eliminate outliers and perform matching in the intersecting region of $\mathcal{X}$ and $\mathcal{R}$ to increase robustness. In [91], the discrimination power of individual vectors is utilized. Each vector is matched against other speakers using a linear discriminant designed in the training phase to separate these two speakers. Discriminant values are then converted into votes, and the number of votes for the target serves as the match score.

Heuristic weighting utilizing discriminatory information of the reference vectors was proposed in [111, 116]. In the training phase, a weight for each reference vector is determined, signifying its distance from the other speakers' vectors. For vectors away from other classes, higher contribution is given in the matching phase. In the matching phase, the weight of the nearest neighbor is retrieved and used in the dissimilarity [111] or similarity [116] measure.

### 2.6.3 Clustering

The size of speaker template can be reduced by clustering [200]. The result of clustering is a *codebook* $C$ of $K$ *code vectors*, denoted as $C = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K\}$. There are two design issues in the codebook generation: (1) the *method* for generating the codebook, and (2) the *size* of the codebook.

General and non-surprising result is that increasing the codebook size reduces recognition error rates [200, 58, 83, 116][**P1**]. A general rule of thumb is to use a codebook of size 64-512 to model spectral parameters of dimensionality 10-50. If the codebook size is set too high, the model gets overfit to the training data and increases errors [202][**P5**]. Larger codebooks increase also matching time. Usually speaker codebooks are equal size for all speakers, but the sizes can also be optimized for each speaker [60].

The most well-known codebook generation algorithm is the *generalized Lloyd algorithm* (GLA) [129], also known as the *Linde-Buzo-Gray* (LBG), or as the *K-means* algorithm depending on the context; the names will be used here interchangeably. The algorithm minimizes the mean square error locally by starting from an initial codebook, which is iteratively refined in two successive steps until the codebook does not change. The codebook is initialized by selecting $K$ disjoint random vectors from the training set.

He *et al.* [83] proposed a discriminative codebook training algorithm. In this method, codebooks are first initialized by the LBG algorithm, and then the code vectors are fine-tuned using learning vector quantization (LVQ) principle [120]. In LVQ, individual vectors are classified using template vectors, and the template vectors are moved either towards (correct classification) or away (misclassification) from the tuning set vectors. In speaker recognition, the task is to classify a sequence of vectors rather than individual vectors. For this reason, He *et al.* modified the LVQ so that a *group* of vectors is classified (using average quantization distortion), and they call their method *group vector quantization* (GVQ). The code vectors are tuned like in standard LVQ. The GVQ method, when combined with the *partition-normalized distance measure* [180], was reported to give the best results among several VQ-based methods compared in [56].

## 2.7 Stochastic Models

### 2.7.1 Gaussian Mixture Model

*Gaussian mixture model* (GMM) [185, 184] is the state-of-the-practise model in text-independent speaker recognition. A GMM trained for short-term spectral features is often taken as the baseline to which new models and features are compared. GMM can be considered as an extension of the VQ model, in which the clusters are overlapping. The power of GMM lies in the fact that it produces smooth density estimate, and that it can be used for modeling arbitrary distributions [25]. On the other hand, a VQ equipped with Mahalanobis distance is very close to GMM.

A GMM is composed of a finite mixture of Gaussian components, and its density function is given by

$$p(\boldsymbol{x}|\mathcal{R}) = \sum_{k=1}^{K} P_k \, \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{2.5}$$

where

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = (2\pi)^{-\frac{d}{2}} |\boldsymbol{\Sigma}_k|^{-\frac{1}{2}} \exp\Big\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\Big\} \tag{2.6}$$

is the $d$-variate Gaussian density function with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. $P_k \geq 0$ are the component prior probabilities and they are constrained by $\sum_{k=1}^{K} P_k = 1$. In the recognition phase, the likelihood of the test sequence is computed as $\prod_{t=1}^{T} p(\boldsymbol{x}_t|\mathcal{R})$.

GMM parameters can be estimated using the *Expectation-Maximization* (EM) algorithm [25], which can be considered as an extension of the K-means. The EM algorithm locally maximizes the likelihood for the training data. Alternatively, GMM can be adapted from a previously trained model called a *world model* or *universal background model* (UBM). The idea in this approach is that parameters are not estimated from scratch, but prior knowledge (“speech data in general”) is utilized. The UBM is trained from a large number of speakers using the EM algorithm, and the speaker-depended parameters are adapted using *maximum a posteriori* (MAP) adaptation [184]. As an example, the mean vectors are adapted as follows:

$$\boldsymbol{\mu}_k = \frac{n_k}{n_k + r} E_k(\boldsymbol{x}) + \left(1 - \frac{n_k}{n_k + r}\right) \boldsymbol{\mu}_k^{\mathrm{UBM}}, \tag{2.7}$$

where $n_k$ is the probabilistic count of vectors assigned to $k$th mixture component, $E_k(\boldsymbol{x})$ is posterior probability weighted centroid of the adaptation data, and $r$ is a fixed *relevance factor* balancing the contribution of the UBM and the adaptation data. Compared to EM training, the MAP approach reduces both the amount of needed training as well as the training time, and it is the preferred method, especially for limited training data.

The UBM can be used in speaker verification to normalize the target score so that it is more robust against environmental variations. The test vectors are scored against the target model and the UBM, and the normalized score is obtained by dividing the target likelihood by the UBM likelihood, giving a relative score. Note that the UBM normalization does not help in closed-set identification, since the background score is the same for each speaker, and will not change the order of scores. In addition to UBM normalization, one can use a set of *cohort models* [92, 185][**P5**, **P6**].

Typically the covariance matrices are taken to be diagonal (i.e. a variance vector for each component) because of both numerical and storage reasons. However, it has been observed that full covariance matrices are more accurate [224]. In [224], the authors propose to use *eigenvalue decomposition* for the covariance matrices, where the eigenvectors are shared by all mixture components but the eigenvalues depend on the component. Although the proposed approach gave slightly smaller errors compared to normal full covariance GMM, the training algorithm is considerably much more complex than the EM algorithm.

Recently, the UBM-GMM has been extended in [219]. In this approach, the background model is presented as a tree created using top-down clustering. From the tree-structured background model, target GMM is adapted using MAP adaptation at each tree level. The idea of this approach is to represent speakers with different resolutions (the uppermost layers corresponding to most "coarse model") to speed up GMM scoring.

Another multilevel model has been proposed in [34] based on phonetically-motivated structuring. Again, the most coarse level presents the regular GMM, the next level contains division into vowels, nasals, voiced and unvoiced fricatives, plosives, liquids and silence. The third and last level consists of the individual phonemes. In this approach, phonetic labeling (e.g. using HMM) of the test vectors is required. Similar but independent study is [84].

Phoneme group specific GMMs have been proposed in [55, 167, 78]. For each speaker, several GMMs are trained, each corresponding to a phoneme class. A neat idea that avoids explicit segmentation in the recognition phase is proposed in [55]. The speaker is modeled using a single GMM consisting of several sub-GMMs, one for each phonetic class. The mixture weight of the sub-GMM is determined from the relative frequency of the corresponding phonetic symbol. Scoring is done in normal way by computing the likelihood; the key point here is that the correct phonetic class of the input frame is selected probabilistically, and there is no need for discrete labeling.

In [209], two GMMs are stored for each speaker. The first one is trained normally from the training data. Using this model, discriminativeness of each training vector is determined and the most discriminative vectors are used for training the second model. In the recognition phase, discriminative frames are selected using the first model and matched against the second (discriminative) model. The discrimination power is measured by deviation of vector likelihood values from a uniform distribution; if likelihood is same for all speakers, it does not help in the discrimination.

A simplified GMM training approach has been proposed in [121, 169], which combines the simplicity of the VQ training algorithm but retains the modeling power of GMM. First, the feature space is partitioned into $K$ disjoint clusters using the LBG algorithm. After this, covariance matrices of each cluster are computed from the vectors that belong to that cluster. The mixing weight of each cluster is computed as the proportion of vectors belonging to that cluster. The results in [121, 169] indicate that this simple algorithm gives similar or better results with the GMM-based speaker recognition with much simpler implementation.

Even more simple approach to avoid training totally is to use *Parzen window* (or *kernel density*) estimate [65] from the speaker's training vectors [186]. Given the training data $\mathcal{R} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_K\}$ for the speaker, the Parzen density estimate is

$$p(\boldsymbol{x}|\mathcal{R}) = \frac{1}{K} \sum_{k=1}^{K} K(\boldsymbol{x} - \boldsymbol{r}_k), \tag{2.8}$$

where $K$ is a symmetric kernel function (e.g. Gaussian) at each reference vector. The shape of the kernel is controlled by a *smoothing parameter* controlling the trade-off between over- and undersmoothing of the density. Indeed, there is no training for this model at all, but the density estimate is formed "on the fly" from the training samples for each test vector. The direct computation of (2.8) is time-consuming for a large number of training samples, so the dataset could be reduced by K-means. Rifkin [186] uses approximate $k$-nearest neighbor search to approximate (2.8) using the $k$ approximate nearest neighbors to the query vector.

### 2.7.2 Monogaussian Model

A special case of the GMM, referred to as *monogaussian model*, is to use a single Gaussian component per speaker [71, 70, 23]. The model consists of a single mean vector $\boldsymbol{\mu}_{\mathcal{R}}$ and a covariance matrix $\boldsymbol{\Sigma}_{\mathcal{R}}$ estimated from the training data $\mathcal{R}$. The small amount of parameter makes the model very simple, small in size, and computationally efficient. Monogaussian model has been reported to give satisfactory results [27, 21, 225]. It is less accurate compared to GMM, but the computational speedup in both training and verification is improved by one to three orders of magnitude according to experiments in [225]. Also, it is pointed out in [23] that monogaussian modeling could serve as a general reference model, since the results are easy to reproduce (in GMM and VQ, the model depends on the initialization).

In some cases, the mean vector of the model can be ignored, leading to a single covariance matrix per speaker. The motivation is that covariance matrix is not affected by constant bias, which could be resulting from convolutive noise (which is additive in cepstral domain). Bimbot *et al.* [23] found out experimentally that when training and matching conditions are clean, including mean vector improves performance, but in the case of telephone quality, the covariance model is better.

Several matching strategies for the monogaussian and covariance-only model have been proposed [71, 70, 23, 27, 214, 225]. The basic idea is to compare the differences in the *parameters* of the test and reference parameters, denoted here as $(\boldsymbol{\mu}_{\mathcal{X}}, \boldsymbol{\Sigma}_{\mathcal{X}})$ and $(\boldsymbol{\mu}_{\mathcal{R}}, \boldsymbol{\Sigma}_{\mathcal{R}})$. This speeds up scoring compared to direct likelihood computation, since the parameters of the test sequence need to be computed once only.

The means are typically compared using Mahalanobis distance, and the covariances matrices are compared using the eigenvalues of the matrix $\boldsymbol{\Sigma}_{\mathcal{X}}\boldsymbol{\Sigma}_{\mathcal{R}}^{-1}$. When the covariance matrices are equal, $\boldsymbol{\Sigma}_{\mathcal{X}}\boldsymbol{\Sigma}_{\mathcal{R}}^{-1} = \boldsymbol{I}$, and the eigenvalues will be equal to 1. Thus, a dissimilarity of the covariance matrices can be defined in the terms of the deviation of the eigenvalues from unity. Gish proposed the sum of absolute deviations from unity [70]. Bimbot *et al.* compare several eigenvalue-based distance measures, and propose different ways of symmetrizing them [23].

In some cases, the eigenvalues do need to be explicitely calculated, but the measures can be represented using traces and determinants. For instance, Bimbot *et al.* derive *arithmetic-geometric sphericity measure* which is the logarithm of the ratio of arithmetic and geometric means of the eigenvalues, and can be calculated as follows:

$$\mathrm{AGSM}(\boldsymbol{\Sigma}_{\mathcal{X}}, \boldsymbol{\Sigma}_{\mathcal{R}}) = \log \frac{\frac{1}{d}\mathrm{tr}\left(\boldsymbol{\Sigma}_{\mathcal{X}}\boldsymbol{\Sigma}_{\mathcal{R}}^{-1}\right)}{\left(|\boldsymbol{\Sigma}_{X}| \big/ |\boldsymbol{\Sigma}_{\mathcal{R}}|\right)^{1/d}}. \tag{2.9}$$

Campbell [27] defines distance between two Gaussian based on divergence [41] and

Bhattacharyya distance [65]. From these, he derives measures that emphasize differences in the shapes of the two distributions. As an example, the measure derived from the divergence, called *divergence shape*, is given by the following equation:

$$\mathrm{DS}(\mathbf{\Sigma}_{\mathcal{X}}, \mathbf{\Sigma}_{\mathcal{R}}) = \frac{1}{2}\mathrm{tr}\Big[\Big(\mathbf{\Sigma}_{X} - \mathbf{\Sigma}_{\mathcal{R}}\Big)\Big(\mathbf{\Sigma}_{\mathcal{R}}^{-1} - \mathbf{\Sigma}_{X}^{-1}\Big)\Big]. \tag{2.10}$$

To sum up, because of the simple form of the density function, the monogaussian model enables usage of powerful parametric similarity and distance measures. More complex models like GMM do not allow easy closed-form solutions to parametric matching.

## 2.8 Other Models

*Neural networks* have been used in various pattern classification problems, including speaker recognition [58, 75, 86, 125, 222]. One advantage of neural networks is that feature extraction and speaker modeling can be combined into a single network [86]. Recently, a promising speaker modeling approach has been the use of *kernel classifiers* (see [150]). The idea in these methods is to use a nonlinear mapping into a high-dimensional feature space, in which simple classifiers can be applied. The idea is different from neural networks, in which the classifier itself has a complex form.

In [29], polynomial functions are used as speaker models. The coefficients of the polynomial form the speaker model, and these are learned using discriminative training. As an example, for two-dimensional vectors $(x_1, x_2)$ and 2nd order polynomial, mapping into 6-dimensional feature space is defined in [29] as follows:

$$(x_1, x_2) \mapsto (1,\ x_1,\ x_2,\ x_1^2,\ x_1x_2,\ x_2^2). \tag{2.11}$$

In the matching phase, each vector is mapped into feature space and the inner product is computed between the speaker coefficient vector, giving an indication of similarity. The utterance-level score is given by the average of the frame-level scores. This model has a very small number of parameters; in [29] the best results were obtained using 455 parameters per speaker.

In [149, 213], the speaker *model* parameters rather than the data are mapped using kernels. This has the advantage that the parameter space has fixed dimensionality. For instance, in [149], the authors measure distance of monogaussian models in the probability density space using divergence (which can be computed analytically in this case). The experiments of [149] indicate that this simple approach outperforms GMM.

*Speaker-specific mapping* approach to speaker recognition has been proposed in [138, 145], in which the focus is in features rather than statistical modeling. The idea is to extract two parallel feature streams with the same frame rate, a feature set representing linguistic information, and a feature set containing both linguistic and speaker information. Denoting the linguistic and linguistic-speaker feature vectors as $(\boldsymbol{l}_t, \boldsymbol{s}_t), t = 1, \ldots, T$, the training consists of finding the parameters of the speaker-specific mapping function $\mathcal{F}$ so that the mean square mapping error

$$E = \frac{1}{T} \sum_{t=1}^{T} \|\boldsymbol{s}_t - \mathcal{F}(\boldsymbol{l}_t)\|^2 \qquad (2.12)$$

is minimized. One can think $\mathcal{F}$ a "speaker coloring" of the "pure linguistic" spectrum: speaker-specific detail features are added on top of the linguistic information to give the final spectrum containing linguistic and speaker features. In [138] the mapping is found using subspace approach, and in [145] using a multilayer perceptron (MLP) network. In the recognition phase, the two feature streams are extracted, and score for the speaker is defined as the mapping error (2.12) using his personal $\mathcal{F}$.

Somewhat similar approach to the linguistic-to-speaker mapping is the autoassociate neural network [75, 222] approach, in which a multilayer perceptron is trained to learn the reconstruction of features via a lower-dimensional subspace. The main difference with [138, 145] is that only one feature stream is used and so that no domain knowledge is used. The input vector and desired output vectors are the same, and the network is trained to minimize the reconstruction error.

## 2.9 Information Fusion

Decision making in human activities involves combining information from several sources (team decision making, voting, combining evidence in the court of law), in the wish to arrive at more reliable decisions. Lately, these ideas have been adopted into pattern recognition systems under the generic term *information fusion*. There has been also a clearly increasing interest towards information fusion in speaker recognition during the past few years [35, 59, 158, 197, 194, 64, 148, 179, 188, 43, 6, 136, 77][**P3**, **P4**].

Information fusion can take several forms, see [179] for an overview in speaker recognition. For instance, the target speaker might be required to utter same utterance several times (*multi-sample fusion*) so that match scores of different utterances can be combined [136]. Alternatively, a set of different features could be extracted from the same utterance (*multi-feature fusion*). Speech signal is complex and enables extraction of several complementary acoustic-phonetic, as well computational features that can capture different aspects of the signal.

In *classifier fusion*, the *same* feature set is modeled using different classifiers [31, 59, 148]. The motivation is that classifiers are based on different underlying theories such as linear/nonlinear decision boundaries, and stochastic/template approaches. It is expected that combining different classifier types, the classifiers could correct misclassifications made by other classifiers.

### 2.9.1 Input and Output Fusion

For multi-feature fusion there are two options available: *input fusion* and *output fusion*. Input fusion refers to combining the features at the frame level into a vector for which a single model is trained. In output fusion, each feature set is modeled using a separate classifier, and the classifier outputs are combined. The classifier outputs can be raw match scores, rank values, or hard decisions [221].

Input fusion, in particular, combining local static spectral features with the corresponding time derivatives to capture transitional spectral information [66, 201], has been very popular. The main advantages are straightforward implementation, the need for a single classifier only, and the fact that feature dependencies are taken into account, providing potentially better discrimination in the high-dimensional space.

However, input fusion has several limitations. For instance, it is difficult to apply when the features to combined have different frame rates, or if some feature stream has discontinuities (like $F_0$ of unvoiced frames). Feature interpolation could be used in these cases, but this is somewhat artificial - creating data that does not exist. Moreover, curse of dimensionality may pose problems especially with limited training data. Careful normalization of the features is also necessary because individual features might have different variances and discrimination power.

Output fusion enables more flexible combination strategy, because the best-suited modeling approaches can be used for different features. Because of the lower dimensionality, simple models can be used and less training data is required. Also, different features can have different meaning, they can have different scales and different number of vectors, and these can be processed in a unified way. In fact, the classifiers might present even different biometric modalities like face and voice [26].

Output fusion has some disadvantages as well. Firstly, some discrimination power might be lost if the features are statistically dependent. Secondly, memory and time requirements are increased if there is a large number of feature streams. However, this is true also for input fusion. Based on these arguments, score fusion is more preferable option in general.

### 2.9.2 Combining Classifier Outputs

If the individual classifiers output crisp labels, they can be combined using *majority voting*, i.e. by assigning the class label to the most voted one; a majority of votes is required. In [6], framework for combining rank-level classifier outputs for speaker recognition is proposed.

If classifier outputs are continuous match scores, they must be converted into compatible scale before combining. Let $s_k(\mathcal{X}, i)$ denote the raw match score for speaker $i$ given by the classifier $k$. It is customary to normalize the scores to be nonnegative and so that they sum to unity. In this way, they can be interpreted as estimates of posterior probabilities or membership degrees. Using a nonnegative function $g(s)$, the normalization [35]

$$s'_k(\mathcal{X}, i) = \frac{g(s_k(\mathcal{X}, i))}{\sum_{j=1}^{N} g(s_k(\mathcal{X}, j))} \tag{2.13}$$

ensures that $s'_k(\mathcal{X}, i) \geq 0$ and $\sum_{i=1}^{N} s'_k(\mathcal{X}, i) = 1$ for all $k$. The function $g(s)$ take different forms depending on the scores. For probabilistic classifier one can select $g(s) = s$ and for distance classifiers $g(s) = \exp(-s)$.

The sum and product rules [119, 205, 4], sometimes refered to also as *linear opinion pool* and *logarithmic opinion pool* respectively, are commonly used. They are given as follows:

$$\begin{aligned} \mathcal{F}_{\text{sum}}(\mathcal{X}, i) &= \sum_{k=1}^{K} w_k s'_k(\mathcal{X}, i) & (2.14) \\ \mathcal{F}_{\text{prod}}(\mathcal{X}, i) &= \prod_{k=1}^{K} s'_k(\mathcal{X}, i)^{w_k}, & (2.15) \end{aligned}$$

where $w_k \geq 0$ are the relative significance of the individual classifiers to the final score. The weights can be determined from the accuracies of the classifiers, from classifier confusion matrices using information theoretic approach [5], or from estimated acoustic mismatch between training and recognition [192]. Properties of the sum and product rules for the equal weights case ($w_k = 1/K$) have been analyzed in [119, 205, 4]. In general, the sum rule is preferred option since the product rule amplifies estimation errors [119].

# Chapter 3

# Feature Extraction

SPEECH signal changes continuously due to the articulatory movements, and the signal must be analyzed in short segments or *frames*, assuming local stationarity within each frame. Typical frame length is 10-30 milliseconds, with an overlap of 25-50 % of the frame length. From each frame, feature vector(s) are computed.

Estimation of the short-term spectrum forms a basis for many feature representations. Spectrum can be estimated using the discrete Fourier transform (DFT) [160], linear prediction [137], or some other methods. Common steps for most spectrum estimation methods in speech processing are *pre-emphasis* and *windowing*, see Fig. 3.1 for an example. Pre-emphasis boosts higher frequency region so that vocal tract related features are emphasized. Pre-emphasis also makes linear prediction (LP) analysis more accurate at higher frequencies. The purpose of windowing is to pick the interesting part of the "infinite-length" signal for short-term analysis.

## 3.1 Spectral Analysis Using DFT

When *discrete Fourier transform* (DFT) [160, 98] is used as a spectrum estimation method, each frame is multiplied by a *window function* to suppress the discontinuity at the frame boundaries. Notice that the "no windowing" case in fact applies a window, however, a rectangular one. Frame multiplication in the time domain corresponds to convolving the true signal spectrum with the spectrum of the window function [81, 45, 177, 160]. In other words, the window function itself causes error to the spectrum estimation (leading to so-called *spectral leakage* effect which means that the spectral energy "leaks" from DFT bins to each other). If frame smoothing is not done, this is equivalent to measuring a blurred version of the actual spectrum.

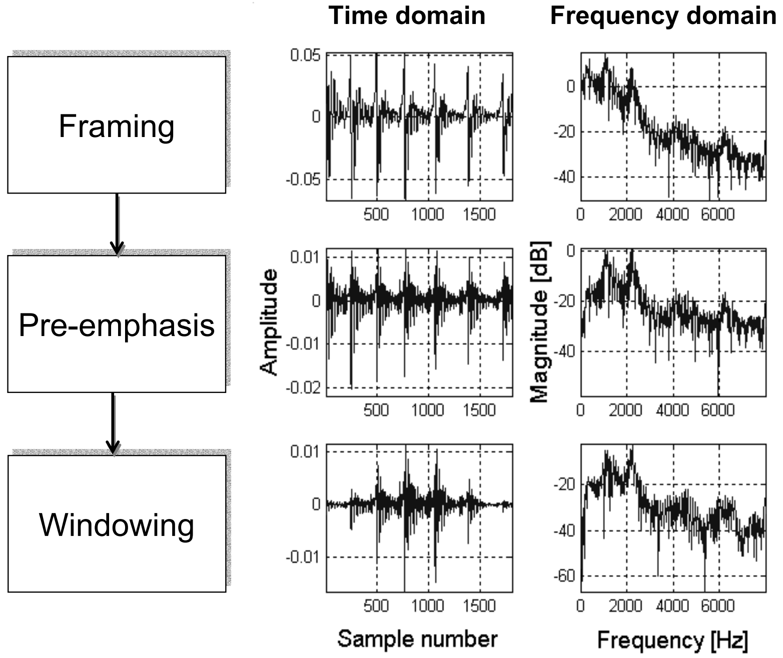


Figure 3.1: Effects of framing, pre-emphasis and windowing in time and frequency domains.

For a detailed discussion of the desired properties for a window function, see [81].

In addition to selecting the window function, other crucial parameters are the frame length and overlap. The frequency resolution of the DFT can be increased by using longer frame, but this leads to decreased time resolution. Wavelets [203] offer a nonuniform tiling of the time-frequency plane, but the short-term DFT remains the mainstream approach.

Framing is straightforward to implement but it has several shortcomings, from which the frame adjustment needs special caution as demonstrated in [108]. The authors demonstrate with a synthetic example that for a periodic signal, two equal length frames started from different positions lead to high spectral distance. As a solution, the authors propose to use a variable frame length chosen to be an integer multiple of the local pitch period.

Pitch-synchronous analysis has also been utilized in [227], but with different motivation than in [108]. Although source and filter features are in theory separated by the conventional spectral feature representations like MFCC and LPCC, in practise the spectral features are affected by pitch. The authors of [227] denote that in NIST evaluations, pitch mismatch between training and recognition has been observed

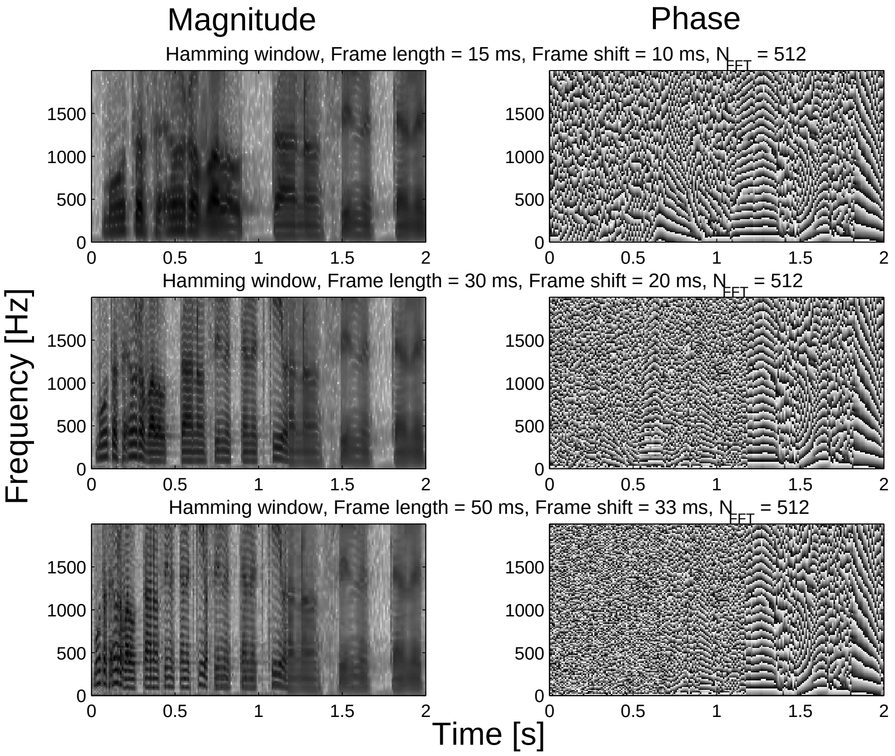


Figure 3.2: Effects of windowing parameters to speech magnitude- and phase spectrograms.

to increase errors, and hypothesize that by removing the harmonic structure from the spectrum, "depitching" the local spectrum, would be advantageous for speaker recognition. Unfortunately, the verification accuracy turned out to be worse for the depitched case. Pitch-class depended spectral feature modeling has been proposed in [54, 8]. In this approach, each pitch class (e.g. voiced/unvoiced) is associated with its own model.

## 3.2 DFT in Feature Extraction

Formally, the result of an $N$-point DFT is an $N$-dimensional complex vector, which can be expressed in equivalent magnitude-phase form by using the polar coordinates. Magnitude and phase spectra are $N/2$-dimensional vectors, from which the original time-domain signal can be restored.

Typically, only the magnitude spectrum is retained and the phase spectrum is neglected. In this way, half of the data is dropped away, and there is no hope of restoring the original signal anymore. Very few studies have considered using phase information as a potential feature for speaker recognition [87]. The motivation for *not* using phase is based on the general belief that phase has little effect on perception of speech [80, 68]. However, there is recent evidence to support just the opposite. In [164], the authors conducted a consonant intelligibility task by resynthesizing auditory stimuli from magnitude or phase spectrum only and destroying the other one. It was observed that the intelligibility is a function of the window function type and the window size. In particular, for longer time windows, the phase spectrum turned out to be more useful.

Figure 3.2 shows the low frequency part of the magnitude and phase spectra from an utterance spoken by a male speaker. Frame length is varied in the three panels, while keeping the frame shift fixed to 2/3 of the frame length. The window function is Hamming and the number of FFT bins is 512. The magnitude spectrum shows somewhat more structured signal, but the phase spectrum does not look completely random either.

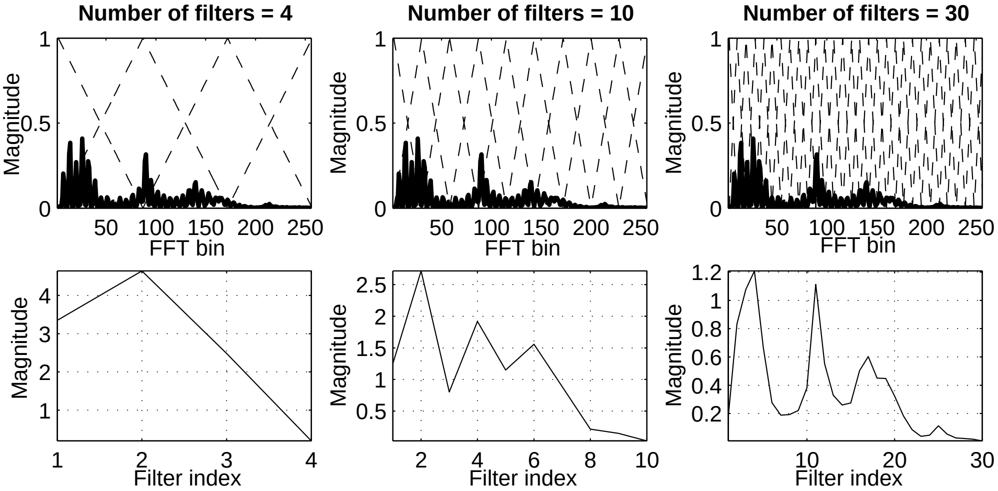


Figure 3.3: An example of filtering in the frequency domain.

DFT has several other limitations in addition to spectral leakage. A fundamental problem is that the base frequencies are constrained to be harmonically related of the form $\omega_k = (2\pi k)/N$, where $k = 0, 1, \ldots, N - 1$. The Fourier spectrum can be considered as a least squares fit of sines and cosines of predetermined frequencies to the input data [99]. The authors in [99] utilize a more flexible approach known as *Hildebrand-Prony* method in which the basis functions are periodic, but not constrained to be harmonically related.

Another alternative spectrum model known as *Fourier-Bessel expansion* was utilized in [74]. The method is similar to standard Fourier analysis in that the basis functions are orthogonal, and the Fourier-Bessel coefficients are unique. The difference is that the basis functions are aperiodic and decay with time, which might better suit to the physics of speech.

A machine-learning approach to finding speaker-depended basis functions was proposed in [103]. The authors compare Fourier basis functions and cosine basis functions (DCT) with principal component analysis (PCA) and independent component analysis (ICA) derived basis functions. The experiments on a subset of TIMIT corpus indicated that ICA-derived basis functions perform the best.

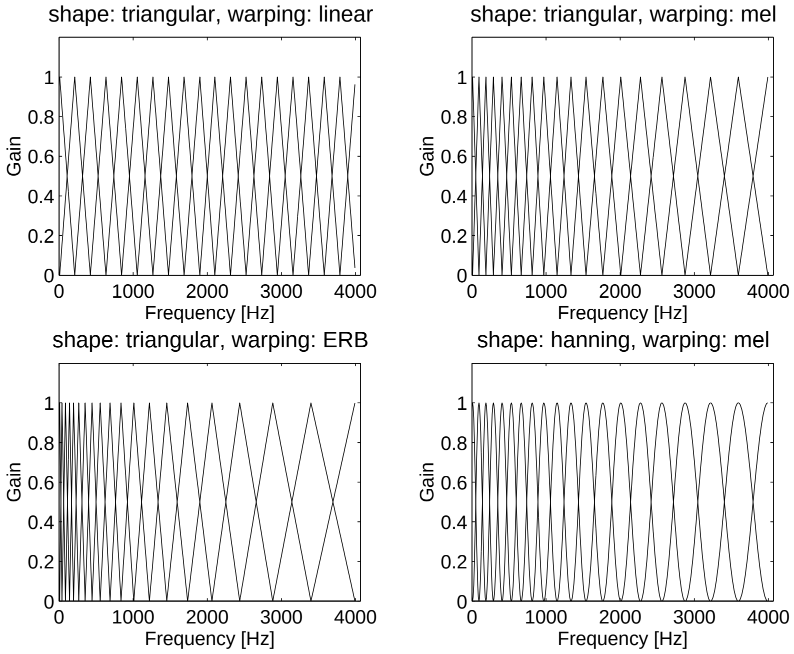


Figure 3.4: Examples of filter banks with different filter shapes and frequency warpings.

## 3.3 Subband Processing

Subband processing enables independent processing and fusion of different subbands. The block carrying out the division into individual frequency bands is called a *filterbank*, and it can be implemented in both time and frequency domains. In the former case, time-domain signal is convolved with a set of bandpass-type of filter kernels, each one designed to pick a certain frequency band. This produces a set of filtered time-domain signals, for which normal overlapping frame feature extraction can be performed. In the frequency-domain approach, DFT is applied to every frame, and the frequency bands are selected from the spectrum by windowing the Fourier spectrum (see Fig. 3.4). The DFT approach (implemented using FFT) is computationally more efficient, and for this reason also more popular. For an example about the convolution approach in speaker recognition, see [43].

Filterbank provide a smoothed version of the raw magnitude spectrum by averaging spectral bands over certain sets of frequencies, see Fig. 3.3 for an example. The upper panels show the DFT magnitude spectrum and the filters. The lower panels show the smoothed spectrum that is obtained as a weighted sum using the filters as the weighting function. In this example, the filters are triangular shaped and their center frequencies are linearly spaced on the frequency axis so that the response of the whole filterbank equals 1 over all frequencies.

### 3.3.1 Emphasizing Important Frequencies

The filter center frequencies can be linearly spaced, or according to a nonlinear *frequency warping* function. Frequency warping can be used to adjust the amount of resolution around specified subbands. Psychoacoustically motivated warping functions have been popular. For instance, the *mel*- and the *Bark*-scales [80] both give more resolution to the low end of the spectrum, and lump together higher frequencies more aggressively. Some examples of warped filterbank magnitude responses are shown in Fig. 3.4 with different filter shapes and warping functions.

In [42, 161], speaker discriminating division of the frequency domain is implemented by placing more filters with narrower bandwidths on the discriminating regions. For instance, the authors of [161] determine the discrimination power using $F$-ratio [27] and a heuristic vector ranking criterion. In [14], the authors hypothesize that it would be advantageous to equalize the error rates over different subbands. Based on the error rates of individual frequency bands and other empirical observations, they designed an equalizing warping function. The experiments confirmed that the subband error rates had more flat distribution, and for a female speaker set the error rates were degreased from mel-warped filterbank. However, for male subset there was no improvement. In [146], a parametric warping function was proposed for speaker recognition. The first parameter defines the frequency band, and the second parameter controls the amount of resolution around this band. The filterbank parameters were optimized for both GMM and VQ models using a gradient-search type algorithm.

### 3.3.2 Subband Modeling

Some subbands might be corrupted by noise whereas others are still usable. Motivated by this, separate model for each subband can be used [22, 21, 223, 195, 43, 144]. For instance, in [22, 21] subbands are modeled using a monogaussian model and in [144] using a GMM.

Damper and Higgins [43] combine the subband likelihoods by the product rule without any weighting, whereas Sivakumaran *et al.* [195] use three weighting approaches based on subband error rate, SNR, and discrimination power using competing speaker models. Discriminative weighting yielded the lowest error rates.

The approach by Ming *et al.* [144] is an interesting one because of its simplicity and minimal assumptions about the type or amount of noise. The method selects automatically subbands that are less contamined by noise. This is based on maximizing the *a posterior* probability of a given speaker model with respect to the uncontaminated subbands. Intuitively, if a subband is contamined by noise, it matches poorly all speaker models, and will not be selected to scoring.

### 3.3.3 Filterbank Cepstrum

Spectral subbands are correlated, and some form of dimensionality reduction should be applied. The most well-known approach is the discrete cosine transform (DCT) applied to mel-warped filterbank outputs, the approach known as *mel-cepstrum* [44]. Denoting the outputs of an $M$-channel filterbank as $Y(m), m = 1, \ldots, M$, the cepstral coefficients are given as follows [94]:

$$c_n = \sum_{m=1}^{M} \log Y(m) \cos\left[\frac{\pi n}{M}\left(m - \frac{1}{2}\right)\right]. \tag{3.1}$$

Notice that $c[0]$ equals the sum of log-compressed filter outputs and correlates with the total energy of the frame. Thus, it depends on the distance to microphone and is not usually included in the cepstral vector. Usually 10-20 low-order coefficients are retained, and possibly weighted (*cepstral liftering*) to increase their robustness or to emphasize speaker differences.

In [90], DCT was replaced by FIR filtering of the log magnitude spectrum, in other words, convolution in the frequency domain. The motivation of the authors was a combined decorrelation and emphasis of important features. The experiments on four different filters indicated that a smaller EER can be obtained by using FIR filtering. The best result was obtained by using a second order bandpass filter $z - z^{-1}$, which corresponds to so-called *bandpass liftering* [178].

# 3.4 Linear Prediction

## 3.4.1 Linear Model of Speech Production

*Linear prediction* (LP) [137] is an alternative spectrum estimation method to DFT. LP can be considered as a rough formulation for the *source-filter* theory of speech production [57]. The "filter" of the LP model represents a transfer function of an all-pole model, consisting of a set of spectral peaks that are more or less related to the resonance structure of the vocal tract, as well as to the spectral properties of the excitation signal. The prediction residual signal represents temporal properties of the signal that are not captured by the all-pole model.

The linear speech production model is given in the time domain by the following equation [178]:

$$s[n] = \sum_{k=1}^{p} a_k s[n-k] + G\, u[n], \tag{3.2}$$

where $s[n]$ is the observed signal, $a_k$ are the *predictor coefficients*, $u[n]$ is the source signal and $G$ is the gain. The predictor equation of LP is given as follows:

$$\tilde{s}[n] = \sum_{k=1}^{p} a_k s[n-k]. \tag{3.3}$$

Equation (3.3) states that current speech sample can be predicted from a linear combination of past $p$ samples, which is an intuitively reasonable assumption in short term (within the analysis frame). The predictor coefficients $a_k$ are determined so that the square error is minimized:

$$\min_{(a_1,\ldots,a_p)} \sum_n \left( s[n] - \sum_{k=1}^{p} a_k s[n-k] \right)^2 \tag{3.4}$$

The coefficients are typically solved using the *Levinson-Durbin* algorithm [178, 94, 80]. Frequency-domain interpretation of the model (3.2) is obtained by taking $Z$-transforms of both sides of (3.2) and solving for the filter transfer function:

$$H(z) = \frac{S(z)}{U(z)} = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}}, \tag{3.5}$$

where $S(z)$ and $U(z)$ are the $Z$-transforms of $s[n]$ and $u[n]$, respectively. This is a transfer function of an *all-pole* filter. The poles are the roots of the denominator, and they correspond to local maxima in the spectrum. Examples of all-pole spectra (bold line) are shown in Fig. 3.5. The DFT spectrum (thin line) is shown for comparison.

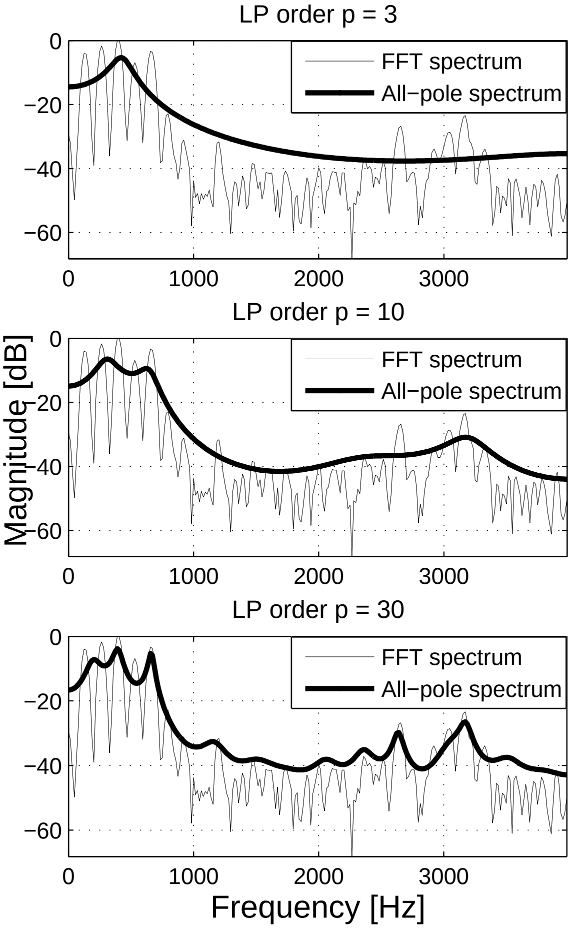


Figure 3.5: FFT versus all-pole spectrum estimation.

It is interesting to note that when the actual process that generated the speech signal is close to (3.2), the *prediction residual* $e[n] = s[n] - \tilde{s}[n]$ should be close to the scaled excitation signal $G\ u[n]$. Thus, the residual signal can be used for extracting voice-source related features. In speech recognition, the residual signal is considered as noise, but it has been shown to contain some speaker related information [206, 61, 175]. For instance, in [175], fine structure of the glottal waveform is estimated by finding the parameters of a parametric glottal flow model.

Selection of correct analysis order is crucial [145]. For a low-order analysis, say $p = 4, \ldots, 10$, the LP envelope represent mainly linguistic information, and due to low dimensionality, the discrimination between speakers is low. For higher orders, say $p > 15$, the LP spectrum represents a mixture of linguistic and speaker information. Although increasing the order makes speaker differences more apparent, for too high an order, LP model starts to capture individual harmonic peaks, and the model becomes more close to Fourier spectrum.

### 3.4.2 LP-Based Features

In addition to the predictor coefficients, the Levinson-Durbin algorithm produces intermediate variables called *reflection coefficients* $k[i], i = 1, \ldots, p$ as a side product. These are interpreted as the reflection coefficients between the tubes in the lossless tube model of the vocal tract [45]. From the reflection coefficients, *log area ratios* (LAR) or *arcus sine reflection coefficients* [27] can be also computed. Formant frequencies and bandwidths can be estimated from the poles $z_1, \ldots, z_p$ of the transfer function as follows [45]:

$$\begin{aligned} \hat{F}_i &= \frac{F_s}{2\pi} \tan^{-1} \left( \frac{\text{Im } z_i}{\text{Re } z_i} \right) && (3.6) \\ \hat{B}_i &= -\frac{F_s}{\pi} \ln |z_i|. && (3.7) \end{aligned}$$

Among a large number of parameters, LPC-derived formant frequencies were experimentally studied in [110]. Formants were observed to perform slightly poorer compared to other spectral features, but they are nevertheless an interesting feature set. The filterbank and cepstral features are *continuous* parameters describing the distribution of amplitudes of all frequencies. Formants, on the other hand, are a discrete parameter set that picks discrete feature points from the spectrum, the *locations* of resonances, and not their amplitudes.

Given the predictor coefficients $a_k$, *linear predictive cepstral coefficients* (LPCC) can be computed as follows [94]:

$$c_n = \begin{cases} a_n + \sum_{k=1}^{n-1} \frac{k}{n} c_k a_{n-k}, & 1 \leq n \leq p \\ \\ \sum_{k=n-p}^{n-1} \frac{k}{n} c_k a_{n-k}, & n > p. \end{cases} \tag{3.8}$$

An equivalent presentation of the predictor coefficients are so-called *line spectral frequencies* (LSF) [45, 94, 68]. Unlike other LP-based features listed here, LSFs have a special property of being ordered according to frequency. In other words, LSFs are not fullband features, and some LSFs can be still usable if some frequency bands are contaminated by noise. LSFs have been applied to speaker recognition in [132, 131, 27, 226, 152, 110].

*Perceptual linear prediction* (PLP) [88] exploits three psychoacoustic principles, namely, critical band analysis (Bark), equal loudness pre-emphasis, and intensity-loudness relationship. PLP and its variants have been used succesfully in speaker

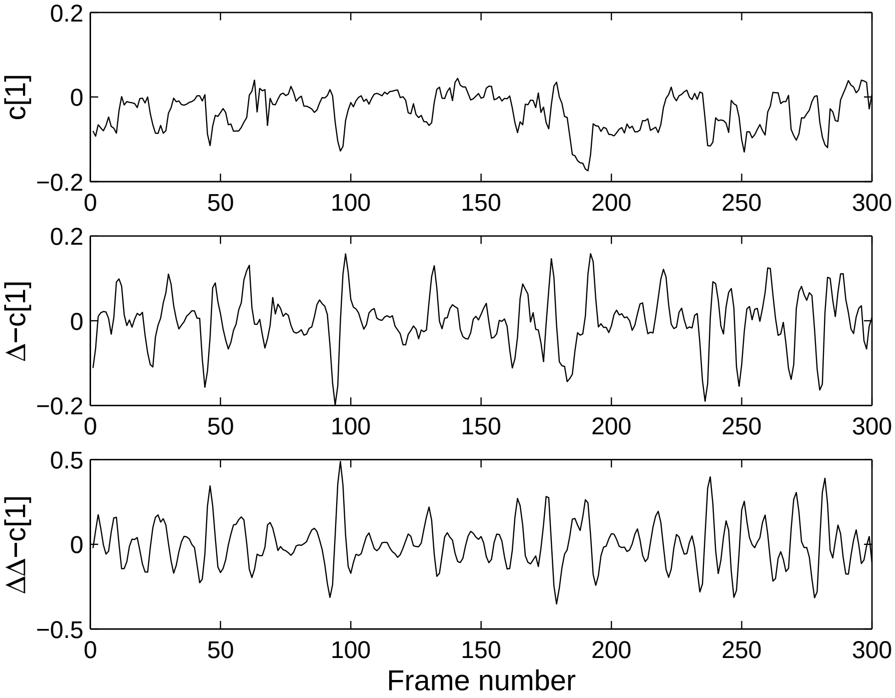


Figure 3.6: Time trajectories of first MFCC coefficient ($c_1$) and its delta and double-delta coefficients.

recognition [220, 159, 181, 211, 79]. Scanning the literature, it seems that conventional features like MFCC can outperform PLP in clean environment, but PLP gives better results in noisy and mismatched conditions.

Atal [13] compared the performance of the LPCC parameters with the following parameters for speaker recognition: LPC coefficients, impulse response of the filter specified by the LPC coefficients, autocorrelation function, and area function. From these features, the LPC cepstral coefficients performed the best. Unfortunately, Atal's data consists only of 10 speakers. In [110], a large number of DFT- and LP-derived spectral features were experimentally compared using two corpora of 110 speakers (Finnish) and 100 speakers (English). Cube root compressed filterbank cepstral coefficients, LPCC, LSF and arcus sine reflection coefficients performed the best when modeled using vector quantization.

## 3.5 Dynamic Features

While speaking the articulators make gradual movements from a configuration to another one, and these movements are reflected in the spectrum. The rate of these spectral changes depends on the speaking style, speaking rate and speech context. Some of these dynamic spectral parameters are clearly indicators of the speaker itself.

So-called *delta features* [66, 201] are the most widely used method for estimating feature dynamics. They give an estimate of the time derivative of the features they are applied to, and they can be estimated by differentiating or by polynomial representations [66, 201]. Figure 3.6 shows an example of the time trajectory of the first MFCC, and the first two derivatives estimated using linear regression over ± 2 frames.

The boundaries in delta processing can be handled by adding extra frames in both ends filled with zeroes, random numbers, or copies of adjacent frames [96]. If higher order derivatives are estimated, the boundaries should be handled with more care since the error accumulates each time the deltas are computed from the previous deltas. It can be noted also that different window lengths should be used for different coefficients, simply because they have different variance [96].

Delta processing is a linear filtering (convolution) in the feature domain, and it would be possible to design more general filters designed to emphasize speaker differences. Importance of modulation frequencies for speaker recognition have been studied in [212], but somewhat surprisingly, there are not many speaker recognition studies in which modulation spectrum would have been used. So-called *RelAtive SpecTrA* (RASTA) processing [89] aims suppressing modulation frequencies that are not important for human hearing. RASTA and related methods have been used for speaker recognition in [181, 79, 159]. In [135], time-frequency features are modeled using principal component analysis. Spectral vectors from a time context of $\pm q$ frames are concatenated into a single vector of dimensionality $(2q + 1)p$, where $p$ is the dimensionality of the original vectors, and PCA is used for reducing the dimensionality. Nonlinear dynamic features have been proposed in [173].

## 3.6 Prosodic and High-Level Features

*Prosodics* refers to non-segmental aspects of speech, including for instance syllable stress, intonation patterns, speaking rate and rhythm. Prosodic features are also called *suprasegmental* features. The main acoustic correlates of prosodic phenomena are fundamental frequency and intensity, which are more or less easily voluntarily controlled by the speaker (see [11] for an imitation study). However, they have shown to be robust against noise and channel effects [30, 124] and experiments have shown that they can complement spectral features [30, 198, 100], especially when the SNR is low.

Pitch information can be also used for noise-robust feature extraction [122]. SNR at the pitch harmonics can be assumed to be higher than on the valleys of the spectrum, and the authors in [122] model harmonic structure by Gaussian pulses whose parameters are estimated, and a noise-free spectrum is estimated as the sum of the pulses. From the conditioned spectrum, MFCCs were extracted, and improvement were obtained in very noisy and mismatched conditions.

In [140, 8] separate GMMs are used for unvoiced and voiced frames. In [140], intercorrelation of $F_0$ and spectral features is modeled by appending $F_0$ to voiced vectors. For the unvoiced case, the original features are used and thus the feature spaces for the two cases have different dimension. In [54], pitch axis is split into four experimentally defined intervals, and for each pitch class, a separate GMM on MFCC features is trained.

Atal utilized pitch contours for text-depended recognition already in 1972 [12] by applying PCA to smoothed pitch contours. In text-independent studies, long-term $F_0$ statistics, especially mean and median have been studied [139, 156, 30]. In [30], mean, variance, skew and kurtosis were used for parameterizing the distributions of $F_0$, energy, and their first two time derivatives. $F_0$ statistics can be matched using simple Euclidean distance, or by divergence [30, 198, 199]. In [30] the divergence was reported to be more accurate.

Sometimes the logarithm of $F_0$ is used instead of $F_0$ [198, 38]. In [38] it was experimentally found out that $\log F_0$ yielded smaller EERs for a Cantonese database. In [198], it is theoretically shown that $\log F_0$ follows normal distribution under some general assumptions (high correlation between successive pitch periods).

Temporal aspects of pitch have been considered in [124, 199, 2]. In [124], the authors simply divide the pitch track into fixed-length segments considered as vectors, which were modeled using the vector quantization approach. Unfortunately, the test material was small (18 speakers), and it was not reported what type of text or language was used. It is likely that the fixed-length segmentation poses problems with other data sets because the vector components are in arbitrary order.

In [199, 2], each voiced segment is parameterized, or *stylized*, by a piecewise linear model, which has two advantages. First, it removes noisy microperturbations of $F_0$ from the general trend, and second, it reduces the amount of data. Thus, contour stylization is feature extraction from the original contour. In [199], median and slope of the stylized contour were extracted, as well as the durations of line segments, voiced segments, and pauses. The median $\log F_0$ and segment slope were modeled using Gaussian distribution, and the duration features with an exponential distribution.

Recently, so-called *high-level* features have reached attention in speaker recognition after the discussion was initiated by Doddington [48]. The idea is to model symbolic information captured by symbol $N$-grams, such as characteristic word usage. For instance, speaker might habitually use phrases like “uh oh” or “well yeah” in conversations. Some examples of symbolic information modeling include word usage [48], prosodics [2, 37], phone sequences [7], and UBM component indices [218].

# Chapter 4

# Summary of the Publications

IN **the first paper** [**P1**], five unsupervised codebook generation algorithms in VQ-based speaker identification are experimentally compared. In addition to the widely used $K$-means algorithm, two hierarchical methods (Split, PNN), self-organizing map (SOM) and randomized local search (RLS) are studied. For the experiments, a database of 25 voluntary participants was collected, consisting of university staff and students. The results indicate that there is not much difference between the methods, and K-means is a good choice in general. Even randomly selected code vectors produce acceptable results (one misclassified speaker) for codebook sizes 128-256.

The result is supported by the observations in another publication [118], in which the clustering structure of short-term spectral features was studied using variance ratio based clustering validity index and principal component analysis. No clear clustering structure was observed, and for this reason, the role of the clustering algorithm is more or less sampling the training data rather than clustering it.

**In the second paper** [**P2**], an alternative front-end to the conventional MFCC processing is proposed, with two major differences. Firstly, conventional MFCC processing treats every frame in a similar manner ignoring phonetic information. In the proposed method, each broad phonetic class is processed differently. Secondly, filterbank is not based on psychoacoustic principles but on the speaker discriminating power of the phonetic class-subband pairs.

The broad phonetic classes are found using unsupervised clustering, which has an advantage that the method can be optimized for different databases and languages without the need for annotated data. In the matching phase, vector quantization is applied for labeling each frame. The experiments on a subset of the TIMIT corpus indicate that the proposed method can decrease the error rate from 38 % to 25 %

compared to conventional MFCC features for a test sample of 1 second. This shows the potential of the proposed approach for very short test segments.

**In the third paper** [**P3**], classifier fusion in a multiparametric speaker profile approach is studied. Distance-based classifier outputs are combined using weighted sum, and different weight assignment methods and feature sets are compared on a database of 110 native Finnish speakers. The proposed scheme is designed for combining diverse feature sets of arbitrary scales, number of vectors and dimensionalities.

Regarding the individual feature sets, the experiments indicate the potential of LTAS and MFCC, giving error rates of 5.4 % and 6.4 % for a test segment of length 1.8 seconds. By combining LTAS, MFCC and $F_0$, the error rate is decreased to 2.7 %. This shows the potential of the proposed fusion approach for relatively short test segments. From the weight assignment methods considered, Fisher's criterion is a practical choice.

**In the fourth paper** [**P4**], the classifier fusion approach is further studied with two goals in mind. Firstly, the complementariness of commonly used short-term spectral feature sets is addressed. Secondly, different combination levels of classifiers is studied: feature level fusion (concatenation), score level fusion (sum rule with equal weights), and decision level fusion (majority voting).

A single spectral feature set (MFCC or LPCC) is usually combined with the delta parameters, prosodic features or other high-level features. Another recent approach has been combining partial match scores from subband classifiers. However, there are few studies dealing with the combination of different *fullband* feature sets systematically. In this study, MFCC, LPCC, arcus sine reflection coefficients, formant frequencies, and the corresponding delta parameters are combined, yielding 8 feature sets. Individually best feature set on a subset of the NIST-1999 corpus is LPCC, giving 16.0 % error rate. The fusion gives slightly better results (14.6 - 14.7 %) if all the subclassifiers are reliable, but the accuracy degrades if the combined classifiers perform poorly. Majority voting is more resistant to errors in the individual classifiers, and gives the best result (12.6 %) when used for combining all the 8 feature sets. This shows that a simple combination strategy can work if there are enough classifiers.

**In the fifth paper** [**P5**], computational complexity of speaker recognition is addressed. Recognition accuracy has been widely addressed in the literature, but the number of studies dealing with time optimization directly is small. Speaker identification from a large database is a challenging task itself, and the aim of the study is to further speed it up.

Both the number of test vectors and the number of speaker models is reduced to decrease the number of distance calculations. An efficient nearest neighbor search structure is also applied in VQ matching. The methods are formulated for the VQ model, but they can be adopted to GMM as demonstrated by the experiments.

The number of speakers is reduced by iteratively pruning out poor-scoring speakers. Three novel pruning variants are proposed: static pruning, adaptive pruning, and confidence-based pruning, and the results are compared with the hierarchical pruning proposed in [165]. According to the experiments on the NIST-1999 corpus, adaptive pruning yields the best time-error tradeoff, giving speedup factors up to 12:1 with modest degradation in accuracy (17.3 % → 19.4 %).

The number of test vectors is reduced by simple decimation and clustering methods (prequantization). The experiments indicate that $K$-means clustering of the test sequence is efficient, especially for GMM. For the laboratory quality TIMIT, prequantization and pruning could be also combined, but this was not successful for the telephone quality NIST corpus. On the other hand, for the NIST corpus, simple prequantization combined with normal GMM scoring yielded a speed-up of 34:1 with degradation 16.9 % → 18.5 %.

Prequantization is also applied for speeding up *unconstrained cohort normalization* (UCN) method [9] for speaker verification. A speed-up of 23:1 was obtained without degradation in EER, giving an average processing time of less than 1 second for a 30 second test sample on the current implementation.

**In the sixth paper** [**P6**], the problem of *cohort model selection* for match score normalization is addressed. In literature, a number of heuristic cohort model selection approaches have been proposed, and there has been controversy over which method should be used. Cohort normalization has been less popular compared to the widely used world model (UBM) normalization, probably because of the difficulties and ambiguities in the selection of the cohort models.

The problem is attacked by optimizing the cohort sets for a given cost function using a genetic algorithm (GA), and by analyzing the cohort sets for the given security-convenience tradeoff. The motivation is not to present a practical selection algorithm, but to analyze the results of the optimized cohorts, and to provide an estimate of the accuracy obtainable by tuning score normalization only.

The main finding of the paper is that there is a lot of room for improving the selection heuristics, especially at the user-convenient end of the error tradeoff curve. Experiments on a subset of the NIST-1999 corpus show that for a FAR $\leq 3\%$, the best heuristic methods yields a FRR of 10.2 %. For a FRR $\leq 3\%$, the best heuristic yields FAR of 31.6 %. The “oracle” selection scheme implemented using GA suggests that it would be possible to reduce these numbers down to FRR = 2.0 % and FAR = 2.7 %.

In comparison of the UBM and cohort approaches, they perform similarly at the user-convenient and EER regions. However, at the secure end, the cohort selection is more accurate. Regarding the design parameters for the cohort approach, larger size is better in general. Even randomly selected cohorts give tremendous improvement to the baseline if the cohort size is large enough. From the studied normalization formula, arithmetic mean is the preferred choice because it has the smallest variance and good performance in overall. In a user-convenient application, the cohort speakers should be selected closer to the target speaker than in secure applications. In particular, it is advantageous to include speaker into his own cohort.

**The contributions** of the thesis can be summarized as follows. The author of this thesis has analyzed and proposed improvements to feature extraction [**P2**], modeling and matching [**P1**, **P5**], multi-feature fusion [**P3**, **P4**], and score normalization [**P6**]. The author of the thesis is the principal author of all publications, and responsible for the ideas presented. In [**P2**] the author also implemented the proposed method and run the experiments. In [**P1**, **P4**], the author implemented the feature extraction scripts.

# Chapter 5

# Summary of the Results

IN this chapter, main results of the original publications [**P1**]-[**P6**] are summarized and compared with the results obtained in literature.

## 5.1 Data Sets

In the experimental part of the original publications, five different data sets were used (see Table 5.1). Four of the datasets are recorded in laboratory environments, and present highly controlled conditions, whereas the fifth dataset includes conversational speech recorded over telephone line. Examples of speech samples from the TIMIT and NIST-1999 corpora are shown in Fig. 5.1.

Table 5.1: Summary of the data sets.

| Description | Self-collected | subset of TIMIT | TIMIT | Helsinki | subset of NIST-1999 |
|---|---|---|---|---|---|
| Language | Finnish | English | English | Finnish | English |
| Speakers | 25 | 100 | 630 | 110 | 207 |
| Speech type | Read | Read | Read | Read | Conversat. |
| Record. condit. | Lab | Lab | Lab | Lab | Teleph. |
| Handset mismatch | No | No | No | No | No |
| Sampling rate | 11.025 kHz | 8.0 kHz | 8.0 kHz | 44.1 kHz | 8.0 kHz |
| Quantization | 16-bit lin. | 16-bit lin. | 16-bit lin. | 16-bit lin. | 8-bit $\mu$-law |
| Train speech | 66 sec. | 15 sec. | 22 sec. | 10 sec. | 119 sec. |
| Test speech | 18 sec. | 1 sec. | 9 sec. | 10 sec. | 30 sec. |
| Publication where used | [**P1**] | [**P2**] | [**P5**] | [**P3**] | [**P4**, **P5**, **P6**] |

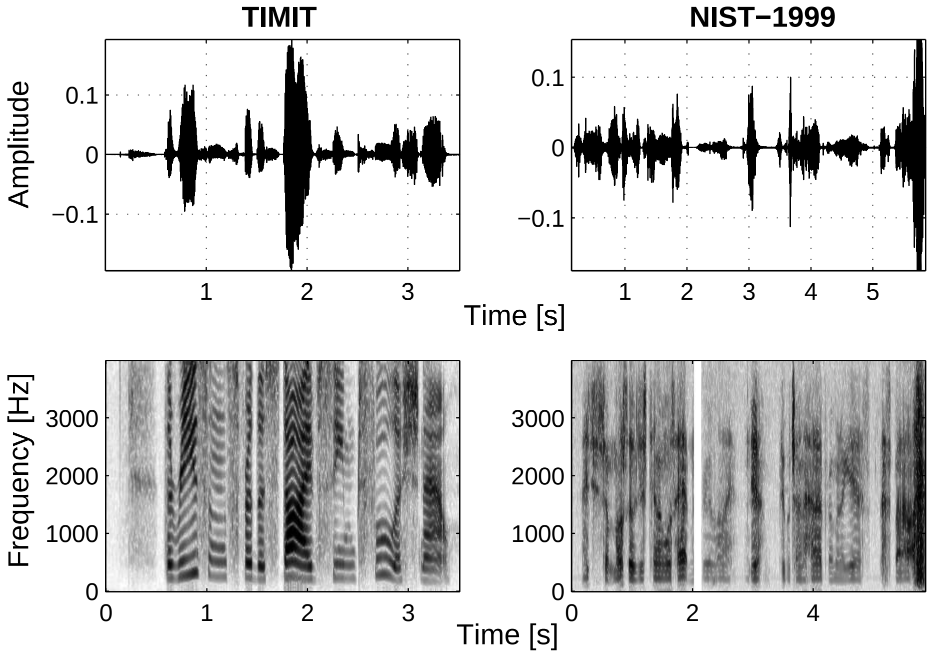


Figure 5.1: Speech samples from TIMIT (file SI2203, female) and NIST-1999 (file 4928b, male).

For the purposes of the first paper [**P1**], a small corpus was collected by recruiting voluntary participants from the university staff and students. Each speaker was prompted to read a long word list designed to include all Finnish phonemes in different contexts, as well as a few sentences from a university brochure. The word list was used as the training set, and the read sentences as the test set. The recordings took place in a normal office room, using a high-quality microphone[1] for the recordings. Slight echoing and background noise is present in the samples arising from the recording computer fans.

For the publication [**P3**], the feature sets were provided by the Department of Phonetics at the University of Helsinki, and the details of the speech material can be found in [53]. We did not have the original audio files.

For the rest of the publications, two standard corpora were used: *TIMIT* and *NIST-1999 Speaker Recognition Evaluation Corpus*, both obtainable from the Linguistic Data Consortium [130]. In the publication [**P2**], a subset of the TIMIT was used for testing; another independent subset of the TIMIT was used for tuning the parameters of the proposed method. In the publication [**P6**], the whole TIMIT cor-

[1] AKG UHF HT40 wireless microphone, `http://www.akg.com`

pus was used in the experiments, and it acted as a preliminary testbed on which the parameters of the proposed realtime algorithms were tuned. The TIMIT corpus was lowpass downsampled to 8 kHz to make it closer to telephone bandwidth.

The most challenging corpus is NIST-1999, and it was used as the testbed in the publications [**P4**, **P5**, **P6**]. The NIST corpus [142] is collected from telephone conversations between two participants who have been randomly paired by the data collection system. There are several differences to TIMIT and other laboratory-quality corpora. Firstly, the data is conversational, including turn-taking, hesitations, laughter, pauses, and simple phrases like "aha", "mmh". Secondly, the data is technically of poor quality as it is recorded over the telephone network and using several different handsets. Thirdly, there is material from several sessions, setting more challenge due to long-term changes in speaker's voice.

For all the publications where the NIST corpus is included [**P4**, **P5**, **P6**], the same subset is used. The data set consists of the male speaker data from the 1-speaker detection task in the *matched telephone line* case. This means that the training and testing telephone numbers are the same for each speaker, and for this reason, the handsets are also very likely matched [142]. However, the handset types can be different for different speakers. There are 230 male speakers in total, and 207 from these fulfill the matched telephone line case. During writing of the paper [**P5**], the authors were not aware of any studies reporting speaker *identification* results on this corpus, and the selection of the subset was therefore arbitrarily made.

The difficulty of the NIST-1999 was studied in publication [**P4**], in which eight different feature sets were combined. The distribution of correct votes is shown in Fig. 5.2. There are 54 test samples which *none* of the eight classifiers voted correctly, and 155 which all the eight classifiers voted correctly.

## 5.2 Main Results

The most interesting results (in the author's personal opinion) are summarized in Table 5.2 for each corpus. The error rates of [**P4**, **P5**, **P6**] are comparable because the same dataset has been used. The best identification result is 12.6 %, which is obtained by majority voting on the eight spectral classifiers [**P4**]. The best verification result is EER of 2.2 %, which is obtained by the "oracle" cohort selection [**P6**].

Unfortunately, due to the diversity of databases and the lack of discipline to follow accurately standard benchmark tests (also in this thesis), the recognition accuries reported here are difficult to compare directly with the literature. After scanning recent literature, we came up with a few references [184, 51, 222, 210] where subsets of the NIST-1999 corpus have been studied, also for the matched conditions

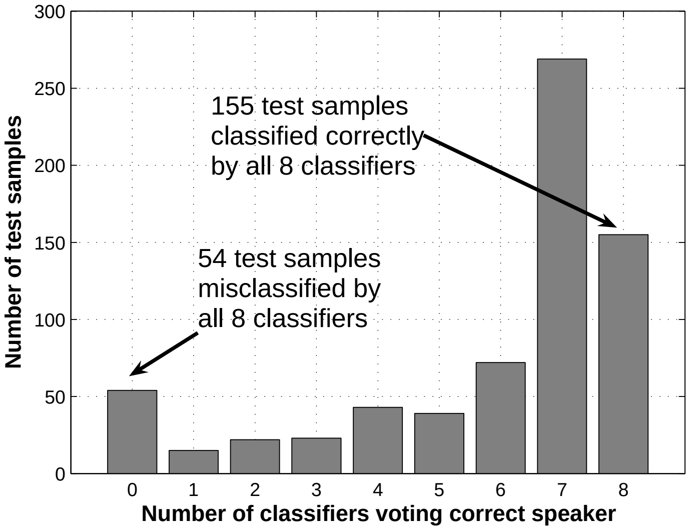


Figure 5.2: Distribution of correct votes over the 8 classifiers on the NIST-1999 corpus [**P4**].

case. The equal error rates (EER) for matched case reported in [184, 51, 222, 210] vary approximately between 5-15 %[2] . The results obtained in this thesis are at the lower end of this range, and the theoretical lower bound estimated using GA in [**P6**] (2.2 % EER) is clearly better. It is unfortunate that the identification problem has been focused much less in literature, and the author is not aware of any identification studies reported on the NIST-1999 corpus.

Regarding computational speedup by trading of the accuracy, the best identification result is 18.5 % (full search rate is 16.9 %). This is obtained by preprocessing the test sequence using K-means and performing normal GMM scoring using the obtained code vectors. This gives a speedup of 34:1 compared to the baseline GMM without prequantization. Making the processing times relative to the average test sample length of 30 seconds, the current implementation runs in 30 times realtime with a modest increase of error. Note that K-means runs relatively slow compared to the splitting method considered in [**P1**], and further speedup would therefore be likely to be obtained by replacing the K-means by the splitting method.

An interesting further question is why pre-quantization works when combined with normal GMM scoring. A possible explanation is that clustering the test sequence reduces the temporal correlations of the test vectors. The partitioning of the test vectors corresponds to segmenting the signal into homogenous units: fea-

[2] In some cases, only the error tradeoff curve is given, so the numbers are approximated from these figures.

Table 5.2: Summary of the main result for each data set. IER = identification error rate, EER = equal error rate.

| Data set | Method | Evaluation |
|---|---|---|
| Self-collected [**P1**] | $K$-means (baseline) | IER = 0.0 % |
| | PNN, Split, RLS | IER = 0.0 % |
| Subset of TIMIT [**P2**] | MFCC (baseline) | IER = 38.0 % |
| | ADFB-cepstrum | IER = 25.1 % |
| TIMIT [**P5**] | VQ full search (baseline) | IER = 0.0 % in 8.2 sec. |
| | PQP | IER = 0.0 % in 0.7 sec. |
| NIST-1999 subset [**P5**] | *Identification* | |
| | GMM full search (baseline) | IER = 16.9 % in 37.9 sec. |
| | PQ + GMM | IER = 18.5 % in 1.1 sec. |
| | *Verification* | |
| | Unconstrained cohort (baseline) | EER = 7.5 % in 18.9 sec. |
| | K-means + GMM | EER = 6.9 % in 0.8 sec. |
| Helsinki [**P3**] | LTAS (baseline) | IER = 5.5 % |
| | Score fusion | IER = 2.7 % |
| NIST-1999 subset [**P4**] | LPCC (baseline) | IER = 16.0 % |
| | Majority voting | IER = 12.6 % |
| | Oracle (*theoretical*) | IER = 7.8 % |
| NIST-1999 subset [**P6**] | GMM/UBM (baseline) | EER = 8.4 % |
| | Max. spread close (baseline) | EER = 7.2 % |
| | GA oracle (*theoretical*) | EER = 2.2 % |

Table 5.3: Comparison of computational speedup methods (IER = identification error rate, EER = equal error rate.)

| Method | Ident./<br>Verif. | Speed-up<br>factor | Effect to<br>accuracy |
|---|---|---|---|
| Hash GMM [15]<br>vs. GMM | Verif. | 10:1 | ∼EER 18 % → 18% |
| Cov. model [225]<br>vs. GMM | Verif. | 17:1 | ∼EER 18 % → 30 % |
| Structural GMM/SBM [219]<br>vs. GMM/UBM | Verif. | 17:1 | EER 12.9 % → 13.5 % |
| Decim. + GMM/UBM [105]<br>vs.GMM/UBM | Verif. | 32:1 | ∼EER 10.5 → 14.0 % |
| Vector reord. + GMM pruning [170]<br>vs. GMM | Ident. | 140:1 | 0.7 IER → 0.7 IER |
| K-means preq. + GMM [**P5**]<br>vs. GMM | Ident. | 34:1 | IER 16.9 % → 18.5 % |
| Fast cohort scoring [**P5**]<br>vs. UCN | Verif. | 23:1 | EER 7.5 % → 6.9 % |

ture vectors close in their spectral shape will be clustered together corresponding to rough phonetic classes of the unknown speaker. On the other hand, due to relatively slow articulatory movements, frames close in time are close to each other in the feature space. Thus, clustering performs a pseudo-temporal segmentation of the test sequence. The resulting code vectors will be less independent, and the independence assumption in the GMM scoring holds better. Intuitively, similar vectors to each other do not bring additional information.

The advantages of prequantizing data prior to GMM are also supported by the experiments in [105], in which a speed-up of 20:1 with minor degradation in accuracy was obtained by simple decimation of the vector sequence, i.e. using every $K$th vector.

Comparison with computational speedup methods studied in the literature are summarized in Table 5.3. Again, the numbers should be read cautiously because of different datasets, features, and measuring protocols. For [105, 15, 225], the error rates are estimated from the figures since the original publication did not contain tabulated values. It can be seen that the studied methods are competitive with the existing methods, except for the method reported in [170], in which a speed-up factor of 140:1 was obtained without degradation in identification accuracy.

# Chapter 6

# Conclusions

"A conclusion is the place where you get tired of thinking."
– Arthur Bloch

IN this thesis, text-independent speaker recognition based on spectral features has been studied. Several improvements have been achieved from both accuracy and computational points of view. Identification error rate on the NIST-1999 dataset was improved from 16.0 % to 12.6 % by using an ensemble classifier of 8 feature sets combined using majority voting. Speedup factors up to 34:1 were were obtained with a modest degradation in the accuracy. An optimization approach to the cohort model selection was proposed, and used for obtaining a lower bound to verification error rates obtainable by MFCC features modeled using GMM. The experiments indicate high potential of the cohort normalization approach, and currently used heuristics do not take full advantage of score normalization. In particular, a large gap between the theoretical error rate and the heuristics was observed at the user-convenient region.

It can be concluded that for laboratory-quality speech and controlled tasks (reading passages, prompted text), speaker recognition is a relatively easy task. For example, for the entire TIMIT corpus (630 speakers), closed-set error rate of 0.0 % can be reached with baseline methods. However, for conversational telephone-quality speech in which handset mismatch and session intervariability are present, the results degrade dramatically as seen also from the results of this thesis. This is evidenced by the fact that the studied subset of the NIST-1999 corpus includes nearly three times *less* speakers compared to TIMIT, and it has five and three times *more* training and testing data, respectively. Nevertheless, the best closed-set identification result obtained in this thesis is 12.5 %.

The data sets studied here reflect only partly the difficulty of text-independent speaker recognition. For mismatched training and recognition conditions, the recognition accuracy is known to degrade dramatically [49], but in this thesis we concentrated on the matched case only. One reason for the author's selection into this direction was that rather many research sites are already concentrating on finding engineering solutions to the mismatch problem, and there is more room in the basic research. Referring to the literature review of the thesis, the field of speaker recognition is hot, and highly multidisciplinary. A large number of interesting ideas are presented in the main forums of the field, and the good (or even bad) ideas could be combined to decrease error rates. As an example, in the SuperSID project [183], an EER of 0.2 % was reported, which was obtained by combining nine subsystem using a neural network, so that each subsystem was based on different features and methods (spectral baseline, pitch statistics, word $N$-grams, etc.)

In general, the fusion approach is promising and worth pursuing further. One possible future direction here is designing a speaker-specific fusion methodology that uses a personal feature set for each speaker, similar to feature selection described in [166]. In order for the fusion approach to be useful in real applications, reducing the computational overhead is also an important issue. When the number of classifiers is increased, the matching time will increase, and especially for the identification task, the multiple classifier approach might not be feasible in practise. A potential future direction would be realtime recognition on multiple classifiers, possibly following the pruning ideas presented in [**P5**], but generalized to multiple classifiers case.

The most serious problem with the spectral features is that they contain a mixture of linguistic, speaker-related and environment-depended factors, which cannot be easily separated. Despite of this, the features are treated by the statistical models as if they would be free of other factors, and they neglect the nature of speech data by considering it as arbitrary data. Moreover, the current spectral representations are crude mathematical models of the physical reality underlying the acoustic-articulatory inter-speaker differences. In order to understand better *what* is individual in the speech spectrum, there is a call for more basic research in speech science.

In automatic speaker recognition, machine learning approaches can be used for finding better feature representations, and this already has been proposed in a few studies [103, 146]. There is more room for studying the speaker-specific mapping approach [138, 145] and speaker-centered feature representations in general. The potential of prosodic features has not been fully exploited yet; for example, modeling *speech rhythm* and other temporal aspects of speech have reached relatively little attention.

The segmentation approach and adaptive weighting of segments based on their speaker-discriminating power would be also an interesting direction to follow. The existing segmentation approaches based on HMM and heuristic rules consider phonetically relevant segmentation. However, it is not clear what type of units are most discriminative, and it would be interesting to consider segmentation based on maximizing speaker differences. By establishing such a procedure, it would be also possible to gain some deeper understanding to the speaker-specific features of spectrum.

# References

[1] Adami, A., and Hermansky, H. Segmentation of speech for speaker and language recognition conditions. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 841–844.

[2] Adami, A., Mihaescu, R., Reynolds, D., and Godfrey, J. Modeling prosodic dynamics for speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (Hong Kong, 2003), pp. 788–791.

[3] Alexander, A., Botti, F., Dessimoz, D., and Drygajlo, A. The effect of mismatched recording conditions on human and automatic speaker recognition in forensic applications. *Forensic Science International 146S* (2004), 95–99.

[4] Alexandre, L., Campilho, A., and Kamel, M. On combining classifiers using sum and product rules. *Pattern Recognition Letters 22* (2001), 1283–1289.

[5] Altinçay, H., and Demirekler, M. An information theoretic framework for weight estimation in the combination of probabilistic classifiers for speaker identification. *Speech Communication 30* (2000), 255–272.

[6] Altinçay, H., and Demirekler, M. Speaker identification by combining multiple classifiers using Dampster-Shafer theory of evidence. *Speech Communication 41* (2003), 531–547.

[7] Andrews, W., Kohler, M., Campbell, J., and Godfrey, J. Phonetic, idiolectal, and acoustic speaker recognition. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)* (Crete, Greece, 2001), pp. 55–63.

[8] Arcienega, M., and Drygajlo, A. Pitch-dependent GMMs for text-independent speaker recognition systems. In *Proc. 7th European Conf. on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, 2001), pp. 2821–2825.

[9] Ariyaeeinia, A., and Sivakumaran, P. Analysis and comparison of score normalization methods for text dependent speaker verification. In *Proc. 5th*

*European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes, Greece, 1997), pp. 1379–1382.

[10] Ariyaeeinia, A., Sivakumaran, P., Pawlewski, M., and Loomes, M. Dynamic weighting of the distortion sequence in text-dependent speaker verification. In *Proc. 6th European Conf. on Speech Communication and Technology (Eurospeech 1999)* (Budapest, Hungary, 1999), pp. 967–970.

[11] Ashour, G., and Gath, I. Characterization of speech during imitation. In *Proc. 6th European Conf. on Speech Communication and Technology (Eurospeech 1999)* (Budapest, Hungary, 1999), pp. 1187–1190.

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

[13] 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.

[14] Auckenthaler, R., and Mason, J. Equalizing sub-band error rates in speaker recognition. In *Proc. 5th European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes, Greece, 1997), vol. 5, pp. 2303–2306.

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

[16] Auckenthaler, R., Parris, E., and Carey, M. Improving a GMM speaker verification system by phonetic weighting. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1999)* (Phoenix, Arizona, USA, 1999), vol. 1, pp. 313–316.

[17] Auckenthaler, R., Parris, E., and Carey, M. Improving a GMM speaker verification system by phonetic weighting. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1999)* (1999), pp. 313–316.

[18] Bachorowski, J.-A., and Owren, M. Acoustic correlates of talker sex and individual talker identity are present in a short vowel segment produced in running speech. *Journal of the Acoustic Society of America 106* (1999), 1054–1063.

[19] Bartkova, K., D.L.Gac, Charlet, D., and Jouvet, D. Prosodic parameter for speaker identification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)* (Denver, Colorado, USA, 2002), pp. 1197–1200.

[20] BenZeghiba, M., and Bourland, H. On the combination of speech and speaker recognition. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 1361–1364.

[21] Besacier, L., Bonastre, J., and Fredouille, C. Localization and selection of speaker-specific information with statistical modeling. *Speech Commu-*

*nication 31* (2000), 89–106.

[22] Besacier, L., and Bonastre, J.-F. Subband architecture for automatic speaker recognition. *Signal Processing 80* (2000), 1245–1259.

[23] Bimbot, F., Magrin-Chagnolleau, I., and Mathan, L. Second-order statistical measures for text-independent speaker identification. *Speech Communication 17* (1995), 177–192.

[24] the Biometric Consortium. WWW page, December 2003. `http://www.biometrics.org/`.

[25] Bishop, C. *Neural Networks for Pattern Recognition.* Oxford University Press, New York, 1996.

[26] Brunelli, R., and Falavigna, D. Person identification using multiple cues. *IEEE Trans. on Pattern Analysis and Machine Intelligence 17*, 10 (1995), 955–966.

[27] Campbell, J. Speaker recognition: a tutorial. *Proc. of the IEEE 85*, 9 (1997), 1437–1462.

[28] Campbell, J., Reynolds, D., and Dunn, R. Fusing high- and low-level features for speaker recognition. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 2665–2668.

[29] Campbell, W., Assaleh, K., and Broun, C. Speaker recognition with polynomial classifiers. *IEEE Trans. on Speech and Audio Processing 10*, 4 (May 2002), 205–212.

[30] Carey, M., Parris, E., Lloyd-Thomas, H., and Bennett, S. Robust prosodic features for speaker identification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1996)* (Philadelphia, Pennsylvania, USA, 1996), pp. 1800–1803.

[31] Castellano, P., Slomka, S., and Sridharan, S. Telephone based speaker recognition using multiple binary classifier and gaussian mixture models. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1997)* (1997), vol. 2, pp. 1075–1078.

[32] Charlet, D., and Jouvet, D. Optimizing feature set for speaker verification. *Pattern Recognition Letters 18* (1997), 873–879.

[33] Charlet, D., Jouvet, D., and Collin, O. An alternative normalization scheme in HMM-based text-dependent speaker verification. *Speech Communication 32* (2000), 113–120.

[34] Chaudhari, U., Navrátil, J., and Maes, S. Multigrained modeling with pattern specific maximum likelihood transformations for text-independent speaker recognition. *IEEE Trans. on Speech and Audio Processing 11*, 1 (2003), 61–69.

[35] Chen, K., Wang, L., and Chi, H. Methods of combining multiple classifiers with different features and their applications to text-independent speaker

recognition. *International Journal of Pattern Recognition and Artificial Intelligence 11*, 3 (1997), 417–445.

[36] Chen, K., Wu, T.-Y., and Zhang, H.-J. On the use of nearest feature line for speaker identification. *Pattern Recognition Letters 23* (2002), 1735–1746.

[37] Chen, Z.-H., Liao, Y.-F., and Juang, Y.-T. Eigen-prosody analysis for robust speaker recognition under mismatch handset environment. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2004)* (Jeju Island, Korea, 2004), pp. 1421–1424.

[38] Cheng, Y., and Leung, H. Speaker verification using fundamental frequency. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (1996), pp. Paper 0228 on the CD–ROM.

[39] Cheung, R., and Eisenstein, B. Feature selection via dynamic programming for text-independent speaker identification. *IEEE Trans. on Acoustics, Speech and Signal Processing 26* (October 1978), 397–403.

[40] Cohen, A., and Zigel, Y. On feature selection for speaker verification. In *Proc. COST 275 workshop on The Advent of Biometrics on the Internet* (2002), pp. 89–92.

[41] Cover, T., and Thomas, J. *Elements of Information Theory.* Wiley Interscience, 1991.

[42] Cristea, P., and Vâlsan, Z. New cepstrum frequency scale for neural network speaker verification. In *The 6th IEEE Int. Conf. on Electronics, Circuits and Systems (ICECS '99)* (1999), vol. 3, pp. 1573–1576.

[43] Damper, R., and Higgins, J. Improving speaker identification in noise by subband processing and decision fusion. *Pattern Recognition Letters 24* (2003), 2167–2173.

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

[45] Deller, J. J., Hansen, J., and Proakis, J. *Discrete-Time Processing of Speech Signals*, second ed. IEEE Press, New York, 2000.

[46] Demirekler, M., and Haydar, A. Feature selection using genetics-based algorithm and its application to speaker identification.

[47] Dersch, D., and King, R. Speaker models designed from complete data sets: a new approach to text-independent speaker verification. In *Proc. 5th European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes, Greece, 1997), pp. 2323–2326.

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

[49] Doddington, G., Przybocki, M., Martin, A., and D.A.Reynolds. The NIST speaker recognition evaluation - overview, methodology, systems,

results, perspective. *Speech Communication 31* (June 2000), 225–254.

[50] Duda, R., Hart, P., and Stork, D. *Pattern Classification*, second ed. Wiley Interscience, New York, 2000.

[51] Dunn, R., Quatieri, T., Reynolds, D., and Campbell, J. Speaker recognition from coded speech in matched and mismatched conditions. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)* (Crete, Greece, 2001), pp. 115–120.

[52] Eatock, J., and Mason, J. A quantitative assesment of the relative speaker discriminating properties of phonemes. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1994)* (Adelaide, Australia, 1994), pp. 133–136.

[53] Eskelinen-Rönkä, P. Raportti automaattisen Puhujan Tunnistaja - tietokantaohjelman testauksesta. MSc Thesis (in Finnish), Department of General Phonetics, University of Helsinki, Helsinki, Finland, 1997.

[54] Ezzaidi, H., Rouat, J., and O'Shaughnessy, D. Towards combining pitch and MFCC for speaker identification systems. In *Proc. 7th European Conf. on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, 2001), pp. 2825–2828.

[55] Faltlhauser, R., and Ruske, G. Improving speaker recognition performance using phonetically structured Gaussian mixture models. In *Proc. 7th European Conf. on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, 2001), pp. 751–754.

[56] Fan, N., and Rosca, J. Enhanced VQ-based algorithms for speech independent speaker identification. In *Proc. Audio- and Video-Based Biometric Authentication (AVBPA 2003)* (Guildford, UK, 2003), pp. 470–477.

[57] Fant, G. *Acoustic Theory of Speech Production.* The Hague, Mouton, 1960.

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

[59] Farrell, K., Ramachandran, R., and Mammone, R. An analysis of data fusion methods for speaker verification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1998)* (Seattle, Washington, USA, 1998), vol. 2, pp. 1129–1132.

[60] Faúndez-Zanuy, M. On the model size selection for speaker identification. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)* (Crete, Greece, 2001), pp. 189–193.

[61] Faúndez-Zanuy, M., and Rodríguez-Porcheron, D. Speaker recognition using residual signal of linear and nonlinear prediction models. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (1998), p. Paper 1102.

[62] Ferrer, L., Bratt, H., Gadde, V., Kajarekar, S., Shriberg, E.,

Sönmez, K., Stolcke, A., and Venkataraman, A. Modeling duration patterns for speaker recognition. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 2017–2020.

[63] Forsyth, M. Discriminating observation probability (DOP) HMM for speaker verification. *Speech Communication 17* (1995), 117–129.

[64] Fredouille, C., Bonastre, J.-F., and Merlin, T. AMIRAL: A block-segmental multirecognizer architecture for automatic speaker recognition. *Digital Signal Processing 10*, 1/2/3 (2000), 172–197.

[65] Fukunaga, K. *Introduction to Statistical Pattern Recognition*, second ed. Academic Press, London, 1990.

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

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

[68] Furui, S. *Digital Speech Processing, Synthesis, and Recognition*, second ed. Marcel Dekker, Inc., New York, 2001.

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

[70] Gish, H. Robust discrimination in automatic speaker identification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1990)* (Albuquerque, New Mexico, USA, 1990), pp. 289–292.

[71] Gish, H., Krasner, M., Russell, W., and Wolf, J. Methods and experiments for text-independent speaker recognition over telephone channels. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1986)* (Tokyo, Japan, 1986), pp. 865–868.

[72] Gish, H., and Schmidt, M. Text-independent speaker identification. *IEEE Signal Processing Magazine 11* (1994), 18–32.

[73] Gonzalez-Rodriguez, J., Garcia-Gomar, D. G.-R. M., Ramos-Castro, D., and Ortega-Garcia, J. Robust likelihood ratio estimation in bayesian forensic speaker recognition. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 693–696.

[74] Gopalan, K., Anderson, T., and Cupples, E. A comparison of speaker identification results using features based on cepstrum and Fourier-Bessel expansion. *IEEE Trans. on Speech and Audio Processing 7*, 3 (1999), 289–294.

[75] Gori, M., Lastrucci, L., and Soda, G. Autoassociator-based models for speaker verification. *Pattern Recognition Letters 17* (1996), 241–250.

[76] Gupta, S., and Savic, M. Text-independent speaker verification based on broad phonetic segmentation of speech. *Digital Signal Processing*, 2 (1992),

69–79.

[77] Hannani, A., Petrovska-Delacrétaz, D., and Chollet, G. Linear and non-linear fusion of ALISP-based and gmm systems for text-independent speaker verification. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2004)* (Toledo, Spain, 2004), pp. 111–116.

[78] Hansen, E., Slyh, R., and Anderson, T. Speaker recognition using phoneme-specific GMMs. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2004)* (Toledo, Spain, 2004), pp. 179–184.

[79] Hardt, D., and Fellbaum, K. Spectral subtraction and RASTA-filtering in text-dependent HMM-based speaker verification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1997)* (Munich, Germany, 1997), pp. 867–870.

[80] Harrington, J., and Cassidy, S. *Techniques in Speech Acoustics*. Kluwer Academic Publishers, Dordrecht, 1999.

[81] Harris, F. On the use of windows for harmonic analysis with the discrete fourier transform. *Proc. of the IEEE 66*, 1 (1978), 51–84.

[82] Hayakawa, S., and Itakura, F. Text-dependent speaker recognition using the information in the higher frequency band. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1994)* (Adelaide, Australia, 1994), pp. 137–140.

[83] He, J., Liu, L., and Palm, G. A discriminative training algorithm for VQ-based speaker identification. *IEEE Trans. on Speech and Audio Processing 7*, 3 (1999), 353–356.

[84] Hébert, M., and Heck, L. Phonetic class-based speaker verification. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 1665–1668.

[85] Heck, L., and Genoud, D. Combining speaker and speech recognition systems. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)* (Denver, Colorado, USA, 2002), pp. 1369–1372.

[86] Heck, L., Konig, Y., Sönmez, M., and Weintraub, M. Robustness to telephone handset distortion in speaker recognition by discriminative feature design. *Speech Communication 31* (2000), 181–192.

[87] Hedge, R., Murthy, H., and Rao, G. Application of the modified group delay function to speaker identification and discrimination. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2004)* (2004), vol. 1, pp. 517–520.

[88] Hermansky, H. Perceptual linear prediction (PLP) analysis for speech. *Journal of the Acoustic Society of America 87* (1990), 1738–1752.

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

[90] Hernando, J., and Nadeu, C. Speaker verification on the polycost data-

base using frequency filtered spectral energies. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (Sydney, Australia, 1998), pp. 129–132.

[91] Higgins, A., and Bahler, L. Text-independent speaker verification by discriminator counting. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1991)* (1991), vol. 1, pp. 405–408.

[92] Higgins, A., Bahler, L., and Porter, J. Speaker verification using randomized phrase prompting. *Digital Signal Processing 1* (1991), 89–106.

[93] Higgins, A., Bahler, L., and Porter, J. Voice identification using nearest-neighbor distance measure. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1993)* (Minneapolis, Minnesota, USA, 1993), pp. 375–378.

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

[95] Huggins, M., and Grieco, J. Confidence metrics for speaker identification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)* (Denver, Colorado, USA, 2002), pp. 1381–1384.

[96] Hume, J. Wavelet-like regression features in the cepstral domain for speaker recognition. In *Proc. 5th European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes, Greece, 1997), pp. 2339–2342.

[97] Hyvärinen, A., Karhunen, J., and Oja, E. *Independent Component Analysis.* John Wiley & Sons, Inc., New York, 2001.

[98] Ifeachor, E., and Lewis, B. *Digital Signal Processing - a Practical Approach*, second ed. Pearson Education Limited, Edinburgh Gate, 2002.

[99] Imperl, B., Kacic, Z., and Horvat, B. A study of harmonic features for the speaker recognition. *Speech Communication 22* (1997), 385–402.

[100] Iwano, K., Asami, T., and Furui, S. Noise-robust speaker verification using $f_0$ features. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2004)* (2004), vol. 2, pp. 1417–1420.

[101] Jain, A., R.P.W.Duin, and J.Mao. Statistical pattern recognition: a review. *IEEE Trans. on Pattern Analysis and Machine Intelligence 22* (2000), 4–37.

[102] Jain, A., and Zongker, D. Feature selection: evaluation, application, and small sample performance. *IEEE Trans. on Pattern Analysis and Machine Intelligence 19* (1997), 153–158.

[103] Jang, G.-J., Lee, T.-W., and Oh, Y.-H. Learning statistically efficient features for speaker recognition. *Neurocomputing 49* (2002), 329–348.

[104] Jang, G.-J., Yun, S.-J., and Oh, Y.-H. Feature vector transformation using independent component analysis and its application to speaker identification. pp. 767–770.

[105] J.McLaughlin, Reynolds, D., and Gleason, T. A study of computa-

tion speed-ups of the GMM-UBM speaker recognition system. In *Proc. 6th European Conf. on Speech Communication and Technology (Eurospeech 1999)* (Budapest, Hungary, 1999), pp. 1215–1218.

[106] Kajarekar, S., and Hermansky, H. Speaker verification based on broad phonetic categories. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)* (Crete, Greece, 2001), pp. 201–206.

[107] Karpov, E., Kinnunen, T., and Fränti, P. Symmetric distortion measure for speaker recognition. In *Proc. 9th Int. Conf. Speech and Computer (SPECOM'2004)* (St. Petersburg, Russia, 2004), pp. 366–370.

[108] Kim, S., Eriksson, T., Kang, H.-G., and Youn, D. A pitch-synchronous feature extraction method for speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2004)* (2004), vol. 1, pp. 405–408.

[109] Kinnunen, T. Designing a speaker-discriminative filter bank for speaker recognition. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)* (Denver, Colorado, USA, 2002), pp. 2325–2328.

[110] Kinnunen, T. *Spectral Features for Automatic Text-Independent Speaker Recognition.* Licentiate's thesis, University of Joensuu, Department of Computer Science, Joensuu, Finland, 2004.

[111] Kinnunen, T., and Fränti, P. Speaker discriminative weighting method for VQ-based speaker identification. In *Proc. Audio- and Video-Based Biometric Authentication (AVBPA 2001)* (Halmstad, Sweden, 2001), pp. 150–156.

[112] Kinnunen, T., Hautamäki, V., and Fränti, P. On the fusion of dissimilarity- based classifiers for speaker identification. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 2641–2644.

[113] Kinnunen, T., Hautamäki, V., and Fränti, P. Fusion of spectral feature sets for accurate speaker identification. In *Proc. 9th Int. Conf. Speech and Computer (SPECOM'2004)* (St. Petersburg, Russia, 2004), pp. 361–365.

[114] Kinnunen, T., Karpov, E., and Fränti, P. Real-time speaker identification and verification. *IEEE Trans. on Speech and Audio Processing*, Accepted for publication.

[115] Kinnunen, T., Kilpeläinen, T., and Fränti, P. Comparison of clustering algorithms in speaker identification. In *Proc. IASTED Int. Conf. Signal Processing and Communications (SPC 2000)* (Marbella, Spain, 2000), pp. 222–227.

[116] Kinnunen, T., and Kärkkäinen, I. Class-discriminative weighted distortion measure for VQ-based speaker identification. In *Proc. Joint IAPR International Workshop on Statistical Pattern Recognition (S+SPR2002)* (Windsor, Canada, 2002), pp. 681–688.

[117] Kinnunen, T., Kärkkäinen, I., and Fränti, P. Mystery of cohort selec-

tion. *IEEE Trans. on Speech and Audio Processing*, Manuscript, submitted for publication.

[118] Kinnunen, T., Kärkkäinen, I., and Fränti, P. Is speech data clustered? - statistical analysis of cepstral features. In *Proc. 7th European Conf. on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, 2001), pp. 2627–2630.

[119] Kittler, J., Hatef, M., Duin, R., and Matas, J. On combining classifiers. *IEEE Trans. on Pattern Analysis and Machine Intelligence 20*, 3 (1998), 226–239.

[120] Kohonen, T. *Self-Organizing Maps*, third extended ed. Springer-Verlag, Berlin, 2001.

[121] Kolano, G., and Regel-Brietzmann, P. Combination of vector quantization and gaussian mixture models for speaker verification. In *Proc. 6th European Conf. on Speech Communication and Technology (Eurospeech 1999)* (Budapest, Hungary, 1999), pp. 1203–1206.

[122] Krishnakumar, S., Kumar, K. P., and Balakrishnan, N. Pitch maxima for robust speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (2003), vol. 2, pp. 201–204.

[123] Kwon, O.-W., Chan, K., Hao, J., and Lee, T.-W. Emotion recognition by speech signals. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 125–128.

[124] Kyung, Y., and Lee, H.-S. Text independent speaker recognition using microprosody. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (Sydney, Australia, 1998).

[125] Lapidot, I., Guterman, H., and Cohen, A. Unsupervised speaker recognition based on competition between self-organizing maps. *IEEE Transactions on Neural Networks 13* (2002), 877–887.

[126] Lee, C., Yildirim, S., Bulut, M., Kazemzadeh, A., Busso, C., Deng, Z., Lee, S., and Narayanan, S. Emotion recognition based on phoneme classes. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2004)* (Jeju Island, Korea, 2004), pp. 889–892.

[127] Li, K.-P., and Porter, J. Normalizations and selection of speech segments for speaker recognition scoring. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1988)* (New York, 1988), pp. 595–598.

[128] Li, Q., Juang, B.-H., and Lee, C.-H. Automatic verbal information verification for user authentication. *IEEE Trans. on Speech and Audio Processing 8* (2000), 585–596.

[129] Linde, Y., Buzo, A., and Gray, R. An algorithm for vector quantizer design. *IEEE Transactions on Communications 28*, 1 (1980), 84–95.

[130] Linguistic data consortium. WWW page, September 2004. `http://www.ldc.upenn.edu/`.

[131] Liu, C.-S., Huang, C.-S., Lin, M.-T., and Wang, H.-C. Automatic speaker recognition based upon various distances of LSP frequencies. In *Proc. 25th Annual 1991 IEEE International Carnahan Conference on Security Technology* (1991), pp. 104–109.

[132] Liu, C.-S., Wang, W.-J., Lin, M.-T., and Wang, H.-C. Study of line spectrum pair frequencies for speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1990)* (Albuquerque, New Mexico, USA, 1990), pp. 277–280.

[133] Liu, L., He, J., and Palm, G. A comparison of human and machine in speaker recognition. In *Proc. 5th European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes, Greece, 1997), pp. 2327–2330.

[134] Louradour, J., André-Obrecht, R., and Daoudi, K. Segmentation and relevance measure for speaker verification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2004)* (2004), pp. 1401–1404.

[135] Magrin-Chagnolleau, I., Duroy, G., and Bimbot, F. Application of time-frequency principal component analysis to text-independent speaker identification. *IEEE Trans. on Speech and Audio Processing 10*, 6 (September 2002), 371–378.

[136] Mak, M.-W., Cheung, M.-C., and Kung, S.-Y. Robust speaker verification from GSM-trascoded speech based on decision fusion and feature transformation. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (Hong Kong, 2003), pp. 745–748.

[137] Makhoul, J. Linear prediction: a tutorial review. *Proc. of the IEEE 64*, 4 (1975), 561–580.

[138] Malayath, N., Hermansky, H., Kajarekar, S., and Yegnanarayana, B. Data-driven temporal filters and alternatives to GMM in speaker verification. *Digital Signal Processing 10* (2000), 55–74.

[139] Markel, J., Oshika, B., and A.H. Gray, j. Long-term feature averaging for speaker recognition. *IEEE Trans. Acoustics, Speech, and Signal Processing 25*, 4 (1977), 330–337.

[140] Markov, K., and Nakagawa, S. Text-independent speaker recognition using multiple information sources. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (Sydney, Australia, 1998), pp. 173–176.

[141] Markov, K., and Nakagawa, S. Text-independent speaker recognition using non-linear frame likelihood transformation. *Speech Communication 24* (1998), 193–209.

[142] Martin, A., and Przybocki, M. The NIST 1999 speaker recognition evaluation - an overview. *Digital Signal Processing 10* (2000), 1–18.

[143] Matsui, T., and Furui, S. A text-independent speaker recognition method robust against utterance variations. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1991)* (Toronto, Canada, 1991), pp. 377–380.

[144] Ming, J., Stewart, D., Hanna, P., Corr, P., Smith, J., and Vaseghi, S. Robust speaker identification using posterior union models. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 2645–2648.

[145] Misra, H., Ikbal, S., and Yegnanarayana, B. Speaker-specific mapping for text-independent speaker recognition. *Speech Communication 39* (2003), 301–310.

[146] Miyajima, C., Watanabe, H., Tokuda, K., Kitamura, T., and Katagiri, S. A new approach to designing a feature extractor in speaker identification based on discriminative feature extraction. *Speech Communication 35* (2001), 203–218.

[147] Mokhtari, P., Clermont, F., and Tanaka, K. Toward an acoustic-articulatory model of inter-speaker variability. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2000)* (Beijing, China, 2000), vol. 2, pp. 158–161.

[148] Moonasar, V., and Venayagamoorthy, G. A committee of neural networks for automatic speaker recognition (asr) systems. In *Proc. Int. Joint Conference on Neural Networks (IJCNN 2001)* (Washington, D.C., USA, 2001), pp. 2936–2940.

[149] Moreno, P., and Purdy, P. A new SVM approach to speaker identification and verification using probabilistic distance kernels. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 2965–2968.

[150] Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K., and Schölkopf, B. An introduction to kernel-based learning algorithms. *IEEE Trans. on Neural Networks 12* (2001), 181–202.

[151] Nakasone, H. Automated speaker recognition in real world conditions: Controlling the uncontrollable. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 697–700.

[152] Nguyen, P., Akagi, M., and Ho, T. Temporal decomposition: a promising approach to VQ-based speaker identification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (Hong Kong, 2003).

[153] Niemi-Laitinen, T. National Bureau of Investigation (personal communication), 2005.

[154] Niemi-Laitinen, T., Saastamoinen, J., Kinnunen, T., and Fränti, P. Applying MFCC-based automatic speaker recognition to gsm and forensic data. In *Accepted for publication in* Human Language Technologies *(HLT 2005).*

[155] Nishida, M., and Ariki, Y. Speaker recognition by separating phonetic space and speaker space. In *Proc. 7th European Conf. on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, 2001), pp. 1381–

1384.

[156] Nolan, F. *The Phonetic Bases of Speaker Recognition.* Cambridge University Press, Cambridge, 1983.

[157] Olsen, J. A two-stage procedure for phone-based speaker verification. pp. 889–897.

[158] Olsen, J. Speaker verification with ensemble classifiers based on linear speech transforms. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (1998), p. Paper 0334.

[159] Openshaw, J., Sun, Z., and Mason, J. A comparison of composite features under degraded speech in speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1993)* (Minneapolis, Minnesota, USA, 1993), pp. 27–30.

[160] Oppenheim, A., and Schafer, R. *Digital Signal Processing.* Prentice Hall, Englewood Cliffs, 1975.

[161] Orman, D., and Arslan, L. Frequency analysis of speaker identification. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)* (Crete, Greece, 2001), pp. 219–222.

[162] Padrta, A., and Radová, V. On the amount of speech data necessary for successful speaker identification. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 3021–3024.

[163] Padrta, A., and Radová, V. Comparison of several speaker verification procedures based on GMM. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2004)* (Jeju Island, Korea, 2004), pp. 1777–1780.

[164] Paliwal, K., and Alsteris, L. Usefulness of phase spectrum in human speech perception. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 2117–2120.

[165] Pan, Z., Kotani, K., and Ohmi, T. An on-line hierarchical method of speaker identification for large population. In *NORSIG 2000* (Kolmården, Sweden, 2000).

[166] Pandit, M., and Kittler, J. Feature selection for a DTW-based speaker verification system. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1998)* (Seattle, Washington, USA, 1998), vol. 2, pp. 769–772.

[167] Park, A., and Hazen, T. ASR dependent techniques for speaker identification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)* (Denver, Colorado, USA, 2002), pp. 1337–1340.

[168] Parris, E., and Carey, M. Discriminative phonemes for speaker identification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1994)* (Yokohama, Japan, 1994), pp. 1843–1846.

[169] Pelecanos, J., Myers, S., Sridharan, S., and Chandran, V. Vec-

tor quantization based gaussian mixture modeling for speaker verification. In *Proc. Int. Conf. on Pattern Recognition (ICPR 2000)* (Barcelona, Spain, 2000), pp. 3298–3301.

[170] Pellom, B., and Hansen, J. An efficient scoring algorithm for gaussian mixture model based speaker identification. *IEEE Signal Processing Letters 5*, 11 (1998), 281–284.

[171] Peskin, B., Navratil, J., Abramson, J., Jones, D., Klusacek, D., Reynolds, D., and Xiang, B. Using prosodic and conversational features for high-performance speaker recognition: report from JHU WS'02. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (Hong Kong, 2003), pp. 792–795.

[172] Petrovska-Delacrétaz, D., Cernocký, J., Hennebert, J., and Chollet, G. Segmental approaches for automatic speaker verification. *Digital Signal Processing 10*, 1 (2000), 198–212.

[173] Petry, A., and Barone, D. Text-dependent speaker verification using Lyapunov exponents. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)* (Denver, Colorado, USA, 2002), pp. 1321–1324.

[174] Pfister, B., and Beutler, R. Estimating the weight of evidence in forensic speaker verification. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 701–704.

[175] Plumpe, M., Quatieri, T., and Reynolds, D. Modeling of the glottal flow derivative waveform with application to speaker identification. *IEEE Trans. on Speech and Audio Processing 7*, 5 (1999), 569–586.

[176] Prabhakar, S., Pankanti, S., and Jain, A. Biometric recognition: security and privacy concerns. *IEEE Security & Privacy Magazine 1* (2003), 33–42.

[177] Proakis, J., and Manolakis, D. *Digital Signal Prosessing. Principles, Algorithms and Applications*, second ed. Macmillan Publishing Company, New York, 1992.

[178] Rabiner, L., and Juang, B.-H. *Fundamentals of Speech Recognition.* Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[179] Ramachandran, R., Farrell, K., Ramachandran, R., and Mammone, R. Speaker recognition - general classifier approaches and data fusion methods. *Pattern Recognition 35* (2002), 2801–2821.

[180] Ren-Hua, W., Lin-Shen, H., and Fujisaki, H. A weighted distance measure based on the fine structure of feature space: application to speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1990)* (Albuquerque, New Mexico, USA, 1990), pp. 273–276.

[181] Reynolds, D. Experimental evaluation of features for robust speaker identification. *IEEE Trans. on Speech and Audio Processing 2* (1994), 639–643.

[182] Reynolds, D. Speaker identification and verification using gaussian mixture

speaker models. *Speech Communication 17* (1995), 91–108.

[183] Reynolds, D., Andrews, W., Campbell, J., Navratil, J., Peskin, B., Adami, A., Jin, Q., Klusacek, D., Abramson, J., Mihaescu, R., Godfrey, J., Jones, D., and Xiang, B. The SuperSID project: exploiting high-level information for high-accuracy speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (Hong Kong, 2003), pp. 784–787.

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

[185] Reynolds, D., and Rose, R. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Trans. on Speech and Audio Processing 3* (1995), 72–83.

[186] Rifkin, R. Speaker recognition using local models. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 3009–3012.

[187] Rodríguez-Liñares, L., and García-Mateo, C. On the use of acoustic segmentation in speaker identification. In *Proc. 5th European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes,Greece, 1997), pp. 2315–2318.

[188] Rodríguez-Liñares, L., García-Mateo, C., and Alba-Castro, J. On combining classifiers for speaker authentication. *Pattern Recognition 36* (2003), 347–359.

[189] Rose, P. *Forensic Speaker Identification.* Taylor & Francis, London, 2002.

[190] Rosenberg, A. Automatic speaker verification: a review. *Proc. of the IEEE 64*, 4 (1976), 475–487.

[191] Sambur, M. Selection of acoustic features for speaker identification. *IEEE Trans. Acoustics, Speech, and Signal Processing 23*, 2 (1975), 176–182.

[192] Sanderson, C., and Paliwal, K. Information fusion for robust speaker verification. In *Proc. 7th European Conf. on Speech Communication and Technology (Eurospeech 2001)* (Aalborg, Denmark, 2001), pp. 755–758.

[193] Schmidt-Nielsen, A., and Crystal, T. Speaker verification by human listeners: experiments comparing human and machine performance using the nist 1998 speaker evaluation data. *Digital Signal Processing 10* (2000), 249–266.

[194] Sivakumaran, P., Ariyaeeinia, A., and Loomes, M. Sub-band based speaker verification using dynamic recombination weights. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (Sydney, Australia, 1998), pp. 77–80.

[195] Sivakumaran, P., Ariyaeeinia, A., and Loomes, M. Sub-band based text-dependent speaker verification. *Speech Communication 41* (2003), 485–

509.

[196] Sivakumaran, P., Fortuna, J., and Ariyaeeinia, A. Score normalization applied to open-set, text-independent speaker identification. In *Proc. 8th European Conf. on Speech Communication and Technology (Eurospeech 2003)* (Geneva, Switzerland, 2003), pp. 2669–2672.

[197] Slomka, S., Sridharan, S., and Chandran, V. A comparison of fusion techniques in mel-cepstral based speaker idenficication. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (Sydney, Australia, 1998), pp. 225–228.

[198] Sönmez, M., Heck, L., Weintraub, M., and Shriberg, E. A lognormal tied mixture model of pitch for prosody-based speaker recognition. In *Proc. 5th European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes, Greece, 1997), pp. 1391–1394.

[199] Sönmez, M., Shriberg, E., Heck, L., and Weintraub, M. Modeling dynamic prosodic variation for speaker verification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (1998), p. Paper 0920.

[200] 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.

[201] Soong, F., and Rosenberg, A. On the use of instantaneous and transitional spectral information in speaker recognition. *IEEE Trans. on Acoustics, Speech and Signal Processing 36*, 6 (1988), 871–879.

[202] Stapert, R., and J.S.Mason. Speaker recognition and the acoustic speech space. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)* (Crete, Greece, 2001), pp. 195–199.

[203] Strang, G., and Nguyen, T. *Wavelets and filter banks.* Wellesley-Cambridge Press, Wellesley, 1996.

[204] Su, L.-S., Li, K.-P., and K.S.Fu. Identification of speakers by use of nasal coarticulation. *Journal of the Acoustic Society of America 56*, 6 (1974), 1876–1882.

[205] Tax, D., Breukelen, M., Duin, R., and Kittler, J. Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition 33* (2000), 1475–1485.

[206] Thévenaz, P., and Hügli, H. Usefulness of the LPC-residue in text-independent speaker verification. *Speech Communication 17*, 1-2 (August 1995), 145–157.

[207] Tran, D., and Wagner, M. Fuzzy C-means clustering-based speaker verification. In *Proc. Advances in Soft Computing (AFSS 2002)* (Calcutta, India, February 2002), pp. 318–324.

[208] Tran, D., Wagner, M., and VanLe, T. A proposed decision rule based on fuzzy c-means clustering for speaker recognition. In *Proc. Int. Conf. on*

*Spoken Language Processing (ICSLP 1998)* (Sydney, Australia, 1998), vol. 2, pp. 755–758.

[209] Vergin, R., and O'Shaughnessy, D. A double gaussian mixture modeling approach to speaker recognition. In *Proc. 5th European Conf. on Speech Communication and Technology (Eurospeech 1997)* (Rhodes,Greece, 1997), pp. 2287–2290.

[210] Vogt, R., and Sridharan, S. Frame-weighted bayes factor scoring for speaker verification. In *Proc. 10th Australian Int. Conf. on Speech Science & Technology* (Sydney, Australia, 2004), pp. 404–409.

[211] Vuuren, S. Comparison of text-independent speaker recognition methods on telephone speech with acoustic mismatch. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1996)* (Philadelphia, Pennsylvania, USA, 1996), pp. 1788–1791.

[212] Vuuren, S., and Hermansky, H. On the importance of components of the modulation spectrum for speaker verification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)* (Sydney, Australia, 1998), pp. 3205–3208.

[213] Wan, V., and Renals, S. Speaker verification using sequence discriminant support vector machines. *IEEE Trans. on Speech and Audio Processing 13*, 2 (2005), 203–210.

[214] Wang, Z.-H., Wu, C., and Lubensky, D. New distance measures for text-independent speaker identification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2000)* (Beijing, China, 2000), vol. 2, pp. 811–814.

[215] Weber, F., Manganaro, L., Peskin, B., and Shriberg, E. Using prosodic and lexical information for speaker identification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2002)* (Orlando, Florida, USA, 2002), pp. 141–144.

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

[217] Wong, E., and Sridharan, S. Comparison of linear prediction cepstrum coefficients and mel-frequency cepstrum coefficients for language identification. In *Proc. of 2001 Int. Symposium on Intelligent Multimedia, Video and Speech Processing* (Hong Kong, 2001), pp. 95 – 98.

[218] Xiang, B. Text-independent speaker verification with dynamic trajectory model. *IEEE Signal Processing Letters 10* (2003), 141–143.

[219] Xiang, B., and Berger, T. Efficient text-independent speaker verification with structural gaussian mixture models and neural network. *IEEE Trans. on Speech and Audio Processing 11* (September 2003), 447–456.

[220] Xu, L., Oglesby, J., and Mason, J. The optimization of percuptually-based features for speaker identification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1989)* (Glasgow, Scotland, 1989), pp. 520–523.

[221] Xu, L., and Suen, C. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. on Systems, Man and Cybernetics 22*, 3 (May/June 1992), 418–435.

[222] Yegnanarayana, B., and Kishore, S. AANN: an alternative to GMM for pattern recognition. *Neural Networks 15* (2002), 459–469.

[223] Yoshida, K., Takagi, K., and Ozeki, K. Speaker identification using subband HMMs. In *Proc. 6th European Conf. on Speech Communication and Technology (Eurospeech 1999)* (Budapest, Hungary, 1999), pp. 1019–1022.

[224] Yuo, K.-H., and Wang, H.-C. Joint estimation of feature transformation parameters and Gaussian mixture model for speaker identification. *Speech Communication 28* (1999), 227–241.

[225] Zilca, R. Text-independent speaker verification using utterance level scoring and covariance modeling. *IEEE Trans. on Speech and Audio Processing 10*, 6 (2002), 363–370.

[226] Zilca, R., and Bistritz, Y. Speaker identification using LSP codebook models and linear discriminant functions. In *Proc. 6th European Conf. on Speech Communication and Technology (Eurospeech 1999)* (Budapest, Hungary, 1999), pp. 799–802.

[227] Zilca, R., Navratil, J., and Ramaswamy, G. Depitch and the role of fundamental frequency in speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)* (Hong Kong, 2003), vol. 2, pp. 81–84.

1

## Publication P1

T. Kinnunen, T. Kilpeläinen, P. Fränti, Comparison of Clustering Algorithms in Speaker Identification, *Proceedings of the IASTED International Conference on Signal Processing and Communications* (SPC 2000), pp. 222-227, Marbella, Spain, September 19-22, 2000.

**Proceedings of the IASTED International Conference**
**SIGNAL PROCESSING AND COMMUNICATIONS**
**September 19-22, 2000, Marbella, Spain**

# COMPARISON OF CLUSTERING ALGORITHMS IN SPEAKER IDENTIFICATION

*TOMI KINNUNEN, TEEMU KILPELÄINEN and PASI FRÄNTI*
*{tkinnu, tkilpela, franti}@cs.joensuu.fi*
*Department of Computer Science, University of Joensuu,*
*P.O.Box 111, 80101 Joensuu, FINLAND.*

## ABSTRACT

In speaker identification, we match a given (unkown) speaker to the set of known speakers in a database. The database is constructed from the speech samples of each known speaker. Feature vectors are extracted from the samples by short-term spectral analysis, and processed further by vector quantization for locating the clusters in the feature space. We study the role of the vector quantization in the speaker identification system. We compare the performance of different clustering algorithms, and the influence of the codebook size. We want to find out, which method provides the best clustering result, and whether the difference in quality contribute to improvement in recognition accuracy of the system.

Keywords: *Speech processing, speaker identification, vector quantization, clustering.*

## 1 INTRODUCTION

*Speaker recognition* is a generic term used for two related problems: speaker *identification* and *verification* [9]. In the identification task the goal is to recognize the unknown speaker from a set of *N* known speakers. In verification, an identity claim (e.g., a username) is given to the recognizer and the goal is to accept or reject the given identity claim. In this work we concentrate on the identification task.

The input of a speaker identification system is a sampled speech data, and the output is the index of the identified speaker. There are three important components in a speaker recognition system: the feature extraction component, the speaker models and the matching algorithm. Feature extractor derives a set of speaker-specific vectors from the input signal. Speaker model is then generated from these vectors for each speaker. The matching procedure performs the comparison of the speaker models. It is expected that the feature extraction is the most critical component of the system but it is also much more difficult part to be designed than the matching procedure.

In this work, we study the role of the *vector quantization* in a VQ-based speaker identification system [13]. We aim at solving this subproblem and give an answer to the question of which clustering algorithm we should use, and how large codebooks should be used. If we manage to do this, then we could fix this part of the algorithm and concentrate on more important subproblems of the system in the future.

We study the performance of several clustering algorithms, including three well known methods: *LBG*, *PNN*, and *self-organizing maps* (*SOM*), and few newer methods such as *iterative splitting* and *randomized local search* (*RLS*). We want to find out, which methods provide the best clustering results, and whether the difference in quality contributes to an improvement in the recognition accuracy of the identification system.

## 2 SPEAKER IDENTIFICATION SYSTEM

The structure of a VQ-based speaker identification system is illustrated in Fig. 1. There are two phases in the speaker identification: *training* and *recognition*. In the training phase, a mathematical model (VQ codebook in our case) is constructed for each speaker from their speech samples and the models are stored in the database. In recognition phase, the speech data of an unknown speakers is analyzed and the best matching model is searched from the database.

The analysis of the speech signals is based on short-term spectral analysis. The speech signal is decomposed into short fixed-length speech frames, which form the *feature vectors*. The feature extraction process is described more detailed in the Section 3.

The extracted feature vectors are processed further by vector quantization for locating the clusters in the feature space and for reducing the amount of data. The input of vector quantization is the set of feature vectors $X$ and the output is a *codebook* $C$ that consists of the cluster centroids, denoted as *code vectors*. The codebook represents the speaker model by approximating the distribution of the feature vectors in the feature space.

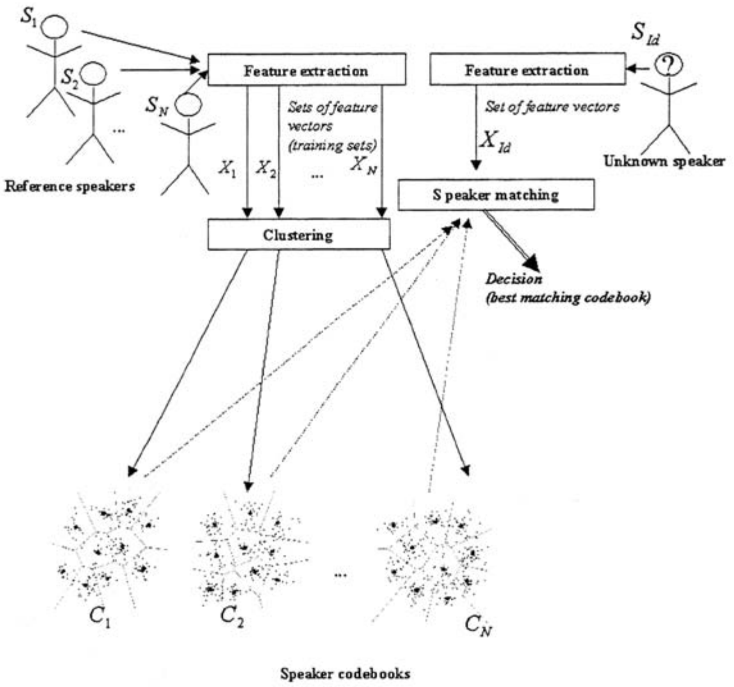


*Fig. 1: Structure of the VQ-based speaker identification system.*

The identification procedure is formulated as follows:

```
1. Compute the set of feature vectors $X = \{ \boldsymbol{x}_i \}$
2. FOR EACH speaker model $C_i$ DO
     Compute the distortion $D_i = d(X, C_i)$ between $X$ and $C_i$.
3. Identify the index of the unknown speaker $Id$ as the one with the smallest distortion, i.e.
     $Id = \operatorname*{argmin}_{i=1,\dots,N}\{D_i\}$.        (2.1)
```

The *distortion measure d* in the second step approximates the *dissimilarity* between the codebook $C_i = \{\boldsymbol{c}_{i1}, \boldsymbol{c}_{i2}, \dots, \boldsymbol{c}_{iK}\}$ and the vector set $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_L\}$. We use the most intuitive distortion measure; map each vector in $X$ to the nearest code vector in $C_i$ and compute the average of these distances:

$$d(X, C_i) = \frac{1}{L}\sum_{j=1}^{L} \min_{k=1}^{K} d_E(\boldsymbol{x}_j, \boldsymbol{c}_{ik}), \qquad (2.2)$$

where $d_E$ is the Euclidean metric:

$$d_E(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{\dim} (x_i - y_i)^2} \qquad (2.3)$$

The distortion measure (2.2), known as the mean square error (*MSE*), gives also a measure for the quality of the codebook constructed from the training set *X*.

Note that in training phase we generate codebooks for the speakers, but in the recognition phase we perform a direct comparison between the set of feature vectors and the codebooks of the known speakers. This arises the question whether we need the codebooks at the first place.

There are two good reasons for this: memory and time requirements. Computational load of the identification process becomes too high if we do not reduce the amount of data. It is very important to remove this kind of bottlenecks from a real-time speaker identification system. Memory consumption could also be a restricting factor in case of very large databases.

We assume that the feature vectors discriminate well the different acoustical units in the speech signal; similar phonemes (vectors) are located near to each other in the feature space while different phonemes are far away from each other. When we perform the clustering of the feature vectors, we obtain efficient mean values of these different short-term acoustical units. The codebooks of different speakers may have some vectors very close to each other, but it is expected that there are enough dissimilar vectors so that the matching process can differentiate between codebooks of different speakers.

## 3 FEATURE EXTRACTION

Next, we describe the procedure for computing the feature vectors from a given speech signal $s(n)$. The most commonly used features in speaker recognition systems are the features derived from the *cepstrum* [1]. Furui [8] was the first who applied cepstral analysis in speaker recognition.

### Pre-emphasis

The speech is processed by a high-emphasis filter before input to the cepstrum analysis. This is due to the well-known fact that the higher frequencies contain more speaker-dependent information than the lower frequencies. We use a high pass filter whose transfer function is

$$H(z) = 1 - az^{-1}. \qquad (3.1)$$

### Framing

The analysis of a discrete-time speech signal is based on *short-term* spectral analyses. This means that the signal is first divided into fixed-length short *frames*, e.g. 20 milliseconds. Adjacent frames usually overlap, e.g. by 10 milliseconds. After framing, these short-length "sub-signals" are considered as independent signals. For each frame, a fixed-length feature vector is computed, which describes the acoustic behavior of that particular frame.

Before frequency analysis, we apply a *window function* to the frames. The most simple windowing function is the *rectangular window*, i.e. "no window at all". However, usually smoother functions are used, and the most common in speech processing is the *Hamming window*. Smoother functions are better than rectangular window because the latter has abrupt discontinuities in its endpoints, which is undesirable for the frequency analysis [2].

### Speech production modelling

Speech production can be well modeled by the *source-filter model* introduced by Fant [4]. According to the model, speech waveform is a result of two independent components: the *source* signal produced by vocal folds and the vocal tract *filter* which emphasizes certain frequencies of the source signal according to how it is configured. To be more precise, let us denote excitation source sequence by $e(n)$ and vocal tract filter signal as $v(n)$. The resulting speech waveform is simply a convolution

$$s(n) = e(n) * v(n). \qquad (3.2)$$

In frequency domain this becomes to

$$S(\omega) = E(\omega)V(\omega). \qquad (3.3)$$

### The cepstrum

Fundamental idea of the cepstrum computation in speaker recognition is to discard the source characteristics because they contain much less information about the speaker identity than the vocal tract characteristics. In practice, the exact extraction of these two nonlinearly mixed signals $e(n)$ and $v(n)$ is impossible, but the cepstrum gives a good approximation for the "slow" spectral variations, i.e. the *envelope structure* of the signal, which characterizes the behavior of the vocal tract. Basically cepstrum computation is a deconvolution operator, which decomposes the signal into its source and filter characteristics. For the details about the way the cepstrum is computed, see e.g. [2].

The result of the deconvolution is a sequence of *cepstral coefficients* $\{c_0, c_1, \ldots, c_{M-1}\}$, where $M$ is the desired number of coefficients. Coefficient $c_0$ corresponds to the total energy of the frame and thus contains no speaker information. Usually $c_0$ is discarded or used for normalization. In the cepstral domain, we use term *liftering* to point out that we want to "lifter" out those coefficients that describe fast spectral variations, i.e. the harmonic structure. In cepstral vector, lower coefficients describe the envelope structure and higher coefficients the harmonic structure [2].

## 3 VECTOR QUANTIZATION OF THE FEATURE VECTORS

There are two important design questions in vector quantization: the method for generating the codebook, and the size of the codebook. Next, we study known clustering algorithms for codebook generation. The question about the codebook size is issued in Section 4.

The clustering problem is defined as follows. Given a set of feature vectors $X = \{ \boldsymbol{x}_i \mid i = 1,\ldots,L\}$ , partition the data set into $K << L$ clusters such that similar vectors are grouped together and vectors with different features belong to different groups. The codebook $C = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K\}$ can then be constructed from the cluster representatives, which are the vector averages of each cluster.

We consider the following clustering algorithms:

- *Random*: Random codebook,

- *GLA*: Generalized Lloyd algorithm [11],
- *SOM*: Self-organizing maps [12],
- *PNN*: Pairwise nearest neighbor [3],
- *Split*: Iterative splitting technique [5],
- *RLS*: Randomized local search [7]

**Random**: A random codebook can be generated by selecting $K$ random feature vectors. It serves as a point of comparison.

**GLA**: Generalized Lloyd algorithm (also known as *LBG*) starts with an initial codebook, which is iteratively improved until a local minimum is reached. In the first step, each feature vector is mapped to the nearest code vector in the current codebook. In the second step, the code vectors are recalculated as the centroids of the new partitions. The algorithm is iterated as long as improvement is achieved.

**SOM**: Self-organizing maps is a neural network approach to the clustering. The neurons in the network are connected with a 1-D or 2-D structure, and they correspond to the codevectors. The feature vectors are feed to the network by finding the nearest codevector for each input vector. The best matched codevector and its neighboring vectors (according to the network structure) are updated by moving it towards the input vector. After processing the training set by a predefined number of times, the neighborhood size is shrunk and the entire process is repeated until the neighborhood shrinks to zero.

**PNN**: Pairwise nearest neighbor generates the codebook hierarchically. It starts by initializing each training vector as a separate code vector. Two code vectors are merged at each step of the algorithm and the process is repeated until the desired size of the codebook is obtained. The code vectors to be merged are always the ones whose merge increase the distortion least. We use the fast exact PNN method introduced in [6].

**Split**: An opposite, top-down approach starts with a single cluster including all the feature vectors. New clusters are then created one at a time by dividing existing clusters. The splitting process is repeated until the desired number of clusters is reached. The divisive approach usually requires much less computation than the PNN. The best known approach for the splitting is to use *principal component analysis* (PCA). This method gives comparable results to that of the PNN with much faster algorithm.

**RLS**: Randomized local search algorithm starts with a random codebook, which is then improved by a predefined number of iterations. At each step, a new candidate solution is generated using the following operations. The clustering structure of the current solution is first modified using so-called *random swap* technique, in which a randomly chosen code vector is replaced by another randomly chosen input vector. The partition of the new solution is then adjusted in respect to the modified codebook. Two iterations of the GLA are then applied to fine-tune the trial solution. The candidate is evaluated and accepted if it improves the previous solution. The algorithm is iterated for a fixed number of iterations.

## 4 EXPERIMENTAL RESULTS

We collected a speaker database of 25 speakers (14 males + 11 females). Speech was recorded in a laboratory environment with a PC computer. For each speaker we recorded two utterances of Finnish speech: one for training and the other for recognition. Every speaker read the same sentences. Summary of the speech database is given in Table 1.

*Table 1: Summary of the speaker database.*

| | |
|---|---|
| # Speakers | 25 (14 M + 11 F) |
| Avg. duration of training utterance | 66.5 s |
| Avg. duration of recognition utterance | 17.7 s |
| Sampling & quantization | 11.025 kHz, 16 bits. |

Before analysis, the speech files were anti-alias filtered and downsampled to 8.0 kHz. After that, silent parts were removed using short-term energy calculations. The feature extraction itself was performed as follows: remove DC offset, high-emphasis filtering with $H(z) = 1 - 0.95z^{-1}$, and, finally, perform short-term mel-cepstrum analysis with a 30 ms Hamming-window, with 10 ms shift. The number of mel-cepstral coefficients (dimension of feature vectors) were selected as 12. As usually, coefficient $c_0$ was discarded.

We evaluated the performance of the five different clustering algorithms of Section 3. As the measure of quality for a given VQ codebook, we use the mean squared error between the training set and the resulting codebook $C_i$. The resulting MSE-values are shown in Fig. 2 and 3 with two different sizes of codebook ($K$=64 and $K$=256).

The results show that there are only a small difference between the best clustering algorithms. Even the standard GLA method gave MSE-values close to that of the best method, RLS. The corresponding identification rates are shown in Fig. 4 and 5. The choice of the clustering method have only a marginal effect on the identification rate.

The effect of the codebook size is illustrated in Fig. 6 and 7 for the best method (RLS) and for the random codebook. The identification rates clearly increases with respect to the codebook size. If it is set to 128 or higher, even with the random codebook the method is capable of identifying 96% of the speakers, which corresponds to a single miss-classification. With the best clustering methods (RLS, SPLIT), the identification rate does not improve anymore for codebooks of sizes > 64.

Finally, the running times for generating the codebooks are shown in Fig. 8. If the database can be constructed off-line, the running times are hardly significant. The RLS method takes slightly longer time than the rest of the methods because it was tuned for quality and not for speed. If the running time was critical, then the SPLIT method would be a good choice.

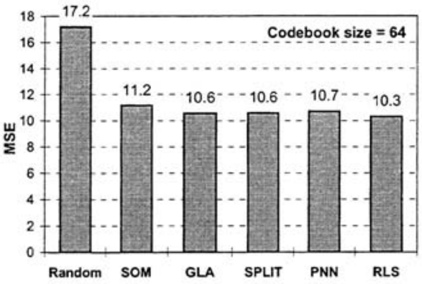


*Fig. 2: Quality of the codebook (scaled MSE-values) using different clustering algorithms. K=128.*

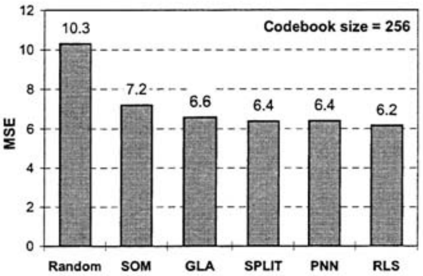


*Fig. 3: Quality of the codebook (scaled MSE-values) using different clustering algorithms. K=256.*

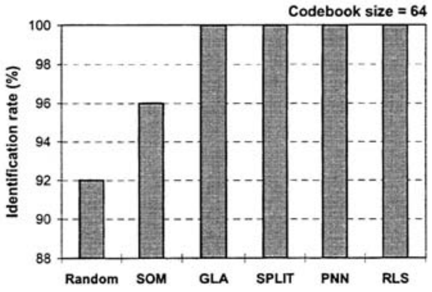


*Fig. 4: Identification accuracy of the algorithms. K=64.*

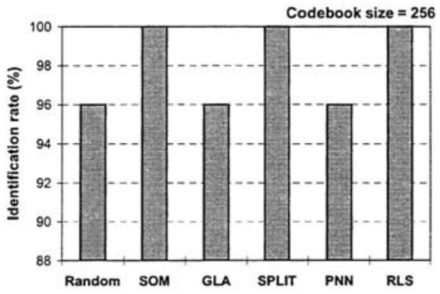


*Fig. 5: Identification accuracy of the algorithms. K=256.*

## 5 CONCLUSIONS

We evaluated the performance of five different clustering algorithms for VQ-based speaker identification. We noticed that the MSE-values of the codebooks produced by the algorithms were only marginally different, and the corresponding recognition rates were rather similar.

The easiest way for improving the identification accuracy was to increase the codebook size high enough. No side-effect was observed due to the increase, except the increase in the running. However, codebooks of size greater than 64 did not have any further impact as the identification rate already reached 100%.

We conclude that the fastest algorithm (SPLIT) should be used if the speaker database is very large and running time important. Otherwise, we recommend to use the best algorithm (RLS) because it is simpler to implement and, after all, it gave the best codebooks even though the difference was only marginal for our database.

It is noted that the speaker database was relatively small, the speech samples were quite long, and they were generated in laboratory conditions. Future experiments must therefore be made in more demanding environments in order to obtain more conclusive results.

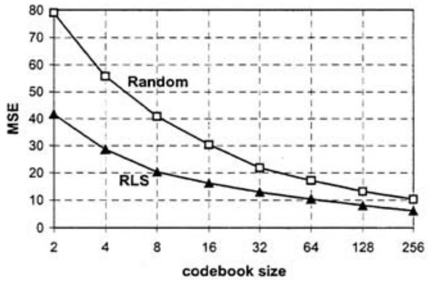


*Fig. 6: Quality of the codebook as a function of the codebook size.*

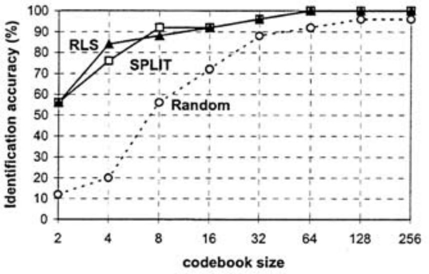


*Fig. 7: Identification accuracy of the algorithms as a function of the codebook size.*

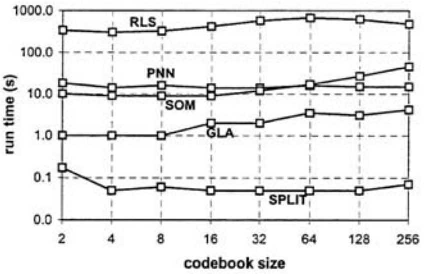


*Fig. 8: Run times of the clustering algorithms.*

## REFERENCES

[1] Bogert B.P., Healy M.J.R., Tukey J.W.: The quefrency alanysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking, *Proc. Symposium Time Series Analysis*, John Wiley and Sons, NY, 209-243, 1963.

[2] Deller Jr. J.R., Proakis J.G., Hansen J.H.L.: *Discrete-time Processing of Speech Signals*. (New York: Macmillan Publishing Company, 2000).

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

[4] Fant G.: *Acoustic Theory of Speech Production*. (Mouton: The Hague, 1960).

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

[6] Fränti P., Kaukoranta T., Shen D.-F., Chang K.-S.: Fast and memory efficient implementation of the exact PNN, *IEEE Trans. on Image Processing*, 9 (5): May 2000.

[7] Fränti P., Kivijärvi J.: Randomized local search algorithm for the clustering problem, *Pattern Analysis and Applications*. (to appear)

[8] Furui S.: Cepstral analysis technique for automatic speaker verification. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 29(2): 254-272, 1981.

[9] Furui S.: Recent advances in speaker recognition. *Pattern Recognition Letters*, 18: 859-872, 1997.

[10] Gersho A., Gray R.M.: *Vector Quantization and Signal Compression*. (Dordrecht: Kluwer Academic Publishers, 1992).

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

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

[13] Soong F.K., Rosenberg A.E., Juang B-H., Rabiner L.R.: A vector quantization approach to speaker recognition, *AT&T Technical Journal*, 66: 14-26, 1987.

## Publication P2

T. Kinnunen, Designing a Speaker-Discriminative Adaptive Filter Bank for Speaker Recognition, *Proceedings of the 7th International Conference on Spoken Language Processing* (ICSLP 2002), pp. 2325-2328, Denver, Colorado, USA, September 16-20, 2002.

# DESIGNING A SPEAKER-DISCRIMINATIVE ADAPTIVE FILTER BANK FOR SPEAKER RECOGNITION

*Tomi Kinnunen*

Department of Computer Science
University of Joensuu, Finland
tkinnu@cs.joensuu.fi

## ABSTRACT

A new filter bank approach for speaker recognition front-end is proposed. The conventional mel-scaled filter bank is replaced with a speaker-discriminative filter bank. Filter bank is selected from a library in adaptive basis, based on the broad phoneme class of the input frame. Each phoneme class is associated with its own filter bank. Each filter bank is designed in a way that emphasizes discriminative subbands that are characteristic for that phoneme. Experiments on TIMIT corpus show that the proposed method outperforms traditional MFCC features.

## 1. INTRODUCTION

Several studies have indicated that different phonemes have unequal discrimination powers between speakers [3, 10, 12]. That is, the inter-speaker variation of certain phonemes are different from other phonemes. For instance, in [3] vowels and nasals were found to be most discriminating phoneme groups.

Discrimination analysis of speech sounds can be, however, carried out from a non-phonetic viewpoint also. In several engineering-oriented studies, evidence of the different discrimination properties of certain frequency bands have been discovered [6, 14, 15]. For example, in [6] the spectra of speech were divided into upper and lower frequency regions with the cutoff frequency being the varied parameter. It was found, among other observations, that regions 0-4 kHz and 4-10 kHz are equally important for speaker recognition.

In [11] a more detailed analysis of frequency band discrimination was performed. Spectral analysis was carried out with a filter bank with triangular overlapping filters. Discrimination powers of these subbands were then evaluated with three different criteria, the *F-ratio* [1] being one criterion. A non-linear frequency warping based on the discrimination values was then proposed: more filters with narrower bandwidths were placed in the discriminative regions, while less filters with broader bandwidth were placed in the non-discriminative regions. The proposed system outperformed conventional mel-frequency warped filter bank.

Although the phonetic studies indicate differences in phoneme-level discrimination powers, no segmentation is usually done prior to discrimination analysis with the engineering-oriented approaches. The problem is, however, that when all different phoneme classes' data are pooled together, some discriminative frequency bands that are characteristic for a certain phoneme may be averaged away. The frequency of occurence of phonemes reflects directly to the discrimination values. As a consequence, if the corpus used in experiments contains a discriminating phoneme which is infrequent, its significance decreases.

In this work, we introduce an approach which falls in the middle ground between the ”phonetical”- and ”engineering”-oriented discrimination analyses.

Idea of the proposed front-end is illustrated in Fig. 1. Each speech frame is processed with a filter bank which is selected from a library of filter banks according to the phoneme class of the frame. Thus, each phoneme class is filtered in a customed way instead of a global filter bank as in [11].

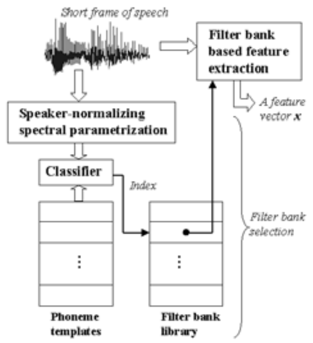


*Figure 1:* The idea of adaptive filter bank

The basic idea of the proposed method is simple. However, there arises immediately the following design issues:

- Which parametrization to choose in the determination of the phoneme class,
- How to generate and represent the phoneme templates,
- What is “optimal number” of the phoneme templates,
- How to compute discriminative values for subbands in phoneme-depended filter banks,
- How to exploit the filter bank in the feature extraction.

These are the substantial topics of this paper.

## 2. THE PHONEME CLASSIFIER

### 2.1 Representation of the phoneme templates

In order to be of general use, the phoneme template model must be speaker (or even language) independent. That is, the same model for all speakers can be used to find the phoneme classes. We denote this model as the *universal phoneme model* (UPM). Due to the requirement of speaker-independence, the UPM must be designed such that it accounts the differences between speakers and other sources of variability.

Note in Fig. 1 the block labeled “speaker-normalizing spectral parametrization”. *Speaker normalization* means that we wish to de-emphasize speaker-depended features. We use the following parametrization which is general in speech recognition [12]:

- High emphasis with $H(z)=1 - 0.97z^{-1}$,
- Frame length 30 ms, Hamming-windowed and shifted by 20 ms (33 % overlap),
- 12 lowest mel-frequency coefficients (MFCC), 20 triangular filters in the bank, coefficient $c_0$ discarded,
- Cepstral coefficients weighted by *raised sine* function.

### 2.2 Generation of the templates

We use clustering techniques [4, 5, 7] for generating the UPM from MFCC vectors. We use 100 speakers from the TIMIT corpus [9] as the training set. For each speaker, we take five speech files in the training data. These are downsampled to 8 kHz and processed with the parametrization given above. Final training set consists of approximately 100,000 vectors.

From the training set, a codebook is generated by the RLS algorithm [4]. The following different codebook sizes $K$ are used: $K$=4, 8, 16, 32, 64.

The worth noticing point here is that we use *unsupervised learning* in the UPM generation; i.e. we do not use any explicit segmentation of speech or labeling of phonemes, since we are not interested in decoding the linguistic message of the input signal.

### 2.3 Classification of a frame

When applying the UPM in the phoneme classification, the class is simply determined by the nearest neighbor rule. Frame is first parametrized in the way described in Section 2.1, resulting in a single MFCC vector $\boldsymbol{x}$. The label of the phonetic class is then given by

$$i^* = \underset{\boldsymbol{p}\in UPM}{\arg\min}\ d(\boldsymbol{x}, \boldsymbol{p}), \tag{1}$$

where $d$ is the squared Euclidean distortion measure. The index $i^*$ is sent to the filter bank library to select the associated filter bank (see Fig. 1).

## 3. DESIGNING THE LIBRARY OF DISCRIMINATIVE FILTER BANKS

### 3.1 Subband processing

We want to assign discriminative values for each subband per each phoneme class present in the UPM. To get started, we must specify what we mean here by a subband.

As in general speech processing front-ends [2] we use overlapping triangular filters to cover the entire frequency range (see Fig. 2). Filters are uniformly spaced and overlap by 50%. In this phase we wish to avoid using any nonlinear frequency warping, such as mel or Bark-scales, in order to be sure that each subband has equal contribution in the discrimination analysis.

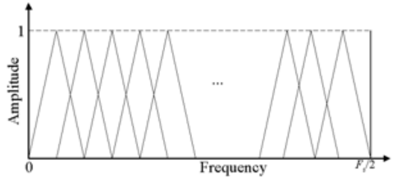


*Figure 2:* Uniform triangular filter bank

For a Hamming-windowed frame $s$, an $N$-point FFT $S[k]$ is first computed. The magnitude spectrum in dB-scale is then computed as $10\log_{10}|S[k]|$. The dB magnitude spectrum is weighted by the triangular filterbank of $M$ filters, thus implying $M$ subband energies $E_j$, $j$=1,...,$M$. These are collected in a $M$-dimensional vector $\boldsymbol{E} = (E_1,...,E_M)^T$. Hereafter, by “$j$th subband” we simply refer to $E_j$. We fix the number of filters to $M$=40. Thus, for the speech with sampling rate $F_s$ = 8 kHz, the bandwidth of each filter is 100 Hz.

### 3.2 Assigning the discrimination values to subbands

We use the *F ratio* [1] for assigning a discrimination value for the $j$th subband of $i$th phoneme:

$$F_{i,j} = \frac{\begin{array}{l}\text{Variance of speaker means of subband } j\\ \text{of phoneme } i\end{array}}{\begin{array}{l}\text{Average intraspeaker variance of subband } j\\ \text{of phoneme } i\end{array}}. \tag{2}$$

If the inter-speaker variability is large while inter-speaker variability being low, F ratio is large.

Since we wish to assign the F ratios for each phoneme-subband pair, we must first segment the training data into phonetic classes using the UPM described in Section 2. Then, for each “phonetic class pool” ($i$) we can compute the discrimination values for subbands ($j$) using F-ratio (2). The segmentation of the data into the pools is outlined in the following pseudocode.

```
Inputs:
•   Frames of speech {$s_{k,t}$}, where $k$ is the speaker number
    and $t$ is the frame number,
•   The UPM of $K$ phoneme templates, UPM={$p_1$,...,$p_K$},
•   The number of subbands, $M$
Outputs:
•   $K$ pools of subband vectors

Procedure:
FOR EACH speaker $k$ DO
    •   Get next frame $s = s_{k,t}$ ;
    •   Compute the MFCC vector $x$ for $s$ (as described in
        Section 2.1) ;
    •   Compute the subband vector $E$ for $s$ (as described in
        Section 3.1) ;
    •   Find nearest phoneme $i^*$ for $x$ from UPM by (1) ;
    •   Add the vector $E$ to the pool of phoneme $i^*$ ;
ENDFOR
```

*Figure 3:* Segmentation of the training data for discrimination analysis

To put it in words, each frame is classified by its phonetic content, the UPM code vectors {$\boldsymbol{p}_i$} serving as the phoneme class representatives. The subband vector of the frame is assigned to the best matching phoneme template.

After the pooling, F ratios of each pool are computed. Indeed, different phonemes have different F curves as seen in Fig. 4, where we have used an UPM of size $K$=8.

A few prelimary observations can be made from the F curves. Firstly, nearly all phonemes have a peak in discrimination values approximately in the subbands 2-4, which correspond to frequency range 50-250 Hz. Secondly, one may see some resemblance of the F ratio shapes with the envelopes of smoothed LPC spectra, thus indicating the importance of formant structure and overall spectral envelope in speaker recognition.

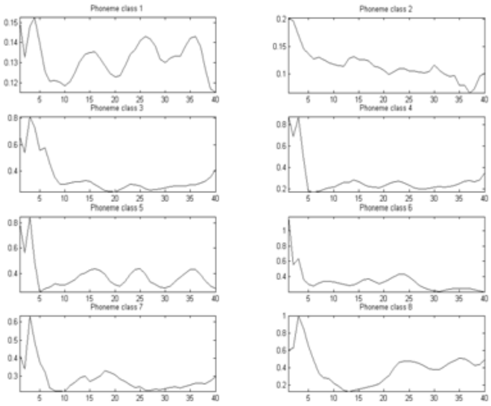


*Figure 3: F ratios of subbands for different phonemes (UPM size K=8)*

We run also an experiment in which, instead of pre-smoothing the spectra with a filter bank, we used all the magnitude values from FFT analysis as such and computed the F ratios. We found soon out that the F curves obtained in this way were very noisy; further, the computational load for this method is huge compared pre-smoothing using the filter bank. For these reasons, we end up using the filter bank.

### 3.3 Utilization of the filter bank in feature extraction

Once the F ratios are computed for each phoneme-subband pair, it is straightforward to utilize them in the feature extraction. The broad phoneme class $i^*$ is first found by (1). This is followed by the subband analysis as described in Section 3.1, leading to vector $\boldsymbol{E}$. The components of $\boldsymbol{E}$ are then weighted by the relative F ratio of the subband:

$$E_j^{'} = E_j \frac{F_{i^*,j}}{\sum_{m=1}^{M} F_{i^*,m}} \tag{3}$$

An example of subband weighting is shown in Fig. 4. The figures from top to down show the magnitude spectrum, filtered magnitude spectrum, relative weights for each subband, and the weighted filter outputs.

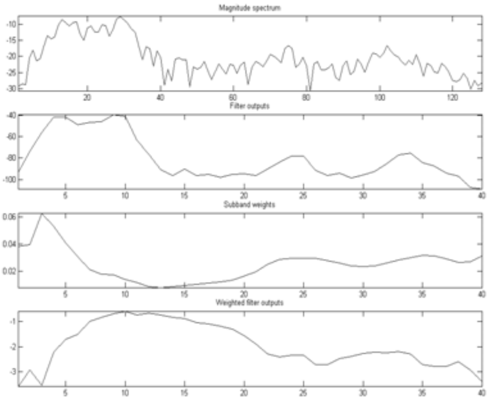


*Figure 4 : An Example of subband weighting*

Weighted filter outputs are then fed to discrete cosine transform (DCT) for decorrelating the features. Only the lowest $L$ coefficients of DCT are retained, excluding the 0th coefficient.

In summary, the processing steps are same to that of the conventional MFCC analysis, except for that the mel-spaced filter bank is replaced with the discriminative filter bank. Hereafter, we abbreviate the features obtained in this way by *ADFB-cep* (standing for *Adaptive Discriminative Filter Bank Cepstrum*).

## 4. RESULTS

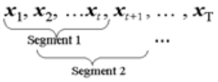
The overall process of evaluating the proposed approach consists of the following steps:

- Create UPM as described in Section 2 (Using speaker set *SET 1*),
- Use independent data for finding the F ratios as described in Sections 3.1 and 3.2 (*SET 2*),
- Using third speaker set (*SET 3*), compute the *ADFB-cep* features as described in Section 3.3. We choose the number of filters to *M*=40 and number of coefficients to *L*=20. *SET 3* is further divided into training and evaluation sets.

All the three sets are disjoint. In this way we ensure that results will be not biased by the tuning to the training set; that is, we wish to have a general front-end without the need to construct the UPM and/or the filter design data each time the database is switched.

Each of the three sets consist of 100 speakers. We use VQ codebooks as speaker models [8, 13], each model having 64 code vectors and created using the RLS clustering method [4]. Average duration of the training speech data is 15 seconds.

Each test set $X = \{\boldsymbol{x}_1,...,\boldsymbol{x}_T\}$ is divided into overlapping segments as shown in the following:


$\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \cdots, \boldsymbol{x}_T$
Segment 1
...
Segment 2

Average duration of the test segment is about 1 second. Each of the segments is classified using the speaker models by the minimum average quantization error rule [13]. We use the percentage of correctly classified segments as the evaluation criterion.. The results for different UPM sizes are shown in Table 1.

*Table 1: Evaluation results*

| UPM size | ID rate (%) |
|---|---|
| 4 | 69.37 |
| 8 | 74.85 |
| 16 | 67.071 |
| 32 | 58.49 |
| 64 | 55.73 |

For comparison, conventional 20 mel-cepstral coefficients (MFCC) were computed with same frame rate and equal parameters: number of mel-filters was 40 and the number of coefficients was 20. The identification rate using MFCC was 61.96.

Based on these experiments, we make several observations. Firstly, the optimum size of UPM is *K*=8. When the UPM size is increased, results get poor. Also, the differences in performance are quite large, which suggests that we should use a linear scale instead of exponential when finding the "optimum size".

Secondly, and more interestingly, the proposed method outperforms MFCC parameters, even if the UPM size is not "optimal". The overall identification rates are quite poor in all cases, due to the very short test segment length.

## 5. CONCLUSIONS

A new feature set based on discriminative weighting of the characteristic subbands for each "phoneme class" was proposed and evaluated experimentally. Prelimary results are very encouraging since they outperform the popular MFCC features. In future experiments, we plan to include cross-language evaluation, careful optimization of the UPM, and other discrimination criteria in addition to F ratio.

## 6. REFERENCES

[1] Campbell, J., "Speaker Recognition: A Tutorial," *Proc. IEEE*, **85**(9): 1437-1462, 1997.

[2] Deller, J.R. Jr., Hansen, J.H.L. and Proakis, J.G., *Discrete-time Processing of Speech Signals*. Macmillan Publishing Company, New York, 2000.

[3] Eatock, J.P. and Mason, J.S., "A Quantitative Assessment of the Relative Speaker Discriminating Properties of Phonemes", *Proc. ICASSP'94*: 133-136, Adelaide, 1994.

[4] Fränti, P. and Kivijärvi, J., "Randomized Local Search Algorithm for the Clustering Problem", *Pattern Analysis and Applications*, **3**(4): 358-369, 2000.

[5] Gersho, A. and Gray, R.M., *Vector Quantization and Signal Compression*, Kluwer Acad. Pub., 1992.

[6] Hayakawa, S. and Itakura, F.: "Text-Dependent Speaker Recognition Using the Information in the Higher Frequency Band", *Proc. ICASSP'94*: 137-140, Adelaide, 1994.

[7] Jain A.K, Murty M.N. and Flynn P.J., "Data Clustering: A Review", *ACM Computing Surveys* **31**(3): 264-323, 1999.

[8] Kinnunen, T., Kärkkäinen, I., "Class-Discriminative Weighted Distortion Measure for VQ-Based Speaker Identification", accepted for publication in *Proc. Joint IAPR International Workshop on Statistical Pattern Recognition (SPR 2002)*, Windsor, August 6-9, 2002.

[9] *Linguistic Data Consortium*, http://www.ldc.upenn.edu/

[10] Nolan, F., *The Phonetic Bases of Speaker Recognition*, Cambridge CUP, 1983.

[11] Orman, Ö.D. and Arslan, L.M., "Frequency Analysis of Speaker Identification", *2001 Speaker Odyssey*, Jerusalem, 2001.

[12] Rabiner, L. and Juang, B.-H., *Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[13] Soong, F.K., Rosenberg, A.E., Juang, B.-H. and Rabiner, L.R.: "A Vector Quantization Approach to Speaker Recognition", *AT&T Technical Journal*, **66**: 14-26, 1987.

[14] v. Vuuren, S. and Hermansky, H., "On the Importance of Components of the Modulation Spectrum for Speaker Verification", *Proc. ICSLP '98*: 3205-3208, Sydney, 1998.

[15] Yoshida, K., Takagi, K. and Ozeki, K., "Speaker Identification Using Subband HMMs", *Proc. EUROSPEECH '99*: 1019 - 1022, Budapest, 1999.

3

## Publication P3

T. Kinnunen, V. Hautamäki, P. Fränti, On the Fusion of Dissimilarity-Based Classifiers for Speaker Identification, *Proceedings of the 8th European Conference on Speech Communication and Technology* (EUROSPEECH 2003), pp. 2641-2644, Geneva, Switzerland, September 1-4, 2003.

# On the Fusion of Dissimilarity-Based Classifiers for Speaker Identification

*Tomi Kinnunen, Ville Hautamäki, Pasi Fränti*

Department of Computer Science
University of Joensuu, Finland
{tkinnu,villeh,franti}@cs.joensuu.fi

## Abstract

In this work, we describe a speaker identification system that uses multiple supplementary information sources for computing a combined match score for the unknown speaker. Each speaker profile in the database consists of multiple feature vector sets that can vary in their scale, dimensionality, and the number of vectors. The evidence from a given feature set is weighted by its reliability that is set in *a priori* fashion. The confidence of the identification result is also estimated. The system is evaluated with a corpus of 110 Finnish speakers. The evaluated feature sets include mel-cepstrum, LPC-cepstrum, dynamic cepstrum, long-term averaged spectrum of /A/ vowel, and F0.

## 1. Introduction

Speaker individuality is a complex phenomenon, where different supplementary information sources contain a part of evidence of the speaker identity. The individual speaker characteristics occur both at the lexical, segmental and prosodic levels [11]. At the lexical level [15] this is reflected, for instance, in usage of certain word patterns. At the segmental level, speaker differences occur at the acoustic differences of phoneme realizations that arise from physiology and anatomy of the voice production organs. Prosodic speaker characteristics are reflected in the usage of pitch, stress and timing.

Extraction of individual characteristics is realized by measuring *acoustic parameters* or *features* from the speech signal. Commonly used features in automatic speaker recognition systems include mel-cepstrum, LPC-cepstrum [1], line spectral frequencies [10], subband processing [6], dynamic cepstral parameters [14], and prosodic parameters [15].

Spectral parameters alone, especially the cepstrum with its variants, have shown good performance in speaker recognition. However, cepstrum carries only one source of evidence. To achieve better recognition accuracy, several supplementary information sources should be used.

The idea of using multiple features in speaker recognition is not new. A well-known classifier fusion strategy is to concatenate the cepstral vectors with their delta- and delta-delta cepstra into a long feature vector [1]. Also the fundamental frequency has been used in addition with the cepstral vectors to improve recognition accuracy. In general, vector concatenation is termed as *classifier input fusion* [12].

Although classifier input fusion is simple to implement and works reasonably well, it has a few shortcomings. Firstly, the feature space formed by concatenation of different features is somewhat superficial. The higher the dimensionality of the space becomes, the less and less effect a single feature has to the overall match score. Also, fusion becomes difficult if the feature is missing (e.g. F0 for unvoiced sounds) or it should be computed with a different frame rate.

Another way of performing classifier fusion is to combine different classifiers. In *classifier output fusion*, each individual data source is modeled separately, and the outputs of the individual classifier scores are combined to give the overall match score. For instance, output fusion of the cepstral and delta-cepstral features has been performed using VQ codebooks [14] and Gaussian mixture models [12] as the individual classifiers.

Slomka & al. [12] compared input and output fusion for the mel-cepstrum and corresponding delta features. They found out that the output fusion performed consistently better. Furthermore, they demonstrated that the computational complexity for the input fusion is higher than that of the output fusion.

Classifier output fusion is, with to many respects, a flexible combination strategy. For instance, it enables the same data source to be modeled by several different classifiers. In [9], a committee of five learning vector quantization (LVQ) networks with different network structures was applied. The combination was done with majority voting rule.

The main objective of this paper is to design the fusion strategy such that evidences from diverse data sets could be combined in a coherent way. Problems arise when the data sources differ in (1) the number of features (dimensionality), (2) the number of measurements, (3) the scales. Furthermore, a model that works well for one data source might not be good to model another feature. Thus, each individual feature stream should be modelled with the most suitable model for that stream. The proposed classifier is invariant to different scales of feature sets, their dimensionality, and the number of measurements. For each feature set, an *a priori* weight is set based on the reliability of the feature set.

This work was carried out in co-operation with the Department of Phonetics at the University of Helsinki as a part of larger speaker recognition project [5]. To be reliable in, for instance, realistic forensic uses, speaker recognition should be based on many parameters instead of only spectral parameters. Forensic speech samples often suffer from different types of noises and distortions, and therefore, supplementary identity cues should be used to give a joint decision. The combination of supplementary evidences from diverse feature sets, however, is not a straightforward task. In this paper, we report the structure of the fusion system we designed for the use of this project. The experiments show that using multiple feature sets together improves recognition accuracy.

## 2. The structure of the system

The structure of the proposed classifier fusion system is shown in Fig. 1. The *profile* of each of the registered $N$ speakers $S(i),\ i = 1, \ldots, N$, consists of $M$ distinct models, $S(i) = \{S_1(i), \ldots, S_M(i)\}$. Each of the models consists of a set of feature vectors. For each model, there is a correspond-

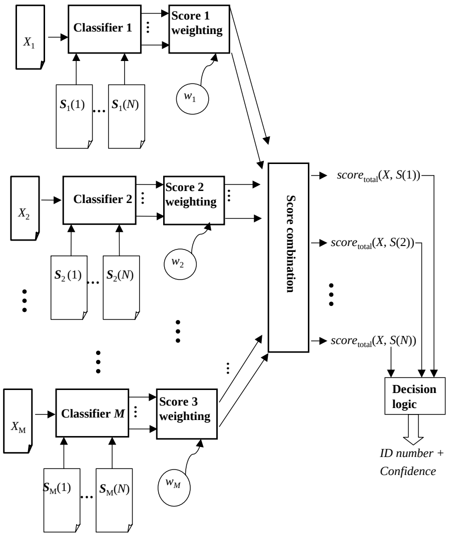


Figure 1: *Structure of the proposed system.*

ing *sub-classifier* or *expert*. Given an unknown speaker profile $X = \{X_1, \ldots, X_M\}$, each of the experts $j$ computes a match score $score(j, i)$ for each speaker $i$. The match score $score(j, i)$ indicates the degree of similarity (or dissimilarity) between point sets $X_j$ and $S_j(i)$.

The individual expert outcomes $score(j, i), j = 1, \ldots, M$ are weighted by *a priori* weights $w(j)$ that indicate the reliability of the expert. The weighted match scores from the different experts are then combined into a single match score $score_{\text{total}}(X, S(i))$ that indicates the degree of similarity (or dissimilarity) between the speakers $X$ and $S(i)$. The decision is given by returning the ID number of the most similar speaker to $X$. The confidence of the decision is also estimated based on the spread of the distribution of the match scores from different speakers.

### 2.1. Sub-classifiers

For simplicity, we will use dissimilarity-based classifiers for all feature sets. For each speaker, the individual feature sets are modeled by codebooks [10, 6, 13] generated by clustering the feature vectors of that feature set by randomized local search algorithm [3].

Dissimilarity of point sets sets $X_j$ and $S_j(i)$ is computed by the average quantization distortion:

$$D(j, i) = \frac{1}{|X_j|} \sum_{\vec{x} \in X_j} \min_{\vec{y} \in S_j(i)} \|\vec{x} - \vec{y}\|^2, \qquad (1)$$

where $|X_j|$ denotes the cardinality of $X_j$ and $\|.\|$ denotes the Euclidean norm. The match score for the sub-classifier is computed as normalized distortion:

$$score(j, i) = \frac{D(j, i)}{\sum_{k=1}^{N} D(j, k)}. \qquad (2)$$

In other words, the distortion of each speaker within the sub-classifier is normalized by the sum of the distortions from all speakers within that sub-classifier. This ensures that $0 \leq score(j, i) \leq 1$. In this way, the outputs of the individual classifiers are in the same order of magnitude regardless of the dimensionality or the number of vectors.

### 2.2. Fusion strategy

There are several options for combining the outputs from the sub-classifiers [2, 7]. Kittler & al. [7] compared several commonly used fusion criteria in the context of of probabilistic classifiers. Their theoretical and experimental results indicated that the *sum rule* is most resilient to estimation errors. Therefore, we define the combination rule as the weighted sum:

$$score_{\text{total}}(X, S(i)) = \sum_{j=1}^{M} w(j)\ score(j, i), \qquad (3)$$

where $w(j)$ is the weight for the feature set $j$. The weights are normalized such that $\sum_{j=1}^{M} w(j) = 1$, which allows the weights to be interpreted as relative importances. For instance, if there are two feature sets and we set $w(1) = 0.2$ and $w(2) = 0.8$, then the second set gets four times more weight in the fusion compared to the first one.

### 2.3. Decision and confidence estimation

The identification decision is the speaker $i^*$ which produces the smallest combined score:

$$i^* = \arg \min_{0 \leq i \leq N} score_{\text{total}}(X, S(i)). \qquad (4)$$

We also estimate the *confidence* of the decision. Intuitively, one should expect high confidence if the selected speaker is very distinctive, i.e. the scores for all other speakers are significantly higher. On the other hand, if there exists another speaker that is close to $i^*$, the decision is more uncertain. Based on this idea, we define the confidence as

$$c = 1 - \frac{score_{\min}}{score_{\min 2}}, \qquad (5)$$

where $score_{\min}$ and $score_{\min 2}$ are the scores for the nearest and second nearest speakers, respectively.

### 2.4. Determination of the weights

We consider two ways of determining the weights in Eq. (3). In both cases, we use the same database for the weight computation and matching itself. In other words, the weights are optimized for the given database. In the first approach, we apply a separability criterion for the within- and between-speaker distance scores within each feature set. The distances are computed between every speaker pair in the given database using Eq. (1). Then, the separability of the within- and between-speaker distance score distributions is computed using the Fisher's criterion [4]:

$$F = \frac{(\mu_w - \mu_b)^2}{\sigma_w^2 + \sigma_b^2}, \qquad (6)$$

where $\mu_w, \mu_b$ and $\sigma_w^2, \sigma_b^2$ are the means and variances of the two distributions, respectively. The Fisher's criterion gives a high value if the two distribution are well-separated.

In the second approach, we use exhaustive search to find the optimum weight combination. In other words, the performance

Table 1: Summary of the data sets.

| | Dimensionality | Vectors | Range |
|---|---|---|---|
| MFCC | 16 | 499 | [-102.9, 48.4] |
| △-MFCC | 16 | 499 | [-12.7,13.4] |
| △△-MFCC | 16 | 499 | [-5.5, 6.5] |
| LFCC | 20 | 1990 | [-18.1,48.4] |
| LTAS | 513 | 1 | [-25.6, 57.6] |
| F0 | 1 | 469 | [57.9, 323.0] |

Table 2: Performances of the subclassifiers.

| | Error rate | Avg. confidence |
|---|---|---|
| MFCC | 6.36 % | 0.14 |
| △-MFCC | 52.72 % | 0.05 |
| △△-MFCC | 46.36 % | 0.04 |
| LFCC | 46.36 % | 0.10 |
| LTAS | 5.45 % | 0.53 |
| F0 | 93.64 % | 0.35 |

of the system is evaluated for every weight combination, and the best weight combination is selected. For a small number of feature sets this approach can be applied.

## 3. Experiments

### 3.1. Corpus description

The test material consists of 110 native Finnish speakers from various dialect regions in Finland [5]. The recordings were done in a silent environment by a professional reporter C-cassette recorder. The data was digitized using 44.1 kHz sampling frequency with 16 bits per sample. All speakers read the same material which was divided into training and evaluation sets of length 10 seconds both.

### 3.2. Acoustic measurements

The original acoustic measurements as provided by the University of Helsinki consisted of four data sets [5]: fundamental frequency (F0), long-term averaged spectrum (LTAS) for vowel /A/ , linear frequency cepstral coefficients (LFCC) and mel-cepstral coefficients (MFCC). We furthermore added the dynamic cepstrum parameters (△-MFCC, △△-MFCC) due to their popularity in automatic speaker recognition systems.

The data sets are summarized in Table 1. From this table, we can see that input fusion would be impossible due to the diversity of the data sets. The fusion system enables using arbitrary feature sets together.

### 3.3. Sub-classifier performance

First, the performances of each feature set alone were evaluated. After some experimentation, we fixed the model sizes as follows. For MFCC, LFCC, △-MFCC and △△-MFCC the models consist of 100 code vectors. For F0, the model consists of 5 code vectors. For LTAS, the model consists of, by definition, one long vector containing 513 averaged subband outputs from different instances of /A/ vowels.

The performances of the individual data sets are summarized in Table 2 for segment length 1.8 seconds. Both the identification error rate and average confidence for the correctly classified speakers are shown.

We found out that in general increasing the model size and the test segment length improves recognition results. An exception was F0, for which the behaviour was somewhat inconsistent with respect both to the model size and to the test segment length. From the six sets, MFCC and LTAS performed best and F0 worst.

Notice that the confidences do not go in parellel with the recognition rates. For instance, F0 gives poor identification result but the confidence for the correctly classifier speakers is higher than that of MFCC, for instance.

### 3.4. Fusion of data sources

Since the fundamental idea of the fusion is that the classifiers could complement each others results, the fusion of correlated classifiers is not reasonable. In other words, if two classifiers misclassify the same speakers, there is little gain in combining their outputs; in fact, the results may even get worse. To attack this potential problem, we computed the correlations between the classifier score outputs which are listed in Table 3.

We can see from Table 3 that LFCC is highly correlated with MFCC. This is an expected result, since both of them describe essentially the same quantity, spectral shape. Also, dynamic cepstral parameters are highly correlated with each other, which can be explained by the method they are computed: △△-MFCC is merely a differenced version of △-MFCC.

From the six data sets, LTAS and F0 are least correlated with the other feature sets. Based on these observations, we selected MFCC, LTAS and F0 for the evaluation of classifier fusion. The results for a test segment of length 1.8 seconds for the two best sub-classifiers and the fusion are compared in Table 4. It can be seen that by combining the data sets, the error rate is halved. This shows that the fusion strategy works as designed.

### 3.5. Weight selection

Next we study the effect of the weight selection. The results for equal weights, Fisher's criterion, and exhaustive search are compared in Fig. 2 for different input segment lengths.

Figure 2 indicates that the selection of weights has some importance. With exhaustive search, we can find the optimum weight combination for given model size and test segment length. However, this is computationally intensive approach and furthermore, the weights computed in this way do not give any insight into data sets themselves. Thus, the Fisher's criterion seems more appropriate choice for practical use. Both of these approaches outperform the equal weights case, which suggests that the feature sets, indeed, have unequal discrimination powers (reliability).

We continue by fixing the weights according to Fisher's cri-

Table 3: *Correlations of the feature sets.*

| | MFCC | LFCC | LTAS | φ-MFCC | φφ-MFCC |
|---|---|---|---|---|---|
| **MFCC** | | | | | |
| **LFCC** | 0.88 | | | | |
| **LTAS** | 0.19 | 0.13 | | | |
| **φ-MFCC** | 0.72 | 0.62 | 0.14 | | |
| **φφ-MFCC** | 0.69 | 0.62 | 0.10 | 0.94 | |
| **F0** | 0.31 | 0.09 | 0.02 | 0.25 | 0.20 |

Table 4: Comparison of the two best subclassifiers and the classifier fusion.

| | Error rate | Avg. confidence |
|---|---|---|
| LTAS | 5.45 % | 0.53 |
| MFCC | 6.36 % | 0.14 |
| Fusion | 2.72 % | 0.19 |

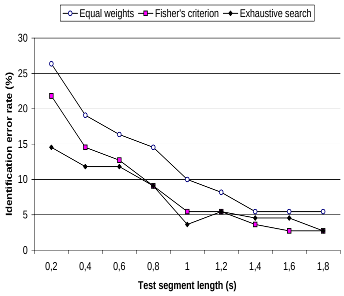


Figure 2: *Comparison of weight selection.*

terion and examine what is the effect of excluding the best feature set, LTAS. The results are compared with MFCC in the Fig. 3. We observe that excluding LTAS increases error rate. Therefore, the gain in the fusion is mostly due to LTAS feature set. Fusion without LTAS is close to the results obtained using MFCC alone. For very short segments, the classifier fusion still improves recognition accuracy.

## 4. Conclusions

Information fusion of diverse data sets is a difficult task. We have evaluated the performance of classifier output fusion for multiparametric speaker identification in the case of dissimilarity-based classifiers. The results indicate that by using multiple uncorrelated feature sets, the recognition performance of the fusion system is better than any of the sub-classifiers alone.

## 5. Acknowledgements

This work was partially carried out under the project ”The Joint Project Finnish Speech Technology” supported by the National Technology Agency (agreements 40285/00, 40406/01, 40238/02) and titled ”Speaker Recognition” (University of Helsinki ProjNr 460325).

## 6. References

[1] J. Campbell, ”Speaker Recognition: A Tutorial,” *Proceedings of the IEEE*, 85(9), pp. 1437-1462, 1997.

[2] R.P.W. Duin, ”The Combining Classifier: To Train Or Not To Train?,” *Proc. 16th International Conference on Pattern Recognition (ICPR 2002)*, Quebec City, Canada, pp. 765-770, 2002.

[3] P. Fränti, J. Kivijärvi, ”Randomized Local Search Algorithm for the Clustering Problem,” *Pattern Analysis and Applications*, 3(4), pp. 358-369, 2000.

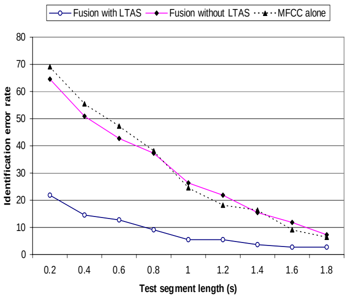


Figure 3: *Excluding the best feature set (LTAS).*

[4] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, New York, 1972.

[5] A. Iivonen, K. Harinen, M. Horppila, L. Keinänen, J. Kirjavainen, H. Liisanantti, E. Meister, L. Perälä, L. Tuuri, L. Vilhunen, ”Development of a Multiparametric Speaker Profile for Speaker Recognition,” manuscript, accepted for publication in *The 15th Int. Congress on Phonetic Sciences (ICPhS 2003)*, Barcelona, Spain, 2003.

[6] T. Kinnunen, ”Designing a Speaker-Discriminative Adaptive Filter Bank for Speaker Recognition,” *Proc. ICSLP 2002*, pp. 2325-2328, Denver, USA, 2002.

[7] J. Kittler, M. Hatef, R.P.W. Duin, J. Matas, ”On Combining Classifiers,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(3), pp. 226-239, 1998.

[8] K.P. Markov, S. Nakagawa, ”Text-Independent Speaker Recognition Using Multiple Information Sources,” *Proc. ICSLP 1998*, pp. 173-176, Sydney, Australia, 1998.

[9] V. Moonasar, G.K. Venayagamoorthy, ”A Committee of Neural Networks for Automatic Speaker Recognition (ASR) Systems,” *Proc. Int. Joint Conference on Neural Networks*, pp. 2936-2940, Washington DC, USA, 2001.

[10] P.C. Nguyen, M. Akagi, T.B. Ho, ”Temporal Decomposition: A Promising Approach to VQ-Based Speaker Identification,” manuscript, accepted for publication in *ICASSP 2003*, Hong Kong, 2003.

[11] P. Rose, *Forensic Speaker Identification*, Taylor & Francis, London, 2002.

[12] S. Slomka, S. Sridharan, V. Chandran, ”A Comparison of Fusion Techniques in Mel-Cepstral Based Speaker Idenficication,” *Proc. ICSLP 1998*, Sydney, Australia, 1998.

[13] F.K. Soong, A.E. Rosenberg, B.-H. Juang, and L.R. Rabiner, ”A Vector Quantization Approach to Speaker Recognition,” *AT & T Technical Journal*, 66, pp. 14-26, 1987.

[14] F.K. Soong and A.E. Rosenberg, ”On the Use of Instantaneous and Transitional Spectral Information in Speaker Recognition,” *IEEE Trans. Acoustics, Speech and Signal Processing*, 36(6), pp. 871-879, 1988.

[15] F. Weber, L. Manganaro, B. Peskin, E. Shriberg, ”Using Prosodic and Lexical Information for Speaker Identification,” *Proc. ICASSP 2002*, pp. 141-144, Orlando, USA, 2002.

## Publication P4

T. Kinnunen, V. Hautamäki, P. Fränti, Fusion of Spectral Feature Sets for Accurate Speaker Identification, *Proceedings of the 9th International Conference on Speech and Computer* (SPECOM 2004), pp. 361-365, St. Petersburg, Russia, September 20-22, 2004.

# Fusion of Spectral Feature Sets for Accurate Speaker Identification

Tomi Kinnunen, Ville Hautamäki, and Pasi Fränti
Department of Computer Science
University of Joensuu, Finland
{tkinnu,villeh,franti}@cs.joensuu.fi

13th August 2004

## Abstract

Several features have been proposed for automatic speaker recognition. Despite their noise sensitivity, low-level spectral features are the most popular ones because of their easy computation. Although in principle different spectral representations carry similar information (spectral shape), in practice the different features differ in their performance. For instance, LPC-cepstrum picks more "details" of the short-term spectrum than the FFT-cepstrum with the same number of coefficients. In this work, we consider using multiple spectral presentations simultaneously for improving the accuracy of speaker recognition. We use the following feature sets: mel-frequency cepstral coefficients (MFCC), LPC-cepstrum (LPCC), arcus sine reflection coefficients (ARCSIN), formant frequencies (FMT), and the corresponding delta-parameters of all feature sets. We study the two ways of combining the feature sets: feature-level fusion (feature vector concatenation), score-level fusion (soft combination of classifier outputs), and decision-level fusion (combination of classifier decision).

## 1 Introduction

*Front-end* or *feature extractor* is the first component in an automatic speaker recognition system. Feature extraction transforms the raw speech signal into a compact but effective representation that is more stable and discriminative than the original signal.

Speaker differences in the acoustic signal are coded in complex way in both *segmental* (phoneme) level, *prosodic* (suprasegmental) level and *lexical* level. Modeling of prosody and lexical features has shown great promises in automatic speaker recognition systems lately [19]. However, the segmental features are still the most popular approach because of their easy extraction and modeling.

In most automatic speaker and speech recognition systems, segmental features are computed over a short time window (around 30 ms), which is shifted forward by a constant amount (around 50-70 % of the window length). Two most popular features are *mel-frequency cepstral coefficients* (MFCC) and *linear predictive cepstral coefficients* (LPCC) [9]. These features are often augmented with the corresponding *delta features*. The delta features give an estimate of the time derivative of each feature, and therefore they are expected to carry information about vocal tract dynamics. Sometimes, the delta parameters of the delta parameters (*double-deltas*) are also used, as well as the *fundamental frequency* (F0). For each time window, the different features are simply concatenated into a one higher dimensional (around $d = 40$) feature vector.

Augmenting the static parameters with the corresponding delta parameters can be seen as one way to perform *information fusion* by using different information sources, in the hope that the recognition accuracy will be better. The vector level feature augmentation is denoted here as *feature-level fusion*.

Although feature-level fusion may improve recognition accuracy, it has several shortcomings. First, fusion becomes difficult if a feature is missing (e.g. F0 of unvoiced sounds) or the frame rates of the features are different. Second, the number of training vectors needed for robust density estimation increases exponentially with the dimensionality. This phenomenon is known as the *curse of dimensionality* [2].

An alternative to feature-level fusion is to model each different feature set separately, design a specialized classifier for this feature set, and combine the classifier output scores. Each of the different feature sets acts as an independent "expert", giving its opinion about the unknown speaker's identity. The *fusion rule* then combines the individual experts' match scores. This approach is referred here as *score-level fusion*.

Score-level fusion strategy can also be abstracted by hardening the decisions of the individual classifiers. In other words, each of the experts produces a speaker label, and the fusion rule combines the individual decisions e.g. by majority voting. We call this fusion strategy *decision-level fusion*.

In a previous work [11], we documented our implementation of an score-level fusion system that uses vector quantization (VQ) based classifiers. The system can be used for combining an arbitrary number of diverse feature sets varying in scale, dimensionality and the number of vectors. For each speaker and feature set, a codebook is trained using a clustering algorithm. In the recognition phase, features extracted from the unknown speaker are presented to the corresponding classifiers. Each vector quantizer computes average quantization distortion of the unknown sequence. Within each quantizer, the distortions are scaled so that they sum up to unity over different speakers. The scaled distortions are then weighted and summed to give the final combined match score. The weights are feature set depended, but same for all speakers.

Extensive experiments in [10] were carried out on two corpora, a 100 speaker subset of the American English TIMIT corpus [16] and a corpus of 110 native Finnish speakers, documented in [6]. There were some differences between the two corpora and feature sets, but these were relatively small; many of the feature sets reached error rates close to zero. Therefore, it seemed unnecessarily to experiment with different fusion strategies with these features since the individual features already performed so well. The reason for this is that the both corpora were recorded in unrealistic laboratory conditions. We have found out that the performance decreases radically in real-world conditions.

In this study, we have selected the spectral parameters that seem most promising in the light of the findings of [10]. We study these on a more realistic corpus, a subset of the 1999 Speaker Recognition Evaluation Corpus. We aim at studying whether different spectral feature sets can complement each other, and which one of the fusion strategies (feature, score, and decision-level) is most appropriate for VQ-based classification in practice.

# 2 Selected Spectral Features

## 2.1 Mel-Frequency Cepstral Coefficients

*Mel-frequency cepstral coefficients* (MFCC) are motivated by studies of the human peripheral auditory system. First, the pre-emphasized and windowed speech frame is converted into spectral domain by the fast Fourier transform (FFT). The magnitude spectrum is then smoothed by a bank of triangular bandpass filters that emulate the critical band processing of the human ear. Each of the bandpass filters computes a weighted average of that subband, which is then compressed by logarithm. The log-compressed filter outputs are then decorrelated using the discrete cosine transform (DCT). The zeroth cepstral coefficient is discarded since it depends on the intensity of the frame.

There are several analytic formulae for the mel scale used in the filterbank design. In this study, we use the following mapping [7]:

$$f_{\text{mel}}(f_{\text{Hz}}) = \frac{1000}{\log_{10} 2} \log_{10}\left(1 + \frac{f_{\text{Hz}}}{1000}\right), \tag{1}$$

having the inverse mapping

$$f_{\text{Hz}}(f_{\text{mel}}) = 1000\Big(1 + 10^{\frac{\log_{10} 2}{1000} f_{\text{mel}}}\Big). \tag{2}$$

First, the number of filters ($M$) is specified. Filter center frequencies are then determined by dividing the mel axis into $M$ uniformly spaced frequencies and computing the corresponding frequencies in the hertz scale with the inverse mapping. The filterbank itself is then designed so that the center frequency of the $m$th filter is the low cutoff frequency of the $(m+1)$th filter. The low and high cutoff frequencies of the first and last filters are set to zero and Nyquist frequencies, respectively.

## 2.2 LPC-Derived Features

In addition to the MFCC coefficients, we consider the following representations that are computed via linear prediction analysis: *arcus sine reflection coefficients* (ARCSIN), *linear predictive cepstral coefficients* (LPCC), and formant frequencies (FMT).

The linear predictive model of speech production [17, 5] is given in the time domain:

$$s[n] \approx \sum_{k=1}^{p} a[k]s[n-k], \tag{3}$$

where $s[n]$ denotes the speech signal samples, $a[k]$ are the *predictor coefficients* and $p$ is the *order* of the predictor. The total squared prediction error is:

$$E = \sum_{n} \left(s[n] - \sum_{k=1}^{p} a[k]s[n-k]\right)^2. \tag{4}$$

The objective of linear predictive analysis is to determine the coefficients $a[k]$ for each speech frame so that (4) is minimized. The problem can be solved by setting the partial derivatives of (4) with respect to $a[k]$ to zero. This leads to so called *Yule-Walker equations* that can be efficiently solved using so-called *Levinson-Durbin recursion* [8].

The Levinson-Durbin recursion generates as its side product so-called *reflection coefficients*, denoted here as $k[i], i = 1, \ldots, p$. The name comes from the multitube model, each reflection coefficient characterizing the transmission/reflection of the acoustic wave at each tube junction. Instead of using the reflection coefficients, we use instead the numerically more stable *arcus sine reflection coefficients* [3].

In the frequency domain, the linear predictive coefficients specify an IIR filter with the transfer function:

$$H(z) = \frac{1}{1 - \sum_{k=1}^{p} a[k]z^{-k}}. \tag{5}$$

The *poles* of the filter (5) are the zeroes of the denominator. They are denoted here as $z_1, z_2, \ldots, z_p$, and they can be found by numerical root-finding techniques. The coefficients $a[k]$ are real, which restricts the poles to be either real or occur in complex conjugate pairs.

If the poles are well separated in the complex plane, they can be used for estimating the formant frequencies [5]:

$$\hat{F}_i = \frac{F_s}{2\pi} \tan^{-1}\left(\frac{\text{Im } z_i}{\text{Re } z_i}\right). \tag{6}$$

Table 1: Summary of the NIST-1999 subset

| | |
|---|---|
| Language | English |
| Speakers | 230 |
| Speech type | Conversational |
| Quality | Telephone |
| Sampling rate | 8.0 kHz |
| Quantization | 8-bit $\mu$-law |
| Training speech (avg.) | 119.0 sec. |
| Evaluation speech (avg.) | 30.4 sec. |

Given the LPC coefficients $a[k], k = 1, \ldots, p$, the LPCC coefficients are computed using the recursion [1]:

$$c[n] = \begin{cases} a[n] + \sum_{k=1}^{n-1} \frac{k}{n} c[k] a[n-k], & 1 \leq n \leq p \\ \sum_{k=n-p}^{n-1} \frac{k}{n} c[k] a[n-k], & n > p. \end{cases} \tag{7}$$

### 2.3 Delta Features

There are two different ways for computing the delta features: (1) differentiating, and (2) fitting a polynomial expansion. We have found out that the differentiator method works systematically better than the first order polynomial, i.e. the linear regression method [10]. Let $f_k[i]$ denote the $i$th feature in the $k$th time frame. The differentiator method estimates the time derivative of the feature as follows [5]:

$$\Delta f_k[i] = f_{k+M}[i] - f_{k-M}[i], \tag{8}$$

where $M$ is typically 1-3 frames.

## 3 Experiments

### 3.1 Speech Material and Parameter Setup

For the experiments, we used a subset of the *NIST 1999 speaker recognition evaluation corpus* [18] (see Table 1). We decided to use the data from the male speakers only. For training, we used both the “a” and “b” sessions. For identification, we used the one speaker test segments from the same telephone line. In general it can be assumed that if two calls are from different lines, the handsets are different, and if they are from the same line, the handsets are the same [18]. In other words, the training and matching conditions have very likely the same handset type (electret/carbon button) for each speaker, but different speakers can have different handsets. The total number of test segments for this condition is 692.

The parameters for different feature sets and training algorithm were based on our previous experiments with the NIST corpus [12]. The frame length and shift were set to 30 ms and 20 ms, respectively, and the window function was Hamming. For MFCC computation, the number of filters was set to 27, and the number of coefficients was 12. For LPCC, ARCSIN and FMT, we used LPC predictor of order $p = 20$. We selected 12 LPCC and ARCSIN coefficients, and 8 formant frequencies. The delta features were computed using the differentiator method with $M = 1$. Throughout the experiments, codebook size was fixed to 64, and the codebooks were trained using the Linde-Buzo-Gray (LBG) clustering algorithm [15].

### 3.2 Individual Feature Sets

The identification error rates of the individual feature sets are reported in Table 2. The static features (MFCC, LPCC, ARCSIN, FMT) all give good results. The delta features, on the other hand, are worse than the static features. The error rate of delta formants is very high.

Table 2: Accuracies of the individual feature sets

| *Static features* | | *Dynamic features* | |
|---|---|---|---|
| Feature set | Error rate (%) | Feature set | Error rate (%) |
| MFCC | 16.8 | ΔMFCC | 21.2 |
| LPCC | 16.0 | ΔLPCC | 25.1 |
| ARCSIN | 17.1 | ΔARCSIN | 28.6 |
| FMT | 19.4 | ΔFMT | 70.5 |

### 3.3 Fusion Results

Next, we experimented by fusing the static parameters and their corresponding delta features using all the three strategies. We also combined all the 8 feature sets. For the feature-level fusion, each feature vector was normalized by its norm, and the normalized vectors were then concatenated. For the score-level fusion, we used the normalized VQ distortions giving unity weights to all feature sets [11]. For the decision-level fusion, we use majority voting, by selecting speaker label that is voted most by all classifiers. If no speaker received majority, then speaker label is selected randomly from the highest number of votes.

The fusion results are shown in Table 3, along with the best individual performance from the pool. The score-level fusion gives the best result in all cases fusing feature with it’s delta parameters, except with the formant data for which fusion is not succesfull. The reason for poor performance in this case is the poor performance of delta formants. Situation could be alleviated by deemphasizing the delta formants.

It can be seen that the feature-level fusion improves the performance over the individual classifier in the case of MFCC and its delta features. However, in all other cases it degrades the performance. The decisionl-level fusion is the best fusion strategy, when all feature sets are used. Majority voting is not applicable for only two classifier system as seen for all other cases, where performance is degraded.

In the case, when user has only feature set and its delta parameters, results show that the score-level fusion seems to be the method to be preferred in the case of reliable experts. However, if some of the “experts” produces a lot of classification errors (ΔFMT), the weight for the unreliable features or feature sets should be set small. In this study, we did not attempt to weight individual features or

Table 3: Accuracies of the fused systems.

| Combination | Best individual | Feature-level | Score-level | Decision-level | Oracle |
|---|---|---|---|---|---|
| MFCC + ΔMFCC | 16.8 | 15.8 | **14.6** | 19.0 | 12.3 |
| LPCC + ΔLPCC | 16.0 | 19.8 | **14.7** | 20.5 | 12.6 |
| ARCSIN + ΔARCSIN | 17.1 | 18.2 | **16.8** | 22.8 | 15.0 |
| FMT + ΔFMT | **19.4** | 29.9 | 52.0 | 44.9 | 18.5 |
| All feature sets | 16.0 | 21.2 | 15.2 | **12.6** | 7.8 |

Table 4: $Q$ statistic between all classifier pairs.

| | MFCC | ΔMFCC | LPCC | ΔLPCC | ARCSIN | ΔARCSIN | FMT | ΔFMT |
|---|---|---|---|---|---|---|---|---|
| MFCC | | 0.916 | 0.976 | 0.861 | 0.953 | 0.875 | 0.925 | 0.594 |
| ΔMFCC | | | 0.909 | 0.934 | 0.869 | 0.847 | 0.838 | 0.527 |
| LPCC | | | | 0.907 | 0.984 | 0.929 | 0.952 | 0.637 |
| ΔLPCC | | | | | 0.866 | 0.898 | 0.854 | 0.517 |
| ARCSIN | | | | | | 0.948 | 0.956 | 0.753 |
| ΔARCSIN | | | | | | | 0.921 | 0.505 |
| FMT | | | | | | | | 0.842 |

feature sets. In the case of feature-level fusion, it is not obvious how the individual features should be weighted.

### 3.4 Feature Set Diversity

Although the fusion improves performance in most cases, the gain is rather low. Intuitively, if the different classifiers misclassify the same speech segments, we do not expect as much improvement as in the case where they complement each other. There are several indices to assess the interrelationships between the classifiers in a classifier ensemble [4].

Given classifiers $i$ and $j$, we compute the $Q$ *statistic* [4]:

$$Q_{i,j} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}, \qquad (9)$$

where $N^{00}$ is the number of test segments misclassified by both $i$ and $j$; $N^{11}$ is the number of segments correctly classified by both; $N^{10}$ and $N^{01}$ are the numbers of segments misclassified by one and correctly classified by the other. It can be easily verified that $-1 \leq Q_{i,j} \leq 1$. The $Q$ value can be considered as a correlation measure between the classifier decisions.

The $Q$ statistics between all feature set pairs are shown in Table 4. It can be seen that all values are positive and relatively high, which indicates that the classifiers function essentially the same way. In other words, the classifiers are *competitive* instead of *complementary* [13]. This partially explains why the performance is not greatly improved by fusion. Interestingly, although the performance of delta formants is very poor, it has lowest $Q$ values on average. This means that delta formants make different decisions compared to other feature sets.

Table 5: Distribution of the number of correct votes.

| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 155 | 269 | 72 | 39 | 43 | 23 | 22 | 15 | 54 |

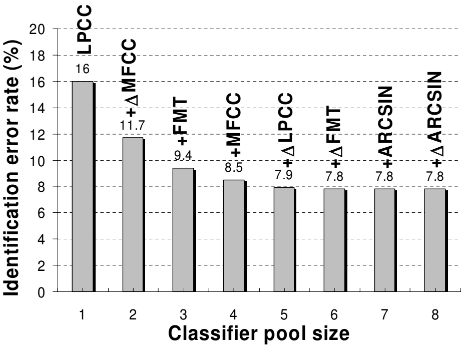


Figure 1: Performance of the "Oracle" classifier.

We can also analyze the difficulty of the test segments. Table 5 shows how many classifiers voted correctly on the same test segments out of 692. Interestingly, most test segments are voted correctly by 6,7 or 8 classifiers (72 %), which means that most of the test segments are relatively "easy". However, in the other end, there were 54 test segments (8 %) that no classifier voted correctly. This shows that some speakers are more difficult to recognize.

### 3.5 "Oracle" Classifier

We can estimate the lower limit of the identification error rate using an abstract machine called *Oracle classifier* [14]. The Oracle assigns correct class label to the test segment if at least one feature set classifies it correctly. Figure 1 shows the performance of this abstract classifier as a function of the classifier pool size. New classifiers are added to the pool in a greedy manner, starting from the best individual feature set (LPCC) and adding the feature set that decreses the error rate most. The lowest error rate (7.8 %) is reached by using six feature sets. The test segments classified correctly by the ARCSIN and $\Delta$ARCSIN feature sets are already classified correctly by some of the other feature sets. It must be emphasized that this is only a theoretical classifier, giving an idea of the lowest possible error rate if the diversity of the feature sets was taken fully into account.

## 4 Conclusions

We have compared and analyzed different ways of using several spectral feature sets for speaker identification. From the individual feature sets considered, linear predictive cepstral coefficients performed the best giving an error rate of 16.0 %. The best fusion result reduced this to 12.6 %, and it was obtained by decision-level fusion with all feature sets. If many different feature sets are availeble we recommend to use majority voting, otherwise in more traditional setting score-level fusion is the best.

Although fusion improves performance, the difference is not big. The analysis of the classifier diversities showed that the different feature sets classify speakers essentially in the same way. It is possible to reduce the error rate further by setting feature set depended weights reflecting the relative importances of the feature set. In future, we plan to use speaker-dependent weights and recent advances in information fusion, e.g. *decision templates* and *consensus classification* [13].

## 5 Acknowledgements

The work of V. Hautamäki was supported by the National Technology Agency of Finland (project "Puhetekniikan uudet menetelmät ja sovellukset", TEKES dnro 8713103).

## References

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

[2] C.M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1996.

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

[4] C.A.Shipp and L.I.Kuncheva. Relationships between combination methods and measures of diversity in combining classifiers. *Information Fusion*, 3:135–148, 2002.

[5] J.R. Jr. Deller, J.H.L. Hansen, and J.G. Proakis. *Discrete-Time Processing of Speech Signals*. IEEE Press, New York, second edition, 2000.

[6] P. Eskelinen-Rönkä. Report on the testing of *Puhujan Tunnistaja* database software. MSc Thesis, Department of General Phonetics, University of Helsinki, Helsinki, Finland, 1997. (in finnish).

[7] G. Fant. *Acoustic Theory of Speech Production*. The Hague, Mouton, 1960.

[8] J. Harrington and S. Cassidy. *Techniques in Speech Acoustics*. Kluwer Academic Publishers, Dordrecht, 1999.

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

[10] T. Kinnunen. *Spectral Features for Automatic Text-Independent Speaker Recognition*. Licentiate's thesis, University of Joensuu, Department of Computer Science, Joensuu, Finland, 2004.

[11] T. Kinnunen, V. Hautamäki, and P. Fränti. On the fusion of dissimilarity- based classifiers for speaker identification. In *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, pages 2641–2644, Geneva, Switzerland, 2003.

[12] T. Kinnunen, E. Karpov, and P. Fränti. Real-time speaker identification. In *Proc. Int. Conf. on Spoken Language 2004 (ICSLP 2004)*, Jeju Island, Korea, 2004. (to appear).

[13] L.I.Kuncheva. *Fuzzy Classifier Design*. Physica Verlag, Heidelberg, 2000.

[14] L.I.Kuncheva, C.J.Whitaker, C.A.Shipp, and R.P.W.Duin. Limits on the majority vote accuracy in classifier fusion. *Pattern Analysis and Applications*, 6:22–31, 2003.

[15] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, 1980.

[16] Linguistic data consortium. WWW page, December 2004. `http://www.ldc.upenn.edu/`.

[17] J. Makhoul. Linear prediction: a tutorial review. *Proceedings of the IEEE*, 64(4):561–580, 1975.

[18] A. Martin and M. Przybocki. The NIST 1999 speaker recognition evaluation - an overview. *Digital Signal Processing*, 10(1-18):1–18, 2000.

[19] D.A. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adami, Q. Jin, D. Klusacek, J. Abramson, R. Mihaescu, J. Godfrey, D. Jones, and B. Xiang. The SuperSID project: exploiting high-level information for high-accuracy speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)*, pages 784–787, Hong Kong, 2003.

5

## Publication P5

T. Kinnunen, E. Karpov, P. Fränti, Real-Time Speaker Identification and Verification, Accepted for publication in *IEEE Transactions on Speech and Audio Processing.*

# Real-Time Speaker Identification and Verification

Tomi Kinnunen, Evgeny Karpov, and Pasi Fränti

Department of Computer Science, University of Joensuu
P.O. Box 111, FIN-80101 Joensuu, FINLAND,

*Abstract*— **In speaker identification, most of the computation originates from the distance or likelihood computations between the feature vectors of the unknown speaker and the models in the database. The identification time depends on the number of feature vectors, their dimensionality, the complexity of the speaker models and the number of speakers. In this paper, we concentrate on optimizing vector quantization (VQ) based speaker identification. We reduce the number of test vectors by pre-quantizing the test sequence prior to matching, and the number of speakers by pruning out unlikely speakers during the identification process. The best variants are then generalized to Gaussian mixture model (GMM) based modeling. We apply the algorithms also to efficient cohort set search for score normalization in speaker verification. We obtain a speed-up factor of 16:1 in the case of VQ-based modeling with minor degradation in the identification accuracy, and 34:1 in the case of GMM-based modeling. An equal error rate of 7 % can be reached in 0.84 seconds on average when the length of test utterance is 30.4 seconds.**

*Index Terms*— **Speaker recognition, real-time, speaker pruning, pre-quantization, VQ, GMM**

## I. Introduction

Speaker recognition refers to two different tasks: *speaker identification* (SI) and *speaker verification* (SV) [1]–[3]. In the identification task, an unknown speaker $X$ is compared against a database of known speakers, and the best matching speaker is given as the identification result. The verification task consists of making a decision whether a voice sample was produced by a claimed person.

### A. Motivation

Applications of speaker *verification* can be found in biometric person authentication such as an additional identity check during credit card payments over the Internet. The potential applications of speaker identification can be found in multi-user systems. For instance, in *speaker tracking* the task is to locate the segments of given speaker(s) in an audio stream [4]–[7]. It has potential applications in automatic segmentation of teleconferences and helping in the transcription of courtroom discussions.

Speaker identification could be used in *adaptive user interfaces*. For instance, a car shared by many people of the same family/community could recognize the driver by his/her voice, and tune the radio to his/her favorite channel. This particular application concept belongs to the more general group of *speaker adaption methods* that are already employed in speech recognition systems [8], [9]. Speaker-specific codecs in *personal speech coding* have been also demonstrated to give smaller bit rates as opposed to a universal speaker-independent codec [10].

Speaker identification have also been applied to the verification problem in [11], where the following simple rank-based verification method was proposed. For the unknown speaker's voice sample, $K$ nearest speakers are searched from the database. If the claimed speaker is among the $K$ best speakers, the speaker is accepted and otherwise rejected. Similar verification strategy is also used in [12].

Speaker identification and adaptation have potentially more applications than verification, which is mostly limited to security systems. However, the verification problem is still much more studied, which might be due to (1) lack of applications concepts for the identification problem, (2) increase in the expected error with growing population size [13], and (3) very high computational cost. Regarding the identification accuracy, it is not always necessary to know the exact speaker identity but the *speaker class* of the current speaker is sufficient (speaker adaptation). However, this has to be performed in real-time. In this paper, we focus on decreasing the computational load of identification while attempting to keep the recognition accuracy reasonably high.

### B. Review of Computational Speed-Up Methods

A large number of methods have been proposed for speeding up the *verification* process. Specifically, Gaussian mixture model (GMM) based verification systems [14], [15] have received much attention, since they are considered as the state-of-the-art method for text-independent recognition. Usually, speaker-dependent GMMs are derived from a speaker-independent *universal background model* (UBM) by adapting the UBM components with *maximum a posteriori* (MAP) adaptation using each speaker's personal training data [15]. This method incudes a natural hierarchy between the UBM and the personal speaker models; for each UBM Gaussian component, there is a corresponding adapted component in the speaker's personal GMM. In the verification phase, each test vector is scored against all UBM Gaussian components, and a small number (typically 5) of the best scoring components in the corresponding speaker-dependent GMMs are scored. This procedure effectively reduces the amount of needed density computations.

In addition to the basic UBM/GMM approach, a number of other hierarchical methods have been considered for GMM.

Corresponding author: Tomi Kinnunen. Contact address: Department of Computer Science, University of Joensuu, P.O. Box 111, FIN-80101 Joensuu, FINLAND. E-mail: Tomi.Kinnunencs.joensuu.fi, Tel. +358 13 251 7905, Telefax. +358 13 251 7955.

Beigi & al. [12] propose a hierarchical structuring of the speaker database with the following merging strategy. Two closest GMMs are merged, and the process is repeated until the number of GMMs is 1. A similar approach using the *ISODATA* clustering algorithm has been recently proposed by Sun & al. [16] for the identification task. They report identification accuracy close to full search with speed-up factors from 3:1 to 6:1. The relative speed-up of their algorithm was higher for increased number of speakers.

Auckenthaler and Mason [17] applied UBM-like *hash model*, in which for each Gaussian component, there is a shortlist of indices of the expected best scoring components for each individual GMM. Using the shortlist of the hash model, only the corresponding components in the individual GMM are then scored. By increasing the lengths of the shortlists, scores can be computed more accurately, but with an increased computational overhead. Auckenthaler and Mason reported a speed-up factor of about 10:1 with a minor degradation in the verification performance.

McLaughlin & al. [18] have studied two simple speed-up methods for the GMM/UBM-based verification system: (1) decreasing the UBM size, and (2) decimating the sequence of test vectors with three simple methods. They noticed that the UBM could be reduced by a factor of 4, and the test sequence up to a factor of about as high as 20 without affecting the verification performance. McLaughlin & al. [18] state (p. 1218):

> "What is surprising is the degree to which feature vectors can be decimated without loss in accuracy. . . . The key factor seems to be the acoustic variety of the vectors scored, not the absolute number of vectors."

However, they did not experiment the combination of decimation and reduced UBM.

An efficient GMM-based speaker identification system has also been presented by Pellom and Hansen [19]. Since the adjacent feature vectors are correlated and the order of the vectors does not affect the final score, the vector sequence can be reordered so that non-adjacent feature vectors are scored first. After the scoring, worst scoring speakers are pruned out using a *beam search* technique where the beam width is updated during processing. Then, a more detailed sampling of the sequence follows. The process is repeated as long as there are unpruned speakers or input data left, and then the best scoring speaker is selected as the winner. Pellom and Hansen reported speed-up factor of 6:1 relative to the baseline beam search.

Recently, more advanced hierarchical models have been proposed for efficient speaker verification [20], [21]. Xiang and Berger [20] construct a tree structure for the UBM. Multilevel MAP adaptation is then used for generating the speaker-specific GMMs with a tree structure. In the verification phase, the target speaker scores and the UBM scores are combined using an MLP neural network. Xiang and Berger reported a speed-up factor of 17:1 with a 5 % relative increase in the EER. They also compared their method with the hash model of Auckenthaler and Mason [17]. Although the method of Xiang and Berger gave slightly better verification accuracy (from EER of 13.9 % to EER of 13.5 %) and speed-up (from 15:1 to 17:1) as compared to the hash GMM, the Xiang's and Berger's method is considerably more complex than the hash GMM.

### C. Contributions of This Study

The literary review herein shows that most of the speed optimizations have been done on GMM-based systems. In this study, we optimize *vector quantization* (VQ) based speaker recognition, because it is straightforward to implement, and according to our experiments, it yields equally good or better identification performance than the baseline GMM based on maximum likelihood training using the EM algorithm.

Most of the computation time in VQ-based speaker identification consists of distance computations between the unknown speaker's feature vectors and the models of the speakers enrolled in the system database. *Speaker pruning* [19], [22], [23] can be used to reduce the search space by dropping out unlikely speakers "on the fly" as more speech data arrives. We survey and compare several speaker pruning variants. We also propose a new speaker pruning variant called *confidence-based speaker pruning*. The idea is to wait for more speech data until we are confident to decide whether a certain speaker could be safely pruned out.

We optimize the other components of the recognition system as well. We reduce the number of test sequence vectors by silence removal and pre-quantization, and show how the pre-quantization methods can be combined with the speaker pruning for more efficient identification. A *vantage-point tree* (VPT) [24] is used for indexing the speakers' code vectors for speeding up the nearest neighbor search. Our main contribution is a systematic comparison and combining of several optimization methods.

Although the framework presented in this study is built around VQ-based speaker modeling, the methods are expected to generalize to other modeling paradigms. We demonstrate this by applying the best pre-quantization and pruning variants to GMM-based identification.

Finally, we demonstrate that the methods apply also to the verification task. Pre-quantization is applied for searching a *cohort set* online for the client speaker during the verification process, based on the closeness to the input vectors. We propose a novel cohort normalization method called *fast cohort scoring* (FCS) which decreases both the verification time and the equal error rate.

The rest of the paper is organized as follows. In Section II, we review the baseline speaker identification, and consider the computational complexity issue in more detail, focusing on the real-time processing in general level. A detailed description of the speaker pruning algorithms follows then in Section III. In Section IV, we utilize the speed-up methods to the verification problem. Section V describes the experimental setup. Test results with discussion are given in Section VI, and conclusions are drawn in Section VII.

## II. VQ-Based Speaker Identification

### A. General Structure

The components of a typical VQ-based speaker identification [25]–[28] system are shown in Fig. 1. *Feature extraction*

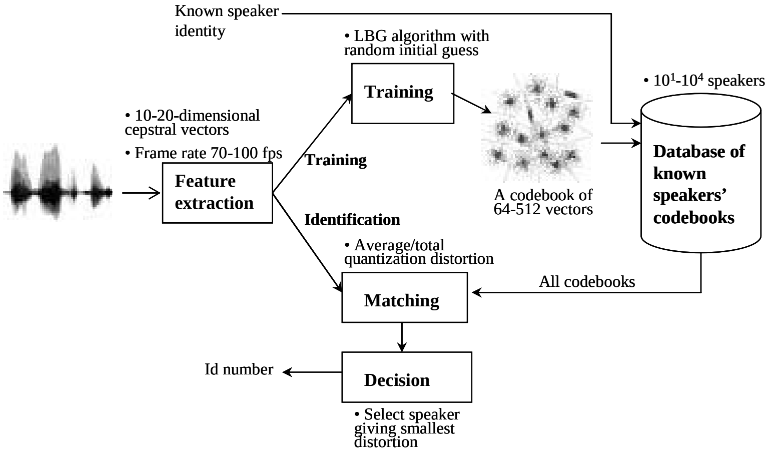

Known speaker identity
• LBG algorithm with random initial guess
Training
• $10^1$-$10^4$ speakers
• 10-20-dimensional cepstral vectors
• Frame rate 70-100 fps
Feature extraction
Training
Identification
A codebook of 64-512 vectors
Database of known speakers' codebooks
• Average/total quantization distortion
All codebooks
Matching
Id number
Decision
• Select speaker giving smallest distortion

Fig. 1. Typical VQ-based closed set speaker identification system.

transforms the raw signal into a sequence of 10- to 20-dimensional feature vectors with the rate of 70-100 frames per second. Commonly used features include *mel-cepstrum* (MFCC) and *LPC-cepstrum* (LPCC) [29], [30]. They measure short-term spectral envelope, which correlates with the physiology of the vocal tract.

In the training phase, a speaker model is created by clustering the training feature vectors into disjoint groups by a clustering algorithm. The *LBG algorithm* [31] is widely used due to its efficiency and simple implementation. However, other clustering methods can also be considered; a comparative study can be found in [32]. The result of clustering is a set of $M$ vectors, $C = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_M\}$, called a *codebook* of the speaker.

In the identification phase, unknown speaker's feature vectors are matched with the models stored in the system database. A *match score* is assigned to every speaker. Finally, a 1-out-of-$N$ decision is made. In a closed-set system this consists of selecting the speaker that yields the smallest distortion.

The match score between the unknown speaker's feature vectors $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ and a given codebook $C = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_M\}$ is computed as the *average quantization distortion* [25]:

$$D_{avg}(X, C) = \frac{1}{T} \sum_{i=1}^{T} e(\boldsymbol{x}_i, C), \tag{1}$$

where $e(\boldsymbol{x}_i, C) = \min_{\boldsymbol{c}_j \in C} \|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2$, and $\| \cdot \|$ denotes the Euclidean norm. Several modifications have been proposed to the baseline VQ distortion matching [27], [33]–[37].

### B. Time Complexity of Identification

In order to optimize speaker identification for real-time processing, first the dominating factors have to be recognized. In order to compute $D_{avg}(X, C)$, the nearest neighbors of each $\boldsymbol{x}_i \in X$ from the codebook $C$ are needed. With a simple linear search this requires $O(TM)$ distance calculations. Computation of the squared Euclidean distance between two $d$-dimensional vectors, in turn, takes $d$ multiplications and $d-1$ additions. Therefore, the total number of floating point operations (flops) for computing $D_{avg}(X, C)$ is $O(TMd)$. The computation of $D_{avg}(X, C)$ is repeated for all $N$ speakers, so the total identification time is $O(NTMd)$.

The efficiency of the feature extraction depends on the selected signal parametrization. Suppose that the extraction of one vector takes $O(f)$ flops. The total number of flops for feature extraction is then $O(Tf)$, where $T$ is the number of vectors. Notice that the feature extraction needs to be done only once. To sum up, total number of flops in identification is $O(Tf + NTMd) = O(T(f + NMd))$. The standard signal processing methods (MFCC, LPCC) themselves are very efficient. By assuming $f \ll NMd$, we can approximate the overall time as $O(TNMd)$.

The dimensionality $d$ is much smaller than $N$, $M$ and $T$. For instance, about 10-20 mel-cepstral coefficients is usually enough due the fast decay of the higher coefficients [29]. There is no reason to use a high number of cepstral coefficients unless they are properly normalized; the coefficients with a small magnitude do not contribute to the distance values much.

### C. Reducing the Computation Time

The dominating factors of the total identification time are the number of speakers ($N$), the number of vectors in the test sequence ($T$), and the codebook sizes ($M$). We reduce the number of speakers by pruning out unlikely speakers during the matching, and the number of vectors by silence removal and by pre-quantizing the input sequence to a smaller number of representative vectors prior to matching. In order to speed up the nearest neighbor search of the codebooks,

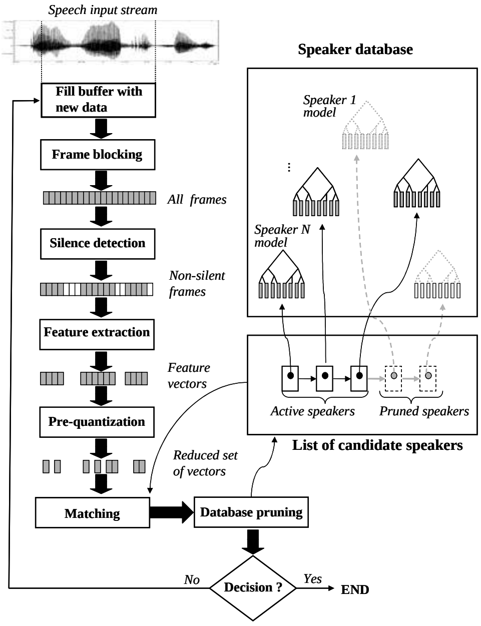


Fig. 2. Diagram of the real-time identification system.

we utilize *vantage-point trees* (VPT) [24] for indexing the code vectors in the models. VPT is a balanced binary search tree where each node represents a code vector. In the best case (fully balanced binary tree), the search takes $O(\log_2 M)$ distance computations. Unfortunately, the VPT as well as other indexing structures [38] apply only to metric distance functions. Since (1) does not satisfy the triangular inequality, we can index only the code vectors but not the codebooks themselves.

### D. Real-Time Speaker Identification

The proposed system architecture is depicted in Fig. 2. The input stream is processed in short buffers. The audio data in the buffer divided into frames, which are then passed through a simple energy-based silence detector in order to drop out non-information bearing frames. For the remaining frames, feature extraction is performed. The feature vectors are pre-quantized to a smaller number of vectors, which are compared against *active speakers* in the database. After the match scores for each speaker have been obtained, a number of speakers are pruned out so that they are not included anymore in the matching on the next iteration. The process is repeated until there is no more input data, or there is only one speaker left in the list of active speakers.

### E. Pre-quantization

In *pre-quantization* (PQ), we replace the original test vector sequence $X$ by a new sequence $\hat{X}$ so that $|\hat{X}| < |X|$. In order to gain time, the total time spent for the PQ and matching must be less than the matching time without PQ. The motivation for using PQ is that, in practise, the adjacent feature vectors are close to each other in the feature space because of the gradual movements of the articulators. McLaughlin & al. [18] applied three simple PQ methods prior to GMM matching, and reported that the test sequence could be compressed by a factor of 20:1 without compromizing the verification accuracy. This clearly suggests that there is a lot of redundancy in the feature vectors.

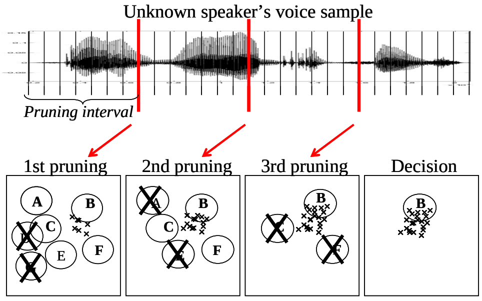


Fig. 3. Illustration of speaker pruning (pruning interval = 7 vectors).

We consider four different pre-quantization techniques: (1) *random subsampling*, (2) *averaging*, (3) *decimation*, and (4) *clustering-based PQ*. In random subsampling and averaging, the input buffer is processed in non-overlapping segments of $M$ >1 vectors. In random subsampling, each segment is represented by a random vector from the segment. In averaging, the representative vector is the centroid (mean vector) of the segment. In decimation, we simply take every $M$th vector of the test sequence, which corresponds to performing feature extraction with a smaller frame rate. In clustering-based PQ, we partition the sequence $X$ into $M$ clusters using the LBG clustering algorithm.

## III. Speaker Pruning

The idea of speaker pruning [19], [22], [23] is illustrated in Fig. 3. We must decide how many new (non-silent) vectors are read into the buffer before next pruning step. We call this the *pruning interval*. We also need to define the *pruning criterion*.

Figure 4 shows an example how the quantization distortion (1) develops with time. The bold line represents the correct speaker. In the beginning, the match scores oscillate, and when more vectors are processed, the distortions tend to stabilize around the expected values of the individual distances because of the averaging in (1). Another important observation is that a small amount of feature vectors is enough to rule out most of the speakers from the set of candidates.

We consider next the following simple pruning variants: *static pruning* [23], *hierarchical pruning* [22], and *adaptive pruning* [23]. We also propose a novel pruning variant called *confidence-based pruning*. The variants differ in their pruning criteria.

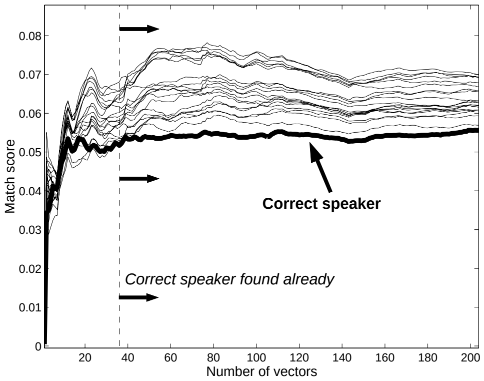


Fig. 4. Illustration of match score saturation ($N = 20$ speakers from the TIMIT corpus).

**Algorithm 1** Static Pruning (SP)

```
$A := \{1, 2, \ldots, N\}$ ; $X := \emptyset$ ;
while ($|A| > 1$) and (speech data left) do
    Insert $M$ new vectors into buffer $X$ ;
    Update $D_{avg}(X, C_i)$ for all $i \in A$ ;
    Prune out $K$ worst speakers from $A$ ;
end while
Decision: $i^* = \arg\min_i \{D(X, C_i) | i \in A\}$ ;
```

The following notations will be used:

| | |
|---|---|
| $X$ | Processing buffer for new vectors |
| $A$ | Indices of the active speakers |
| $C_i$ | Codebook of speaker $i$ |
| $N$ | Size of the speaker database |

### A. Static Pruning (SP)

The idea is to maintain an ordered list of the best matching speakers. At each iteration, $M$ new vectors are read in, match scores of the active speakers are updated, and $K$ worst matching speakers are pruned out (Algorithm 1). The update of the match scores can be done efficiently by using cumulative counts of the scores. The control parameters of the method are $M$ and $K$. Fig. 3 gives an example of the method with parameters $M = 7$ and $K = 2$.

### B. Hierarchical Pruning (HP)

For each speaker $i$, two codebooks are stored in the database: a *coarse* and a *detail* codebook, denoted here as $C_i^c$ and $C_i^d$, respectively. Both codebooks are generated from the same training data, but the coarse codebook is much smaller than the detail one: $|C_i^c| \ll |C_i^d|$. First, $K$ worst speakers are pruned out by matching the vectors against the coarse models. Scores of the remaining models are then recomputed using the detail models (Algorithm 2). The control parameters of the method are the the sizes of the codebooks and $K$.

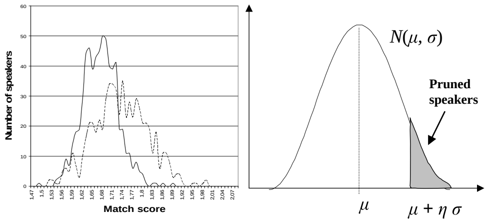


Fig. 5. Left: Match score distributions from the TIMIT corpus. Right: Illustration of the pruning threshold.

**Algorithm 2** Hierarchical Pruning (HP)

```
Let $C_c = \{C_1^c, \ldots, C_N^c\}$ be the coarse models ;
Let $C_d = \{C_1^d, \ldots, C_N^d\}$ be the detail models ;
$A := \{1, 2, \ldots, N\}$ ;
Read the whole test sequence into buffer $X$ ;
Compute $D_{avg}(X, C_i^c)$ for all $i \in A$ ;
Prune out $K$ worst speakers from $A$ ;
Compute $D_{avg}(X, C_i^d)$ for all $i \in A$ ;
Decision: $i^* = \arg\min_i \{D_{avg}(X, C_i^d) | i \in A\}$ ;
```

**Algorithm 3** Adaptive Pruning (AP)

```
$A := \{1, 2, \ldots, N\}$ ; $X := \emptyset$ ;
while ($|A| > 1$) and (speech data left) do
    Insert $M$ new vectors into buffer $X$ ;
    Update $D_{avg}(X, C_i)$ for all $i \in A$ ;
    Update Pruning threshold $\Theta$ ;
    Prune out speaker $i$ if $D_{avg}(X, C_i) > \Theta$ ;
end while
Decision: $i^* = \arg\min_i \{D_{avg}(X, C_i) | i \in A\}$ ;
```

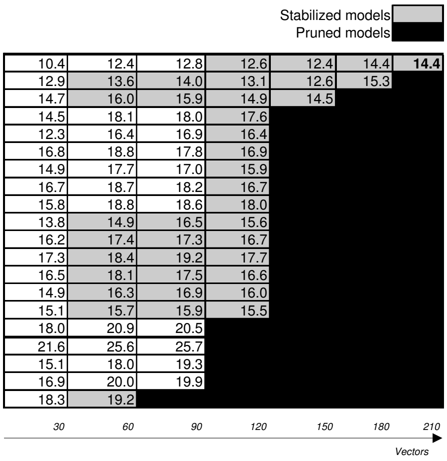
### C. Adaptive Pruning (AP)

Instead of pruning a fixed number of speakers, a pruning threshold $\Theta$ based on the distribution of the scores is computed, and the speakers whose score exceeds this are pruned out (see Algorithm 3). The pruning threshold $\Theta$ is computed as

$$\Theta = \mu_D + \eta \cdot \sigma_D, \tag{2}$$

where $\mu_D$ and $\sigma_D$ are the mean and the standard deviation of the active speakers' match scores, and $\eta$ is a control parameter. The larger $\eta$ is, the less speakers are pruned out, and vice versa. The formula (2) has the following interpretation. Assuming that the match scores follow a Gaussian distribution, the pruning threshold corresponds a certain *confidence interval* of the normal distribution, and $\eta$ specifies its width. For $\eta$ = 1, the speakers above the 68 % confidence interval of the match score distribution will be pruned out; that is approximately (100-68)/2 = 16 % of the speakers. This interpretation is illustrated in the right panel of Fig. 5. We have found out experimentally that the Gaussian assumption holds sufficiently well in practise. The left panel of Fig. 5 shows two real score distributions computed from two different subsets of the TIMIT corpus [39].

Notice that the mean and variance of the score distribution can be updated efficiently using the running values for these. Since the unlikely speakers (large scores) are pruned out



| 30 | 60 | 90 | 120 | 150 | 180 | 210 |
|---|---|---|---|---|---|---|
| 10.4 | 12.4 | 12.8 | 12.6 | 12.4 | 14.4 | **14.4** |
| 12.9 | 13.6 | 14.0 | 13.1 | 12.6 | 15.3 | |
| 14.7 | 16.0 | 15.9 | 14.9 | 14.5 | | |
| 14.5 | 18.1 | 18.0 | 17.6 | | | |
| 12.3 | 16.4 | 16.9 | 16.4 | | | |
| 16.8 | 18.8 | 17.8 | 16.9 | | | |
| 14.9 | 17.7 | 17.0 | 15.9 | | | |
| 16.7 | 18.7 | 18.2 | 16.7 | | | |
| 15.8 | 18.8 | 18.6 | 18.0 | | | |
| 13.8 | 14.9 | 16.5 | 15.6 | | | |
| 16.2 | 17.4 | 17.3 | 16.7 | | | |
| 17.3 | 18.4 | 19.2 | 17.7 | | | |
| 16.5 | 18.1 | 17.5 | 16.6 | | | |
| 14.9 | 16.3 | 16.9 | 16.0 | | | |
| 15.1 | 15.7 | 15.9 | 15.5 | | | |
| 18.0 | 20.9 | 20.5 | | | | |
| 21.6 | 25.6 | 25.7 | | | | |
| 15.1 | 18.0 | 19.3 | | | | |
| 16.9 | 20.0 | 19.9 | | | | |
| 18.3 | 19.2 | | | | | |

Fig. 6. Illustration of the confidence-based pruning.

iteratively, the variance of the match scores decreases with time. The control parameters of the method are $M$ and $\eta$.

### D. Confidence-Based Pruning (CP)

In confidence-based pruning, only speakers whose match scores have stabilized are considered for pruning. If the match score is poor but it still oscillates, the speaker can still change its rank and become the winner. Thus, we remove only speakers that have already stabilized and whose match score is below a given threshold. This is illustrated in Fig. 6, in which the speakers are at given one per line, and the time (vector count) increases from left to right. The numbers in the cells show the match scores, gray color indicates that the speaker has stabilized, and black indicates that the speaker has been pruned out. Notice that both the stabilization and pruning can happen in the same iteration.

The pseudocode of the method is given in Algorithm 4. Two score values are maintained for each active speaker $i$: the one from the previous iteration ($D_{prev}[i]$), and the one from the current iteration ($D_{curr}[i]$). When these two are close enough to each other, we mark the speaker as stabilized. Stabilized speakers are then checked against the pruning threshold as defined in (2). There are three adjustable parameters: the pruning interval ($M$), the stabilization threshold ($\epsilon$) and the pruning threshold control parameter ($\eta$).

### E. Combining PQ and Pruning (PQP)

Pre-quantization and pruning can be combined. Algorithm 5 combines clustering-based PQ and static pruning. First, the whole input data is pre-quantized by the LBG algorithm [31]. Using the match scores for the quantized data, $K$ worst scoring speakers are pruned out, and the final decision is based on comparing the unquantized data with the remaining speaker models. We refer the ratio of the number of pruned speakers to the number of all speakers as the *pruning rate*.

**Algorithm 4** Confidence-Based Pruning (CP)

```
$A := \{1, 2, \ldots, N\}$ ; $X := \emptyset$ ;
for $i := 1, \ldots, N$ do
    $D_{prev}[i] := 0$ ; stable[$i$] := false ;
end for
while ($|A| > 1$) and (speech data left) do
    Insert $M$ new vectors into buffer $X$ ;
    Update $D_{avg}(X, C_i)$ for all $i \in A$ ;
    Update pruning threshold $\Theta$ ;
    for $i \in A$ do
        $D_{curr}[i] := D_{avg}(X, C_i)$ ;
    end for
    for $i \in A$ do
        if ( $|1 - D_{prev}[i]/D_{curr}[i]| < \epsilon$ ) then
            stable[$i$] = true ;
        end if
        if (stable[$i$]) and ($D_{curr}(X, C_i) > \Theta$) then
            Prune out speaker $i$ from $A$ ;
        else
            $D_{prev}[i] := D_{curr}[i]$ ;
        end if
    end for
end while
Decision: $i^* = \arg\min_i \{D_{avg}(X, C_i) | i \in A\}$ ;
```

**Algorithm 5** PQ + Static Pruning (PQP)

```
$A := \{1, 2, \ldots, N\}$ ;
Read new data into buffer $X$ ;
$\hat{X} :=$ LBG-Clustering$(X, M)$
Compute $D_{avg}(\hat{X}, C_i)$ for all $i \in A$ ;
Prune out $K$ worst speakers from $A$ ;
Compute $D_{avg}(X, C_i)$ for all $i \in A$ ;
Decision: $i^* = \arg\min_i \{D_{avg}(X, C_i) | i \in A\}$ ;
```

## IV. Efficient Cohort Scoring for Verification

In this Section, we apply pre-quantization for speeding up the scoring in the verification task. Current state-of-the-art speaker verification systems use the Bayesian likelihood ratio [40] for normalizing the match scores [41], [42]. The purpose of the normalization is to reduce the effects of undesirable variation that arise from mismatch between the input and training utterances.

Given an identity claim that speaker $S$ produced the vectors $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, two likelihoods $p(X|S)$ and $p(X|\bar{S})$ are estimated. The former presents the likelihood that speaker $S$ produced $X$ (*null hypothesis*), and the latter presents the likelihood that $X$ was produced by someone else (*alternative hypothesis*). The two likelihoods are combined using the log-likelihood ratio [1]:

$$score(X, S) = \log p(X|S) - \log p(X|\bar{S}). \qquad (3)$$

This score is then compared with a predefined verification threshold. The speaker is accepted if the score exceeds the verification threshold, and otherwise rejected. We assume a common (global) threshold for all speakers.

The problem in the computation of (3) is that the likelihood of the alternative hypothesis is not directly accessible since this requires information of *all other speakers of the world*. There are two main approaches for the estimation of $p(X|\bar{S})$ [41]: *universal background model* (or *world model*) and *cohort set*.

**Algorithm 6** Fast Cohort Scoring (FCS)

```
Let $X$ be the unknown speaker's feature vectors ;
Let $C_S$ be the claimed speaker's codebook ;
Let $K > 1$ be the desired cohort size ;
$\hat{X} :=$ LBG-Clustering$(X, M)$ ;
Let $Coh := K$ best scoring speakers based on $D_{avg}(\hat{X}, C_i)$,
excluding the client ;
$score(X, S) = D_{avg}(\hat{X}, C_S) / \frac{1}{K} \sum_{i \in Coh} D_{avg}(\hat{X}, C_i)$ ;
```

The world model is generated from a large set of speakers, and it attempts to model speech in general. In the cohort approach, for each client speaker, an individual set of cohort speakers is defined. Usually the cohort set contains the nearest speakers to the client, since intuitively these are the "best" impostors to the client speaker. We are not aware of large-scale comparison of the world model and cohort approaches, and it seems that currently there is no consensus which one of these is more accurate.

Cohort normalization methods can be divided into two classes: those that select the cohort speakers *offline* in the training phase [43], and those that select the cohort *online* [44] based on the closeness to the test vector sequence $X$. The online approach, also known as *unconstrained cohort normalization* (UCN) [41], [44], has been observed to be more accurate [42], [44], probably due to its adaptive nature. Another desirable feature of the UCN is that it does not require updating of the cohort sets when new speakers are enrolled in the system.

The usefulness of the online cohort selection is limited by its computational complexity. The computation of the normalized score (3) includes searching the cohort speakers, whose time increases linearly with the number of cohort candidates. Ariyaeeinia and Sivakumaran [44] noticed that a smaller equal error rate (EER) is obtained, if the cohort is selected among the client speakers instead of using an external cohort set.

We propose to use pre-quantization for reducing the computational load of cohort search (see Algorithm 6). The input sequence $X$ is first quantized into a smaller set $\hat{X}$ using the LBG algorithm [31], and majority of the speakers are pruned out based on the scores $D_{avg}(\hat{X}, C_i), \; i = 1, \ldots, N$. The remaining set of $K > 1$ best scoring speakers constitutes the cohort for the client speaker. The client score is also computed using the quantized sequence, and the normalized match score is computed as the ratio between the client score and average cohort speaker score. A small value indicates that the client score deviates clearly from the impostor distribution. The control parameters of the algorithm are the cohort size ($K$) and the size of the quantized test set ($M$).

In acoustically mismatched conditions, both the client and cohort scores are expected to degrade, but their ratio is assumed to remains the same. This is the fundamental rationale behind score normalization. In other words, we assume:

$$\frac{D_{avg}(X, C_S)}{\sum_j D_{avg}(X, C_j)} \approx \frac{D_{avg}(\hat{X}, C_S)}{\sum_k D_{avg}(\hat{X}, C_k)}, \tag{4}$$

where $j$ and $k$ go over the indices of the cohort speakers selected using $X$ and $\hat{X}$, respectively. The approximation (4) is good when $X$ and $\hat{X}$ follow the same probability distribution.

TABLE I

SUMMARY OF THE CORPORA USED

| | TIMIT | NIST |
|---|---|---|
| Language | English | English |
| Speakers | 630 | 230 |
| Speech type | Read speech | Conversational |
| Quality | Clean (hi-fi) | Telephone |
| Sampling rate | 8.0 kHz | 8.0 kHz |
| Quantization | 16-bit linear | 8-bit $\mu$-law |
| Training speech (avg.) | 21.9 sec. | 119.0 sec. |
| Evaluation speech (avg.) | 8.9 sec. | 30.4 sec. |

## V. EXPERIMENTS

### A. Speech Material

For the experiments, we used two corpora, the *TIMIT* corpus [39] and the *NIST 1999 speaker recognition evaluation corpus* [45]. The TIMIT corpus was used for tuning the parameters of the algorithms, and the results were then validated using the NIST corpus.

Main features of the evaluated corpora are summarized in Table I. For consistency, the TIMIT files were downsampled from 16 to 8 kHz. This was preceded by alias cancellation using a digital low-pass FIR filter. TIMIT contains 10 files for each speaker, of which we selected 7 for training and 3 for testing. The files "sa" and "sx" having the same phonetic content for all speakers were included in the training material.

To our knowledge, no speaker identification experiments have been performed previously on the NIST-1999 corpus, and therefore, we needed to design the test setup ourselves. We selected to use the data from the male speakers only. Because we do not apply any channel compensation methods, we selected the training and recognition conditions to match closely. For training, we used both the "a" and "b" files for each speaker. For identification, we used the one speaker test segments from the same telephone line. In general it can be assumed that if two calls are from different lines, the handsets are different, and if they are from the same line, the handsets are the same [45]. In other words, the training and matching conditions have very likely the same handset type (electret/carbon button) for each speaker, but different speakers can have different handsets. The total number of test segments for this condition is 692.

### B. Feature Extraction, Modeling and Matching

We use the standard MFCCs as the features [29]. A pre-emphasiz filter $H(z) = 1 - 0.97z^{-1}$ is used before framing. Each frame is multiplied with a 30 ms Hamming window, shifted by 20 ms. From the windowed frame, FFT is computed, and the magnitude spectrum is filtered with a bank of 27 triangular filters spaced linearly on the mel-scale. The log-compressed filter outputs are converted into cepstral coefficients by DCT, and the $0^{\text{th}}$ cepstral coefficient is ignored. Speaker models are generated by the LBG clustering algorithm [31]. The quantization distortion (1) with Euclidean distance is used as the matching function.

### C. Performance Evaluation

The recognition accuracy of identification is measured by identification error rate, and the accuracy of the verification is measured by the equal error rate (EER). The methods were implemented using C/C++ languages. All experiments were carried out on a computing cluster of two Dell Optiplex G270 computers, each having 2.8 GHz processor and 1024 MB of memory. The operating system is Red Hat Linux release 9 with 2.4.22-openmosix2 kernel. We use system function `clock` divided by the constant `CLOCKS_PER_SEC` to measure the running time.

## VI. Results and Discussion

### A. Baseline System

First, a few preliminary tests were carried out on the TIMIT corpus in order to find out suitable silence detection threshold. The number of MFCCs and model sizes were fixed to 12 and 64, respectively. With the best silence threshold (lowest error rate), about 11-12 % of the frames were classified as silent and the average identification time improved by about 10 % as compared without silence detection. Recognition accuracy also improved slightly when silence detection was used (626/630 correct → 627/630 correct). Using the same silence detection threshold on the NIST, only 2.6 % of the frames were classified as silent, and there was no improvement in the identification time.

The effect of the number of MFCCs was studied next. Increasing the number of coefficients improved the identification accuracy up to 10-15 coefficients, after which the error rates stabilized. For the rest of the experiments, we fixed the number of coefficients to 12.

Table II summarizes the performance of the baseline system on the TIMIT corpus. The identification times are reported both for the full-search and for the VPT-indexed code vectors. The last row (no model) shows the results for using all training vectors directly as the speaker model as suggested in [46]. Increasing the model size improves the performance up to $M = 256$. After that, the results start to detoriate due to the overfitting effect, as observed also in [47]. The identification time increases with the codebook size. For small codebooks, VPT indexing does not have much effect on the identification times, but it becomes effective when $M \geq 32$. For the rest of the experiments, VPT indexing is used.

TABLE II
Performance of the baseline system (TIMIT).

| Codebook size | Error rate (%) | Avg. id. time (s) | |
|---|---|---|---|
| | | Full search | VPT |
| 8 | 10.5 | 0.29 | 0.33 |
| 16 | 2.22 | 0.57 | 0.62 |
| 32 | 0.63 | 1.15 | 1.11 |
| 64 | 0.48 | 2.37 | 2.07 |
| 128 | 0.16 | 4.82 | 4.14 |
| 256 | 0.16 | 10.2 | 8.21 |
| 512 | 0.32 | 21.6 | 12.9 |
| No model | 1.59 | 42.8 | 23.7 |

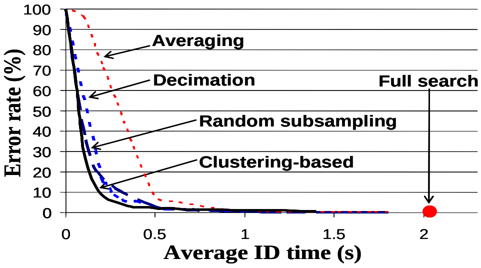


Fig. 7. Comparison of the PQ methods with codebook size 64 (TIMIT).

### B. Pre-Quantization

Next, we compare the pre-quantization methods with codebook size fixed to $M = 64$. Parameters were optimized with extensive testing for each PQ method separately. The best time-error curves for each method are shown in Fig. 7. We observe that the clustering PQ gives the best results, especially at the low-end when time is critical. In general, PQ can be used to reduce the time about to 50 % of the full search with a minor degradation in the accuracy.

### C. Speaker pruning

Next, we evaluate the performance of the speaker pruning variants with the pre-quantization turned off and speaker model size fixed to 64. Several experiments were carried out in order to find out the critical parameters. First, the variants were considered individually (see Figs 8 to 11).

For the SP algorithm, we fixed the pruning interval ($M = 5, 10, 15$ vectors) and varied the number of pruned speakers ($K$). The shortest pruning interval ($M = 5$) gives the poorest results and the largest interval ($M = 15$) the best. The difference between $M = 10$ and $M = 15$ is relatively small.

For the HP algorithm, we fixed the coarse speaker model size ($M = 4, 8, 16$) and varied the number of pruned speakers ($K$). We observe that the model sizes $M = 4$ and $M = 8$ give the best trade-off between the time and identification accuracy. If the codebook size is increased, more time is spent but the relative gain in accuracy is small.

For the AP algorithm, we fixed the parameter $\eta$ in (2) to $\eta = \{0.0, 0.1, 0.5, 0.9\}$ and varied the pruning interval ($M$). The values $\eta = 0.5$ and $\eta = 0.9$ give the best results.

For the CP algorithm, we fixed the two thresholds ($\epsilon = 0.1, 0.5$ ; $\eta = 0.1, 1.0$) and varied the pruning interval. The best result is obtained with combination $\eta = 1.0, \epsilon = 0.5$. The selection of the stabilization threshold $\epsilon$ seems to be less crucial than the pruning parameter $\eta$.

The pruning variants are compared in Fig. 12. The AP variant gives the best results, whereas the static pruning gives the poorest results. Next, we select the best PQ and pruning variants as well as the combination of PQ and pruning (PQP) as described in Section III-E and compare their performance. From the Fig. 13 we observe that the pruning approach gives slightly better results. However, in a time-critical application PQ might be slightly better. The combination of

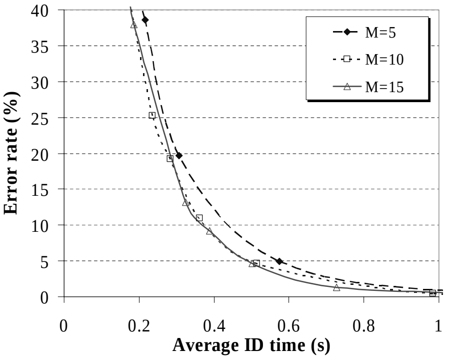


Fig. 8. Performance of the SP algorithm for different pruning intervals (TIMIT).

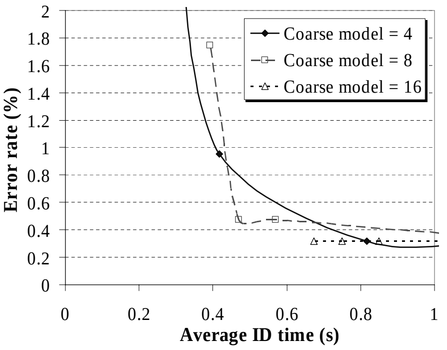


Fig. 9. Performance of the HP algorithm for different coarse model sizes with detail model size 64 (TIMIT).

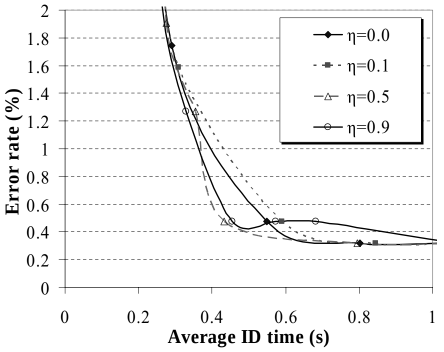


Fig. 10. Performance of the AP algorithm for different pruning thresholds (TIMIT).

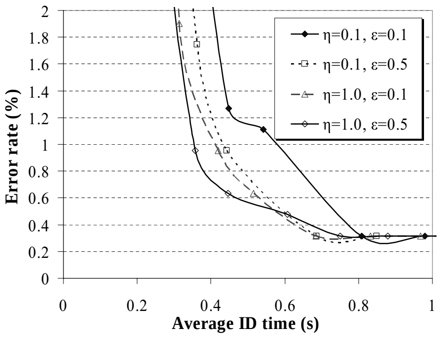


Fig. 11. Performance of the CP algorithm for different parameters (TIMIT).

pre-quantization and pruning (PQP) gives the best result as expected.

### *D. Validation with NIST and GMM*

Since TIMIT is known to give overly optimistic performance due to its laboratory quality and lack of intersession data, we validate the results on the NIST corpus. The best pre-quantization and pruning variants are also generalized to GMM modeling [14] as follows. Instead of using the log-likelihood $\log p(X|\text{GMM}_i)$ as score, we use $-\log p(X|\text{GMM}_i)$ instead. In this way, the scores are interpreted as dissimilarities, and the algorithms do not require any changes. We used diagonal covariance GMMs since they are widely used with the MFCC features, and they require significantly less computation and storage.

The best results for both corpora and model types are summarized in Tables III and IV. For pre-quantization, we use the clustering-based method, and for the pruning we use the adaptive variant. For the combination, we selected the clustering PQ and static pruning.

We optimized the model sizes for VQ and GMM separately. For VQ, larger codebook give more accurate results on both corpora as expected. GMM, on the other hand, is more sensitive to the selection of the model size. With TIMIT, model sizes larger than 64 degraded results dramatically (for model size 256 the error rate was 16.5 %). There is simply not enough training data for robust parameter estimation of the models. For NIST, there is 5 times more training data, and therefore large models can be used.

The problem of limited training data for GMM parameter estimation could be attacked by using, instead of the maximum likelihood (ML) training, the maximum a posteriori parameter (MAP) adaptation from the world model as described in [15]. Taking advantage of the relationship between the world model and the speaker-depended GMMs, it would also possible to reduce the matching time [15], [20]. In this paper, however, we restricted the study on the baseline ML method.

From the results of Tables III and IV we can make the following observations:

- Identification time depends on the size and the type of the model.

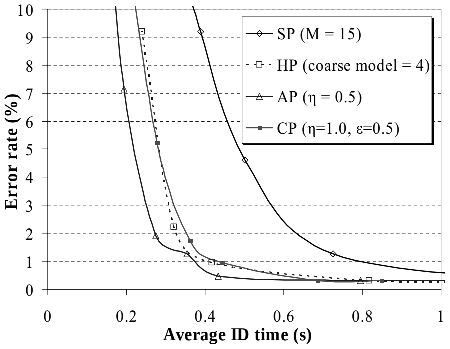


Fig. 12. Comparison of the pruning variants with speaker model size 64 (TIMIT).

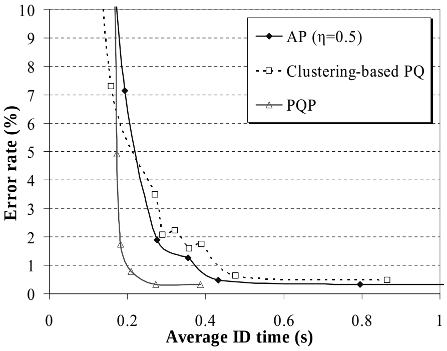


Fig. 13. Comparison of the best PQ and speaker pruning variants with speaker model size 64 (TIMIT).

- The error rates are approximately of the same order for both VQ and GMM. For TIMIT, the error rates are close to zero, and for NIST they are around 17-19 %.
- The speed-up factor of PQ increases with the model size as expected. Relative speed-up is higher for GMM than for VQ. Improvement of the pruning, on the other hand, depends much less on the model size.
- With TIMIT, PQP doubles the speed-up relative to PQ. With NIST, on the other hand, the PQP is not successful.
- The best speed-up factor for NIST with VQ is 16:1 increasing the error rate from 17.34 % to 18.20 %. For GMM, the corresponding speed-up factor is 34:1 with the increase of the error rate from 16.90 % to 18.50 %.

In general, we conclude that the results obtained with TIMIT hold also for NIST although there are differences between the corpora. More importantly, the studied algorithms generalize to GMM-based modeling. In fact, the speed-up factors are better for GMM than for VQ on the NIST corpus. The optimized systems are close to each other both in time and accuracy, and we cannot state that one of the models would be better than the other in terms of time/error trade-off. The ease of implementation, however, makes the VQ approach more attractive. In fact, prototype implementation for Symbian series 60 operating system for mobile devices is currently in progress.

The combination of PQ and GMM gives a good time-accuracy trade-off, which is consistent with the verification experiments carried out by McLaughlin & al. [18]. They noticed that the test sequence could be decimated up to factor 20:1 with minor effect on the verification performance. They found out that the fixed decimation (every $K$th vector) gave the best performance. However, as we can see from the Fig. 7, the clustering based pre-quantization performs better. This explains partially why we obtained a better speed-up (up to 34:1).

### *E. Fast Cohort Scoring for Verification*

The proposed cohort normalization method (FCS) was studied next on the NIST corpus. We used the same subset for verification than for the identification experiments, thus consisting of $N = 692$ genuine speaker trials and $N(N-1)/2 = 239086$ impostor trials. The speaker model size was set to 128 for both VQ and GMM based on the identification results, and the PQ codebook size for the FCS method was set to 32 after preliminary experiments. In both normalization methods, the client score is divided by the average cohort score. In the case of VQ, models are scored using the quantization distortion, and in the case of GMM, the log likelihood.

We consider the following methods:

- No normalization
- Closest impostors to the test sequence
- Fast cohort scoring (FCS)

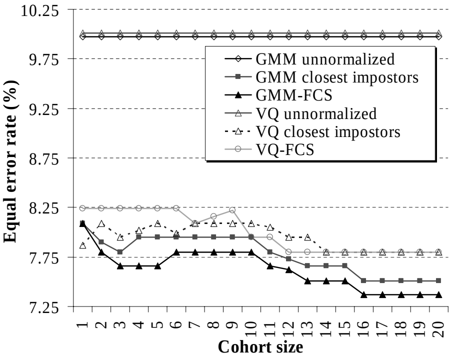


Fig. 14. Effect of the cohort size using different scoring methods (model sizes = 128; $M$ = 32) (NIST).

The cohort size is varied from $K = 1$ to $K = 20$. The equal error rates of the normalization methods are shown in Fig. 14, along with the unnormalized case as a reference. We observe an decreasing trend in EER with increasing cohort size for both normalization methods and for both modeling techniques. GMM gives better results for both normalization methods. More interestingly, the proposed method (FCS) out-

TABLE III

SUMMARY OF THE BEST RESULTS ON THE TIMIT CORPUS.

| | **Vector quantization (VQ)** | | | | **Gaussian mixture model (GMM)** | | | |
|---|---|---|---|---|---|---|---|---|
| Setup | Model size | Error rate (%) | Time (s) | Speed-up factor | Model size | Error rate (%) | Time (s) | Speed-up factor |
| Baseline | 64 | 0.32 | 2.07 | 1:1 | 8 | 0.95 | 0.93 | 1:1 |
| PQ | | 0.64 | 0.48 | 4:1 | | 0.95 | 0.49 | 2:1 |
| Pruning | | 0.48 | 0.43 | 5:1 | | 1.11 | 0.21 | 4:1 |
| PQP | | 0.32 | 0.27 | 8:1 | | 0.95 | 0.21 | 4:1 |
| Baseline | 128 | 0.00 | 4.14 | 1:1 | 16 | 0.16 | 1.77 | 1:1 |
| PQ | | 0.64 | 0.59 | 7:1 | | 0.48 | 0.77 | 2:1 |
| Pruning | | 0.00 | 1.88 | 2:1 | | 0.16 | 0.92 | 2:1 |
| PQP | | 0.00 | 0.31 | 13:1 | | 0.16 | 0.18 | 10:1 |
| Baseline | 256 | 0.00 | 8.21 | 1:1 | 32 | 0.32 | 3.47 | 1:1 |
| PQ | | 0.64 | 1.18 | 7:1 | | 0.32 | 0.72 | 5:1 |
| Pruning | | 0.00 | 3.28 | 3:1 | | 0.32 | 1.80 | 2:1 |
| PQP | | 0.00 | 0.65 | 13:1 | | 0.32 | 0.40 | 9:1 |

TABLE IV

SUMMARY OF THE BEST RESULTS ON THE NIST 1999 CORPUS.

| | **Vector quantization (VQ)** | | | | **Gaussian mixture model (GMM)** | | | |
|---|---|---|---|---|---|---|---|---|
| Setup | Model size | Error rate (%) | Time (s) | Speed-up factor | Model size | Error rate (%) | Time (s) | Speed-up factor |
| Baseline | 64 | 18.06 | 2.92 | 1:1 | 64 | 17.34 | 9.58 | 1:1 |
| PQ | | 18.20 | 0.62 | 5:1 | | 18.79 | 0.73 | 13:1 |
| Pruning | | 19.22 | 0.48 | 6:1 | | 19.36 | 0.82 | 12:1 |
| PQP | | 18.06 | 0.50 | 6:1 | | 17.34 | 0.94 | 10:1 |
| Baseline | 128 | 17.78 | 5.80 | 1:1 | 128 | 17.05 | 18.90 | 1:1 |
| PQ | | 18.93 | 0.64 | 9:1 | | 18.20 | 0.84 | 23:1 |
| Pruning | | 18.49 | 0.86 | 7:1 | | 17.34 | 2.88 | 7:1 |
| PQP | | 17.78 | 0.67 | 9:1 | | 17.63 | 1.34 | 14:1 |
| Baseline | 256 | 17.34 | 11.40 | 1:1 | 256 | 16.90 | 37.93 | 1:1 |
| PQ | | 18.20 | 0.70 | 16:1 | | 18.50 | 1.11 | 34:1 |
| Pruning | | 17.49 | 1.46 | 8:1 | | 17.48 | 5.78 | 7:1 |
| PQP | | 17.49 | 0.90 | 13:1 | | 18.06 | 2.34 | 16:1 |

performs the method of closest impostors even though only the quantized test sequence is used for scoring. This result supports the claim that redundancy in the test sequence should be removed. The result also indicates that the assumption (4) holds in practise.

Table V summarizes the performances of the two score normalization methods. The speed-up factor is relative to the closest impostors method. The proposed method speeds up the verification by a factor of 23:1 and it also decreases the error rate at the same time. The equal error rates are relatively high in general, which is because of a simple acoustic front-end. We did not apply either delta processing nor channel compensation methods such as cepstral mean subtraction.

TABLE V

SUMMARY OF THE COHORT SELECTION METHODS (COHORT SIZE = 20; MODEL SIZES = 128; $M$ = 32) (NIST).

| Method | Model | EER (%) | Avg. verif. time (s) | Speed-up factor |
|---|---|---|---|---|
| Closest impostors | VQ<br>GMM | 7.80<br>7.51 | 5.79<br>18.94 | 1:1<br>1:1 |
| FCS | VQ<br>GMM | 7.48<br>6.94 | 0.65<br>0.84 | 9:1<br>23:1 |

## VII. CONCLUSIONS

A real-time speaker identification system based on vector quantization (VQ) has been proposed. The most dominating factors of the identification time are the number of test vectors and the number of speakers. We used silence detection and pre-quantization for the reduction of the vectors, and speaker pruning for the reduction of the speakers. A VPT tree was applied for speeding up the nearest neighbor search from the speaker codebook.

We used the TIMIT corpus for tuning the parameters, and validated the results using the NIST-1999 speaker recognition evaluation corpus. With TIMIT, a speed-up factor of 13:1 was achieved without degradation in the identification accuracy. With NIST, a speed-up factor of 16:1 was achieved with a small degradation in the accuracy (17.34 % vs. 18.20 %).

We demonstrated that the methods formulated for VQ modeling generalize to GMM modeling. With TIMIT, a speed-up factor of 10:1 was achieved. With NIST, a speed-up factor of 34:1 was achieved with a small degradation (16.90 % vs. 18.50 %) in the accuracy.

We also applied pre-quantization for efficient cohort normalization in speaker verification. The proposed method turned out to be both faster and more accurate than the commonly used method of closest impostors. An EER of 6.94 % was reached in average verification time of 0.84 seconds when the length of test utterance is 30.4 seconds, with a speed-up of 23:1 compared to the widely used closest impostors method.

Regarding the selection between pre-quantization and pruning methods, the former seems more attractive in the light of the experimental results on the NIST corpus, and the findings reported in [18]. Clustering can be effectively applied for removing redundancy from the test sequence with small or no degradation in the accuracy. A possible future direction could be towards developing more adaptive pre-quantization methods (all pre-quantization methods studied here assume either fixed buffer or codebook size).

In this paper we restricted the study of the GMM to the baseline ML method. However, it is expected that the studied methods generalize to the UBM/GMM framework [15] and further speedups are possible by combining UBM/GMM with pre-quantization and speaker pruning. It is also possible to use UBM idea in the VQ context in the same way by generating a large speaker-independent codebook and adapting the speaker-dependent codebooks from it.

Finally, it must be noted that the acoustic front-end was fixed to MFCC processing in this study, and it seems that further speed optimization with these features is difficult. A possible future direction could be to use multiparametric classification: a rough estimate of the speaker class could be based on pitch features, and the matching could then be refined using spectral features. Alternatively, one could use initially high-dimensional features, such as a combination of cepstrum, delta-parameters, F0 features and voicing information, followed by a mapping into a low-dimensional space by linear discriminant analysis (LDA), principal component analysis (PCA), or neural networks. In this way, probably more discriminative low-dimensional features could be derived.

## Acknowledgments

The work of E. Karpov was supported by the National Technology Agency of Finland (TEKES) (project "Puhetekniikan uudet menetelmät ja sovellukset", TEKES dnro 8713103).

## References

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

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

[3] D.A. Reynolds. An overview of automatic speaker recognition technology. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2002)*, pages 4072–4075, Orlando, Florida, USA, 2002.

[4] A. Martin and M. Przybocki. Speaker recognition in a multi-speaker environment. In *Proc. 7th European Conference on Speech Communication and Technology (Eurospeech 2001)*, pages 787–790, Aalborg, Denmark, 2001.

[5] I. Lapidot, H. Guterman, and A. Cohen. Unsupervised speaker recognition based on competition between self-organizing maps. *IEEE Transactions on Neural Networks*, 13:877–887, 2002.

[6] D. Liu and F. Kubala. Fast speaker change detection for broadcast news transcription and indexing. In *Proc. 6th European Conference on Speech Communication and Technology (Eurospeech 1999)*, pages 1031–1034, Budapest, Hungary, 1999.

[7] S. Kwon and S. Narayanan. Speaker change detection using a new weighted distance measure. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)*, pages 2537–2540, Denver, Colorado, USA, 2002.

[8] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski. Rapid speaker adaptation in eigenvoice space. *IEEE Trans. on Speech and Audio Processing*, 8:695–707, 2000.

[9] X. He and Y. Zhao. Fast model selection based speaker adaptation for nonnative speech. *IEEE Trans. on Speech and Audio Processing*, 11(4):298–307, 2003.

[10] W. Jia and W.-Y. Chan. An experimental assessment of personal speech coding. *Speech Communications*, 30(1):1–8, 2000.

[11] A. Glaeser and F. Bimbot. Steps toward the integration of speaker recognition in real-world telecom applications. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)*, Sydney, Australia, 1998.

[12] H.S.M. Beigi, S.H. Maes, J.S. Sorensen, and U.V.Chaudhari. A hierarchical approach to large-scale speaker recognition. In *Proc. 6th European Conference on Speech Communication and Technology (Eurospeech 1999)*, pages 2203–2206, Budapest, Hungary, 1999.

[13] S. Furui. *Digital Speech Processing, Synthesis, and Recognition*. Marcel Dekker, Inc., New York, second edition, 2001.

[14] D.A. Reynolds and R.C. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Trans. on Speech and Audio Processing*, 3:72–83, 1995.

[15] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10(1):19–41, 2000.

[16] B. Sun, W. Liu, and Q. Zhong. Hierarchical speaker identification using speaker clustering. In *Proc. International Conference on Natural Language Processing and Knowledge Engineering 2003*, pages 299–304, Beijing, China, 2003.

[17] R. Auckenthaler and J.S. Mason. Gaussian selection applied to text-independent speaker verification. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)*, pages 83–88, Crete, Greece, 2001.

[18] 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 Conference on Speech Communication and Technology (Eurospeech 1999)*, pages 1215–1218, Budapest, Hungary, 1999.

[19] B.L. Pellom and J.H.L. Hansen. An efficient scoring algorithm for gaussian mixture model based speaker identification. *IEEE Signal Processing Letters*, 5(11):281–284, 1998.

[20] B. Xiang and T. Berger. Efficient text-independent speaker verification with structural gaussian mixture models and neural network. *IEEE Trans. on Speech and Audio Processing*, 11:447–456, September 2003.

[21] 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)*, pages 1353–1356, Denver, Colorado, USA, 2002.

[22] Z.Pan, K. Kotani, and T. Ohmi. An on-line hierarchical method of speaker identification for large population. In *NORSIG 2000*, Kolmården, Sweden, 2000.

[23] T. Kinnunen, E. Karpov, and P. Fränti. A speaker pruning algorithm for real-time speaker identification. In *Proc. Audio- and Video-Based Biometric Authentication (AVBPA 2003)*, pages 639–646, Guildford, UK, 2003.

[24] J. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40:175–230, 1991.

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

[26] J. He, L. Liu, and G. Palm. A discriminative training algorithm for VQ-based speaker identification. *IEEE Trans. on Speech and Audio Processing*, 7(3):353–356, 1999.

[27] T. Kinnunen and I. Kärkkäinen. Class-discriminative weighted distortion measure for VQ-based speaker identification. In *Proc. Joint IAPR International Workshop on Statistical Pattern Recognition (S+SPR2002)*, pages 681–688, Windsor, Canada, 2002.

[28] G. Singh, A. Panda, S. Bhattacharyya, and T. Srikanthan. Vector quantization techniques for GMM based speaker verification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2003)*, Hong Kong, 2003.

[29] J.R. Jr. Deller, J.H.L. Hansen, and J.G. Proakis. *Discrete-Time Processing of Speech Signals*. IEEE Press, New York, second edition, 2000.

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

[31] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, 1980.

[32] T. Kinnunen, T. Kilpeläinen, and P. Fränti. Comparison of clustering algorithms in speaker identification. In *Proc. IASTED Int. Conf. Signal Processing and Communications (SPC 2000)*, pages 222–227, Marbella, Spain, 2000.

[33] R.-H. Wang, L.-S. He, and H. Fujisaki. A weighted distance measure based on the fine structure of feature space: application to speaker recognition. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1990)*, pages 273–276, Albuquerque, New Mexico, USA, 1990.

[34] T. Matsui and S. Furui. A text-independent speaker recognition method robust against utterance variations. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1991)*, pages 377–380, Toronto, Canada, 1991.

[35] A.L. Higgins, L.G. Bahler, and J.E. Porter. Voice identification using nearest-neighbor distance measure. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1993)*, pages 375–378, Minneapolis, Minnesota, USA, 1993.

[36] T. Kinnunen and P. Fränti. Speaker discriminative weighting method for VQ-based speaker identification. In *Proc. Audio- and Video-Based Biometric Authentication (AVBPA 2001)*, pages 150–156, Halmstad, Sweden, 2001.

[37] N. Fan and J. Rosca. Enhanced VQ-based algorithms for speech independent speaker identification. In *Proc. Audio- and Video-Based Biometric Authentication (AVBPA 2003)*, pages 470–477, Guildford, UK, 2003.

[38] E. Chavez, G. Navarro, R. Baeza-Yates, and J.L. Marroquin. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, 2001.

[39] Linguistic data consortium. WWW page, September 2004. `http://www.ldc.upenn.edu/`.

[40] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, London, second edition, 1990.

[41] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas. Score normalization for text-independent speaker verification systems. *Digital Signal Processing*, 10:42–54, 2000.

[42] Y. Zigel and A. Cohen. On cohort selection for speaker verification. In *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, pages 2977–2980, Geneva, Switzerland, 2003.

[43] R.A. Finan, A.T. Sapeluk, and R.I. Damper. Impostor cohort selection for score normalization in speaker verification. *Pattern Recognition Letters*, 18:881–888, 1997.

[44] A.M. Ariyaeeinia and P. Sivakumaran. Analysis and comparison of score normalization methods for text dependent speaker verification. In *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech 1997)*, pages 1379–1382, Rhodes,Greece, 1997.

[45] A. Martin and M. Przybocki. The NIST 1999 speaker recognition evaluation - an overview. *Digital Signal Processing*, 10:1–18, 2000.

[46] D.R. Dersch and R.W King. Speaker models designed from complete data sets: a new approach to text-independent speaker verification. In *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech 1997)*, pages 2323–2326, Rhodos, Greece, 1997.

[47] R. Stapert and J.S.Mason. Speaker recognition and the acoustic speech space. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2001)*, pages 195–199, Crete, Greece, 2001.

PLACE PHOTO HERE

**Tomi Kinnunen** received his M.Sc. and Ph.Lic. degrees in computer science from the University of Joensuu, Finland, in 1999 and 2004, respectively. Currently he is a doctoral student in the same department, and his research topics include automatic speaker recognition and speech signal processing.

PLACE PHOTO HERE

**Evgeny Karpov** received his M.Sc in applied mathematics from Saint-Petersburg state University, Russia, in 2001, and M.Sc. in computer science from the University of Joensuu, Finland, in 2003. Currently he is a doctoral student in the same department, and his research topics include automatic speaker recognition and signal processing algorithms for mobile devices.

PLACE PHOTO HERE

**Pasi Fränti** received his M.Sc. and Ph.D. degrees in computer science from the University of Turku, Finland, in 1991 and 1994, respectively. From 1996 to 1999 he was a postdoctoral researcher with the University of Joensuu (funded by the Academy of Finland), where he has been a professor since 2000. His primary research interests are in image compression, pattern recognition and data mining.

6

## Publication P6

T. Kinnunen, I. Kärkkäinen, P. Fränti, The Mystery of Cohort Selection, *Report A-2005-1, Report series A*, University of Joensuu, Department of Computer Science (ISBN 952-458-676-2, ISSN 0789-7316).

# The Mystery of Cohort Selection

**Tomi Kinnunen, Ismo Kärkkäinen, Pasi Fränti**
Speech and Image Processing Unit, Department of Computer Science
University of Joensuu, Finland

**Abstract**

In speaker verification, *cohort* refers to a speaker-depended set of "anti-speakers" that are used in match score normalization. A large number of heuristic methods have been proposed for the selection of cohort models. In this paper, we use genetic algorithm (GA) for minimizing a cost function for a given security-convenience cost balance. The GA jointly optimizes the cohort sets and the global verification threshold. Our motivation is to use GA as an analysis tool. When comparing with heuristic selection methods, GA is used for obtaining a lower bound to error rates reachable by MFCC-GMM verification system. On the other hand, we analyze the models selected by GA, attempting to gain understanding into how cohort models should be selected for an application with given security-convenience tradeoff. Our findings with a subset of the NIST-1999 corpus suggest that in user-convenient application, the cohort models should be selected more close to the target than in secure application. The lower bounds in turn show that that there is a lot of room for further studies in score normalization, especially in the user-convenient end of the detection error tradeoff (DET) curve.

## 1 Introduction

*Speaker verification* [1] is the task of deciding whether a given speech utterance was produced by a claimed person (*target*). In biometric verification, two errors are possible: *false acceptance* (FA) and *false rejection* (FR). The former means accepting an impostor, and the latter refers to rejecting a genuine speaker. By adjusting the verification threshold, the system administrator can balance between the error types. By lowering the threshold, the number of false rejections can be reduced ("user-convenient" applications), but with the cost of increased number of false acceptances. By setting a high threshold, the number of false acceptances can be reduced ("secure" application).

In state-of-the-art verification systems, the features extracted from the unknown speaker's utterance are matched against the target and nontarget models. *Normalized score* [2, 3, 4, 5] is a function of the two scores, and it is compared with the verification threshold. The rationale is to make the match score relative to other models so that it is more robust against acoustic mismatches between training and recognition. Setting of speaker independent verification threshold becomes also easier because the scores are in common range.

The nontarget hypothesis represents the possibility that anyone else expect the target produced the unknown utterance. Thus, in principle the nontarget model should be composed of all possible speakers. Two popular approaches for approximating the nontarget likelihood are *world modeling* [5] and *cohort modeling* [6, 3, 7, 8, 9, 4, 10, 11], see Fig. 1. The world model, or *universal background model* (UBM), represents "the world of all possible speakers", and it is represented by a single model, which is same for all speakers. In the cohort approach, nontarget likelihood is approximated using a small number of speaker-depended "antispeakers", called the *cohort set* of the speaker.

The UBM normalization is straightforward and computationally efficient, but there are two motivations to study cohort selection more closely. Firstly, since the normalization depends on the speaker, it can change speaker rankings and could be also applied in the identification task (1:$N$ matching); the UBM normalization does not help in this because the match scores are scaled by the same number. The second motivation comes from the field of forensic speaker identification [12]. In forensic cases, the acoustic evidence must be contrasted against a relevant background population (e.g. speakers of same gender and dialectal region) to estimate the likelihood of a random match. Cohort selection could be applied to find the background population automatically from a database of several thousands of speakers.

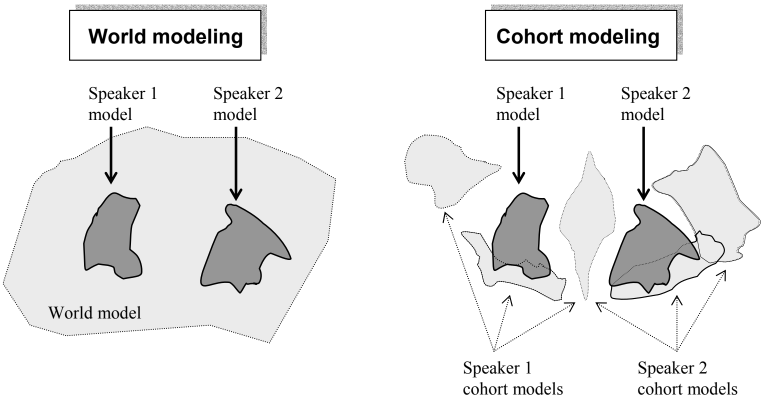


Figure 1: Illustration of the world and cohort modeling approaches.

In addition to the verification threshold, the selection of cohort models has influence on the accuracy. Traditionally, the balancing between FA/FR errors has been tackled by adjusting the verification threshold. However, the FA and FR errors are functions of both the score distributions *and* the verification threshold, and therefore, should be optimized jointly when setting up the verification system for a certain application.

Our goal is to gain some insight into the selection of the cohort models for a given secure-convenience balance. We approach the problem from two directions. Firstly, we give experimental comparison of existing cohort selection methods by comparing their performance at three different operating points. Secondly, we consider the cohort selection as a combinatorial optimization problem which we attack by a genetic algorithm. Both the cohort sets and the verification threshold are jointly optimized to minimize detection cost function (DCF). In this way, we can estimate a lower bound reachable by the acoustic features and model if the cohort models would be selected optimally. We also analyze the distances of the selected cohort models to the target speaker.

The rest of the paper is organized as follows. In Section 2 we review the background of GMM-based speaker verification. In Section 3 we define the optimization problem and formula the GA for solving it. Section 4 includes experiments and discussion. Finally, conclusions are drawn in Section 5.

# 2 Verification Background

## 2.1 GMM Speaker Modeling

The state-of-the-practise text-independent speaker model is the *Gaussian mixture model* (GMM) [13, 5]. GMM is well-suited for modeling of short-term spectral features like mel-frequency cepstral coefficients (MFCC) and linear predictive cepstral coefficients (LPCC) (see [14]), possibly appended with the corresponding dynamic features [15, 16].

A GMM of speaker $i$, denoted as $\mathcal{R}^{(i)}$, consists of a linear mixture of $K$ Gaussian components. Its density function is

$$p(\boldsymbol{x}|\mathcal{R}^{(i)}) = \sum_{k=1}^{K} P_k^{(i)} \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k^{(i)}, \boldsymbol{\Sigma}_k^{(i)}), \tag{1}$$

where $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k^{(i)}, \boldsymbol{\Sigma}_k^{(i)})$ denotes multivariate Gaussian density function with mean vector $\boldsymbol{\mu}_k^{(i)}$ and covariance matrix $\boldsymbol{\Sigma}_k^{(i)}$. $P_k^{(i)}$ are the component prior probabilities (mixing weights) and they are constrained by $P_k^{(i)} \geq 0$, $\sum_{k=1}^{K} P_k^{(i)} = 1$.

Assuming independent and identically distributed (i.i.d.) observations $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, the likelihood given a GMM $\mathcal{R}^{(i)}$ is

$$p(X|\mathcal{R}^{(i)}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|\mathcal{R}^{(i)}) = \prod_{t=1}^{T} \sum_{k=1}^{K} P_k^{(i)} \mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_k^{(i)}, \boldsymbol{\Sigma}_k^{(i)}), \tag{2}$$

and the log-likelihood is

$$\log p(X|\mathcal{R}^{(i)}) = \sum_{t=1}^{T} \log \sum_{k=1}^{K} P_k^{(i)} \mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_k^{(i)}, \boldsymbol{\Sigma}_k^{(i)}). \tag{3}$$

Usually GMM is trained with *maximum a posteriori adaptation* (MAP) from a *universal background model* (UBM) [5]. The UBM is a GMM trained from a large pool of different speakers and it is supposed to represent the distribution of speech parameters in general. In this way, the amount of training data can be small since the parameters are not estimated from scratch. A *relevance factor* parameter is used for balancing between the background model and the new data.

## 2.2 Bayesian Framework

In speaker verification, we are given an input sample $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, and an identity claim. The verification is defined as a two-class classification problem (or *hypothesis testing*) with the following possible decisions:

$$\begin{cases} \textit{Accept} \text{ identity claim, i.e. classify } X \rightarrow \text{Target} \\ \textit{Reject} \text{ identity claim, i.e. classify } X \rightarrow \text{Nontarget.} \end{cases}$$

We set nonnegative *decision costs* $C_{\text{FR}}$ and $C_{\text{FA}}$ for the FA and FR error types. As an example, for a high security system, we might set $C_{\text{FR}} = 1$ and $C_{\text{FA}} = 10$, i.e. accepting an impostor is ten times more costly than rejecting a true speaker. According to *Bayes' rule for minimum risk classification* [17], speaker is accepted if

$$\frac{p(X|\text{Target})}{p(X|\text{Nontarget})} \geq \frac{P(\text{Nontarget})}{P(\text{Target})} \cdot \frac{C_{\text{FA}}}{C_{\text{FR}}}, \tag{4}$$

where $p(X|\cdot)$ are the likelihoods and $P(\cdot)$ are the prior probabilities. Notice that the right hand side of (4) does not depend on $X$, and therefore, decision rule is of the form $l(X) \geq \Theta$, where

$$l(X) = \frac{p(X|\text{Target})}{p(X|\text{Nontarget})} \tag{5}$$

is the *likelihood ratio* and $\Theta$ is the verification threshold. Equivalently, for the log likelihood ratio, we accept speaker if

$$\log p(X|\text{Target}) - \log p(X|\text{Nontarget}) \geq \log \Theta. \tag{6}$$

The likelihood ratio concept is intuitively easy to understand: when the evidence in favor of the target hypothesis is large while the evidence for the nontarget hypothesis is small, we are confident that the speaker is the one who he claims to be. On the other hand, when $l(X) \ll 1$, we are confident that the speaker is not the claimed one, and the case $l(X) = 1$ corresponds to the most uncertain case ("no decision").

The likelihood ratio $l(X)$ is called *normalized score* as it is a relative score computed by normalizing the target score by the nontarget score. Score normalization is expected to reduce the acoustic mismatch between training and testing. When the acoustic conditions change, both the target and nontarget scores change but the relative score is expected to remain unchanged [18]. The same idea can be applied to other than likelihood scores. In addition to cohort and world modeling approaches, the scores can be normalized using impostor score distribution mean and variance [2, 4]. Some of the various background normalization methods have been compared experimentally in [19, 20, 10, 21].

### 2.3 World and Cohort Normalization

In the world modeling (UBM) approach, nontarget likelihood is computed using a single world model $p(X|\mathcal{R}^{\mathrm{UBM}})$. Thus, the log likelihood ratio for speaker $i$ is simply

$$\log l(X) = \log p(X|\mathcal{R}^{(i)}) - \log p(X|\mathcal{R}^{\mathrm{UBM}}). \tag{7}$$

In the cohort approach, each speaker has a set of personal cohort[1] models which we index by $\mathcal{C}_i$. In addition to the target likelihood $p(X|\mathcal{R}^{(i)})$, we have the cohort likelihoods $p(X|\mathcal{R}^{(j)})$, where $j \in \mathcal{C}_i$. The nontarget likelihood can be approximated by applying geometric mean [7], arithmetic mean [3] or maximum [18] to the cohort likelihoods. For cohort size $M = |\mathcal{C}_i|$, the log likelihood ratios for these are given respectively by

$$\begin{aligned}
\log l(X) &= \log p(X|\mathcal{R}^{(i)}) - \frac{1}{M}\sum_{j\in\mathcal{C}_i} \log p(X|\mathcal{R}^{(j)}) && (8)\\
\log l(X) &= \log p(X|\mathcal{R}^{(i)}) - \log \frac{1}{M}\sum_{j\in\mathcal{C}_i} p(X|\mathcal{R}^{(j)}) && (9)\\
\log l(X) &= \log p(X|\mathcal{R}^{(i)}) - \max_{j\in\mathcal{C}_i} \log p(X|\mathcal{R}^{(j)}). && (10)
\end{aligned}$$

Different normalization approaches have been proposed e.g. in [22, 23, 24].

The world model approach is more popular because of the following reasons. Firstly, in the MAP adaptation [5], the world model is needed anyway, so it integrates into the GMM framework naturally without extra storage requirements.

[1] According to *Oxford English Dictionary*, cohort was a body of infantry in the Roman army, of which there were ten in a legion, each consisting of from 300 to 600 men. In demography, cohort refers to a group of persons having a common statistical characteristic, for instance, being born in the same year.

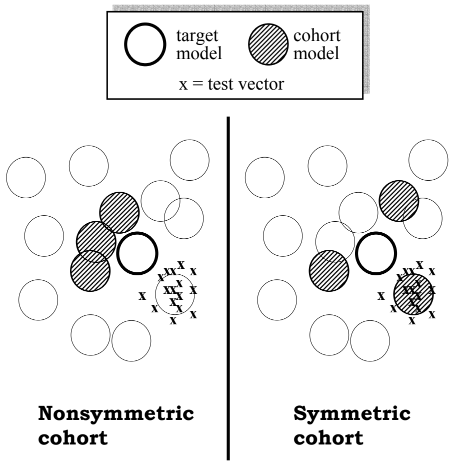


Figure 2: Problem of redundant cohort models.

Secondly, there is no ambiguity in defining the normalized score, whereas the cohort approach requires selection of the cohort speakers and fixing both the normalization formula and the cohort size. However, the cohort approach is intuitively reasonable, and because of the flexibility, it is potentially more accurate.

### 2.4 Cohort Selection

A large number of cohort selection methods have been proposed [6, 3, 7, 8, 25, 9, 4, 26, 10, 11]. Closest speakers to the target are the most competitive ones, and they are good candidates for the cohort speakers. This approach [6, 25, 8, 27, 4, 21] is the most commonly used one, and will referred here to as the *closest impostors* (CI) method. One problem with this approach is that it prepares for impostor attacks only against "similar" speakers. However, if the impostor is dissimilar (e.g. another gender), the data will be in the tails of both target and nontarget distributions, giving rise to poorly estimated likelihood ratio [28]. Thus, the cohort should include models both from close and far from the target [3].

If the cohort size is small, selection of redundant models should be avoided, see Fig. 2 for an illustration. Approaches presented in [3, 10] prevent adding redundant models into the cohorts. In both studies, initial cohort candidate set is first constructed, and the final cohort set is obtained by pruning out similar models [3] or by clustering them [10].

Cohort speakers are usually selected in the training phase because of computational reasons. *Unconstrained cohort selection* (UCN) that selects the competing models based on the test utterance likelihood is proposed in [8]. This method is computationally expensive, but it can be made more efficient by clustering the test sequence [11]. Usually cohort sets are composed of *full* speaker models; an alternative approach has been proposed in [9, 29], in which the impostor model is built from the individual Gaussian components of different speakers.

In the model selection algorithms, a similarity or distance measure between two GMMs is needed. Rosenberg *et al.* [6] propose the following similarity measure:

$$s(\mathcal{R}^{(i)}, \mathcal{R}^{(j)}) = \frac{1}{2}\Big\{ \log p(X_i|\mathcal{R}^{(j)}) + \log p(X_j|\mathcal{R}^{(i)}) \Big\}, \tag{11}$$

where $X_i$ and $X_j$ are the training data used for constructing the models $\mathcal{R}_i$ and $\mathcal{R}_j$, respectively. Reynolds [3] proposes the following divergence-like dissimilarity measure:

$$d(\mathcal{R}^{(i)}, \mathcal{R}^{(j)}) = \log \frac{p(X_i|\mathcal{R}^{(i)})}{p(X_i|\mathcal{R}^{(j)})} + \log \frac{p(X_j|\mathcal{R}^{(j)})}{p(X_j|\mathcal{R}^{(i)})}. \tag{12}$$

# 3 Optimization Framework

We assume that the speaker models $\mathcal{R}^{(i)}, i = 1, \ldots, N$ have already been trained. In general, these can be other than GMMs since we operate on the score space. All cohort sets are denoted collectively as $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_N)$. We consider each speaker's model $\mathcal{R}_i$ and the cohort models $\{\mathcal{R}^{(j)}|j \in \mathcal{C}_i\}$ together as a one model, called the *compound model.* The compound model for speaker $i$ is denoted as $\mathcal{M}^{(i)} = (\mathcal{R}^{(i)}, \{\mathcal{R}^{(j)}|j \in \mathcal{C}_i\})$, and we will denote the normalized match score as $s(X, \mathcal{M}^{(i)})$. The task is to optimize the compound models $\mathcal{M}^{(i)}$ from the existing single models so that a cost function is minimized. In a sense, cohort selection can be seen as *discriminative training* of speaker models.

## 3.1 False Acceptance and Rejection

The match score $s(X, \mathcal{M}^{(i)}) \in \mathbb{R}$ is a continuous random variable with an unknown probability distribution $p(s)$ which can be divided into genuine and impostor distributions $p(s|\text{genuine}), p(s|\text{impostor})$, see upper panel of Fig. 3. These represent the distributions obtained by matching a random utterance $X$ against genuine speaker model (the speaker who actually produced $X$) and someone else's model, respectively.

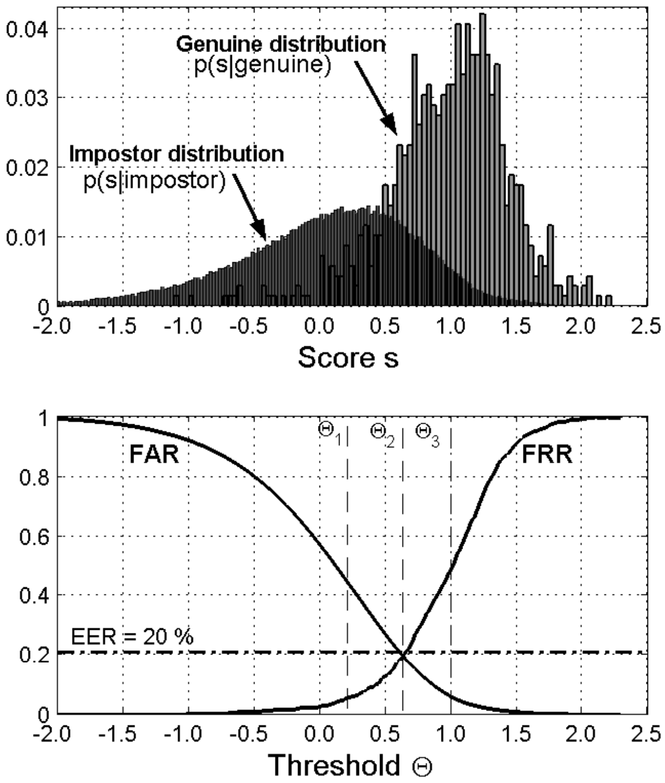


Figure 3: Increasing $\Theta$ decreases false acceptances and increases false rejections.

The true distributions $p(s|\text{target}), p(s|\text{nontarget})$ are not available, so we need to estimate them empirically. For this, we use a labeled development set $\mathcal{Z} = \{(X_j, Y_j) | j = 1, 2, \ldots, L\}$, including at least one segment per speaker ($L \geq N$). Here, $X_j$'s are the test segments, and $Y_j$'s are the correct class labels ($Y_j \in \{1, \ldots, N\}$).

First, we define the error counts $\mathrm{FR}_i$ and $\mathrm{FA}_i$ for each speaker $i$ as follows:

$$\begin{aligned} \mathrm{FR}_i &= \sum_{j=1}^{L} \mathcal{I}\{Y_j = i \wedge s(X_j, \mathcal{M}_i) < \Theta\} \quad (13) \\ \mathrm{FA}_i &= \sum_{j=1}^{L} \mathcal{I}\{Y_j \neq i \wedge s(X_j, \mathcal{M}_i) \geq \Theta\}, \quad (14) \end{aligned}$$

where $\mathcal{I}\{A\} = 1$, if proposition $A$ is true and 0 otherwise. False rejection rate (FRR) and false acceptance rate (FAR) can now be calculated as

$$\mathrm{FRR}(\mathcal{C}, \Theta) = \frac{1}{N \cdot L} \sum_{i=1}^{N} \mathrm{FR}(\mathcal{C}_i, \Theta) \quad (15)$$

$$\mathrm{FAR}(\mathcal{C}, \Theta) = \frac{1}{N \cdot L} \sum_{i=1}^{N} \mathrm{FA}(\mathcal{C}_i, \Theta), \quad (16)$$

where we used the notation to emphasize their dependence on both the cohort sets and the verification threshold $\Theta$. Because the errors depend on both, they should be jointly optimized.

By keeping the cohort sets fixed and sweeping the verification threshold over the real line, we can calculate FRR and FAR at every threshold. By plotting FRR as a function of FAR, we get a curve that shows the trade-off between the two error types. On the other hand, by varying the cohort sets, we get different score distributions. Again, we get a new error trade-off curve by sweeping the threshold over the real line. Each point at each curve corresponds to a certain $(\mathcal{C}, \Theta)$ pair, and the error values $\mathrm{FRR}(\mathcal{C}, \Theta)$, $\mathrm{FAR}(\mathcal{C}, \Theta)$ for this pair are known. The optimization task can be formulated as finding the pair $(\mathcal{C}, \Theta)$ for which an objective function depending on FRR and FAR is minimized.

## 3.2 Detection Cost Function

Decreased FAR implies increased FRR, and vice versa. In most applications, either one of the error types can be considered more costly than the other one. Following the detection cost function (DCF) defined by NIST [30], we define the optimization problem as finding $(\mathcal{C}, \Theta)$ for which the weighted sum of errors is

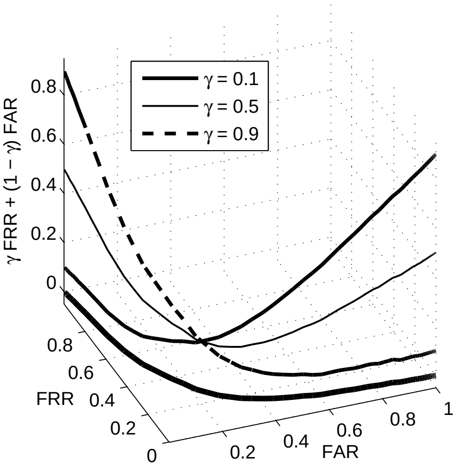


Figure 4: Illustration of the cost function along with the error tradeoff curve in the $xy$-plane.

minimized:

$$\min_{(\mathcal{C},\Theta)} \Big\{ \gamma \cdot \mathrm{FRR}(\mathcal{C},\Theta) + (1-\gamma) \cdot \mathrm{FAR}(\mathcal{C},\Theta) \Big\}, \tag{17}$$

where $0 < \gamma < 1$ is a design parameter controlling the tradeoff between the errors. An illustration of the cost function is shown in Fig. 4.

Since the cohort sets $\mathcal{C}_i$ do not depend on each other, the cost function can be written as a sum of cost functions over different speakers:

$$\min_{(\mathcal{C},\Theta)} \sum_{i=1}^{N} \Big\{ \gamma \cdot \mathrm{FR}(\mathcal{C}_i,\Theta) + (1-\gamma) \cdot \mathrm{FA}(\mathcal{C}_i,\Theta) \Big\} \tag{18}$$

We can separate $\mathcal{C}$ and $\Theta$ by defining the optimal threshold $\Theta^*(\mathcal{C})$ for a given $\mathcal{C}$ as

$$\Theta^*(\mathcal{C}) = \arg\min_{\Theta} \sum_{i=1}^{N} \Big\{ \gamma \cdot \mathrm{FR}(\mathcal{C}_i,\Theta) + (1-\gamma) \cdot \mathrm{FA}(\mathcal{C}_i,\Theta) \Big\}, \tag{19}$$

which can be found by linear search by sweeping $\Theta$ over the genuine and impostor score distributions. The optimization problem becomes

$$\min_{\mathcal{C}} \sum_{i=1}^{N} \Big\{ \gamma \cdot \mathrm{FR}\Big(\mathcal{C}_i, \Theta^*(\mathcal{C})\Big) + (1-\gamma) \cdot \mathrm{FA}\Big(\mathcal{C}_i, \Theta^*(\mathcal{C})\Big) \Big\}. \tag{20}$$

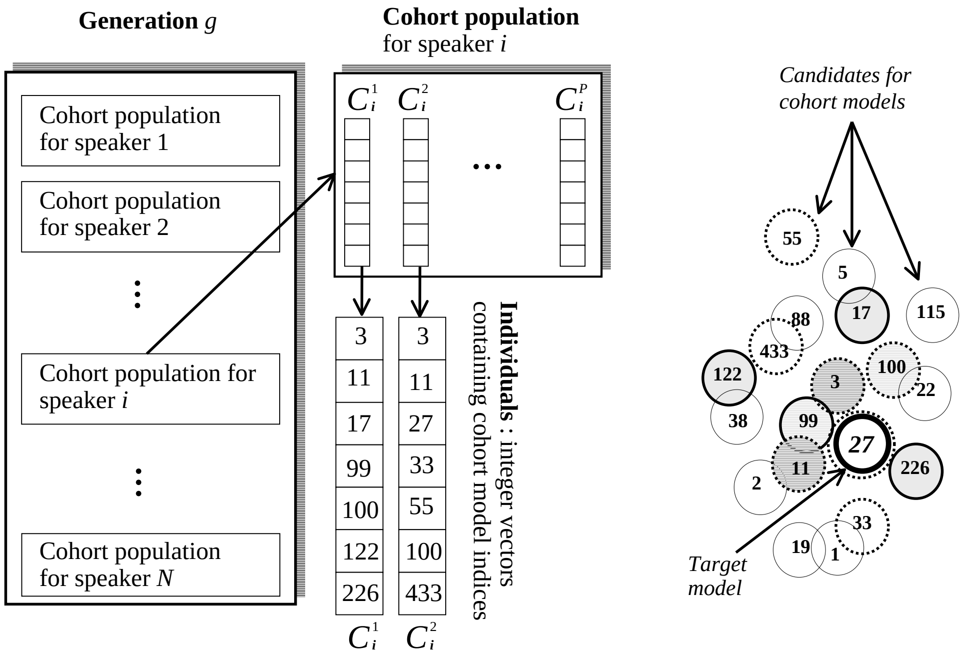

Generation g
Cohort population for speaker 1
Cohort population for speaker 2
Cohort population for speaker i
Cohort population for speaker N
Cohort population for speaker i
$C_i^1$ $C_i^2$ $C_i^P$
3 11 17 99 100 122 226
3 11 27 33 55 100 433
$C_i^1$ $C_i^2$
Individuals : integer vectors containing cohort model indices
Candidates for cohort models
55
5
17
115
88
433
122
3
100
22
38
99
27
226
2
11
33
19
1
Target model

Figure 5: Basic data structures in the GA-based cohort optimization.

### 3.3 Genetic Algorithm for Minimizing DCF

Brute force optimization requires evaluating an exponential number of cohort sets and is out of question. We use a *genetic algorithm* (GA) [31] to minimize DCF. We maintain a separate population for each speaker, see Fig. 5 for the data structures. Individuals are integer vectors of dimensionality $M$ (cohort size). The $j$th individual for speaker $i$ is denoted as $\mathcal{C}_i^j$.

Pseudocode for the GA is given in Algorithm 1. Initialization is done by selecting $M$ disjoint random integers as the individuals. New candidates are generated using crossover and mutation operators, which doubles the sizes of the cohort populations. Next, we compute the normalized match scores using a labeled tuning set $\mathcal{Z}$.

Since computation of the fitness values $\text{DCF}(\mathcal{C}_i^j, \Theta)$ requires the common threshold, we must pool together all genuine and impostor trial scores over all speakers and cohorts. In practise, we use histograms for reducing the number op-

**Algorithm 1** Outline of the GA-based cohort optimization.

```
$\mathcal{P} \leftarrow$ InitializePopulations() ;
for $g = 1, 2, \ldots,$NumGenerations do
  $\mathcal{P}_{\text{cand}} \leftarrow$ GenerateNewCandidates($\mathcal{P}$) ;
  (G,I) $\leftarrow$ ComputeNormalizedScores($\mathcal{R}, \mathcal{P} \cup \mathcal{P}_{\text{cand}}, \mathcal{Z}$);
  $\Theta_{\text{opt}} \leftarrow$ ComputeOptimalThreshold(G,I,$C_{\text{FA}}, C_{\text{FR}}$) ;
  $\mathcal{F} \leftarrow$ ComputeDCFValues(G,I,$\Theta_{\text{opt}}$) ;
  $(\mathcal{P}, \mathcal{F}) \leftarrow$ SelectSurvivors($\mathcal{P} \cup \mathcal{P}_{\text{cand}}, \mathcal{F}$) ;
end for
return $(\mathcal{P}, \Theta_{\text{opt}})$ ;
```

erating points before pooling. As a result, we have the genuine and impostor trial score distributions (G, I). Using these, we find the optimal threshold as (19). After the threshold $\Theta^*(\mathcal{C})$ is found, the fitness values are calculated as $\text{DCF}(\mathcal{C}_i^j, \Theta)$.

New candidates are generated by pairing the vectors randomly and performing crossover. The parents and the offspring are pooled, and for the pooled population, every vector is mutated with a probability $P_m$. Crossover is implemented by duplicating the parent vectors into the offspring vectors and swapping their elements with probability $P_c$. In mutation, we replace a randomly selected index by a random number.

For selection, we sort the vectors according to their fitness (DCF) values. The best individual (smallest DCF) is always selected to the next generation. For the remaining ones, we compare successive pairs, and select the better one. The worst individual dies out.

# 4 Experiments

## 4.1 Corpus, Feature Extraction, and Modeling

For the experiments, we use the male subset of *NIST 1999 Speaker Recognition Evaluation* corpus [32]. Both the “a” and “b” files are used for training the 64 component diagonal covariance GMMs, whereas the 1-speaker male test segments are used as the tuning set $\mathcal{Z}$ for the cohorts.

In the current implementation, we use a simple MFCC front end without

Table 1: Summary of the corpus.

| | |
|---|---|
| Language | English |
| Speakers | 207 |
| Speech type | Conversational |
| Quality | Telephone |
| Sampling rate | 8.0 kHz |
| Quantization | 8-bit $\mu$-law |
| Training speech (avg.) | 119.0 sec. |
| Evaluation speech (avg.) | 30.4 sec. |

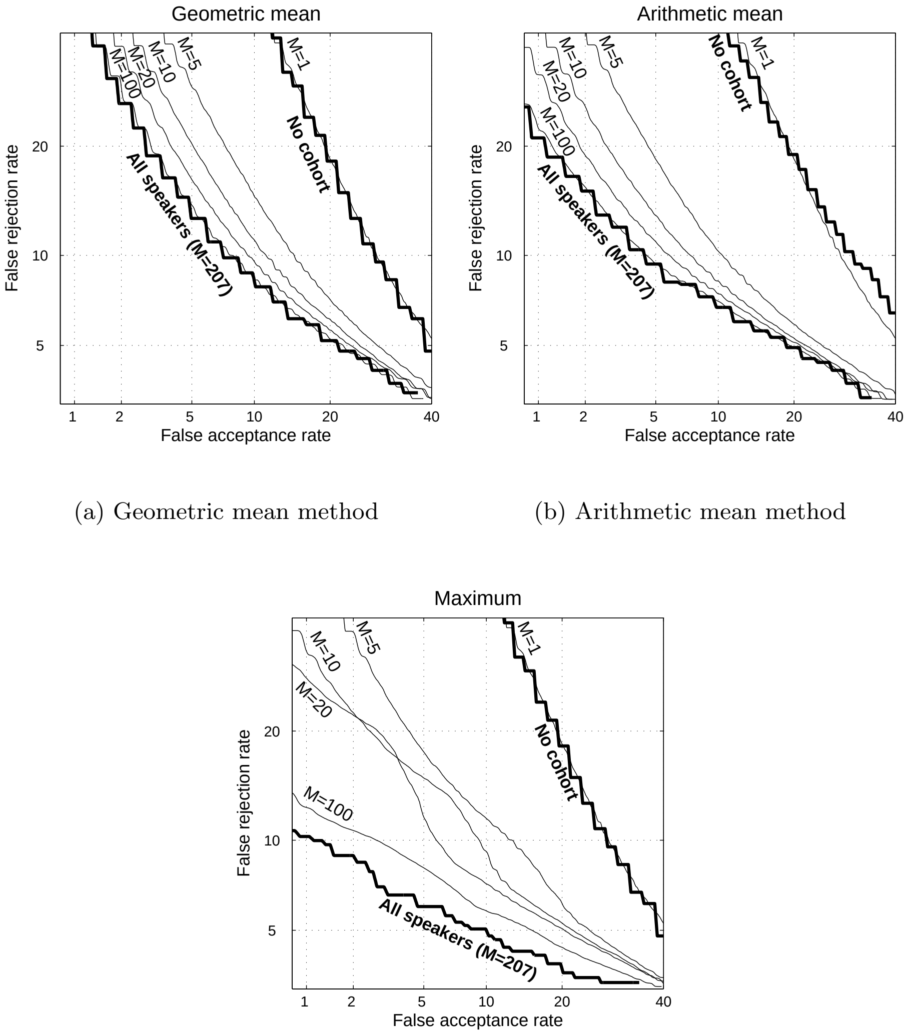
channel normalization, so we decided to restrict the experiments to matched telephone lines case. There are 230 male speakers in total, and from these 207 fulfill the matched telephone line case.

The UBM is trained by using all the two-speaker detection task files from the same corpus, including both males and females. From this, speaker-depended GMMs are derived by adapting the mean vectors using the MAP procedure [5]. MFCC features are computed from Hamming-windowed and pre-emphasized 30 ms frame with 10 ms overlap. We retain the 12 lowest MFCC coefficinets (excluding $c_0$) from the log-compressed 27-channel filterbank outputs using DCT.

Throughout the experiments, we consider three operating points corresponding to the following application scenario:

- Secure scenario (low FAR)
- 50-50 scenario (low EER)
- User-convenient (low FRR)

For the secure scenario, we require false acceptance rate to be at most 3 %, and compare the obtained FRRs for different approaches. Similarly, for the user-convenient scenario, we require the FRR to be at most 3 % and compare the obtained FARs.



(a) Geometric mean method

(b) Arithmetic mean method

(c) Maximum method

Figure 6: The effect of the normalization formula and cohort size (randomly selected cohorts, averaged DET curves for 100 repetitions).

## 4.2 Normalization Formula

First, we study the behavior of the normalization formulae (8)-(10), with the focus on their robustness. For this, we select the cohort models randomly and

Table 2: Standard deviations of errors using random cohort (100 repetitions).

| | **Secure**<br>FRR @ FAR = 3 % | | | **50-50**<br>EER | | | **User convenient**<br>FAR @ FRR = 3 % | | |
|---|---|---|---|---|---|---|---|---|---|
| Cohort size | 5 | 10 | 20 | 5 | 10 | 20 | 5 | 10 | 20 |
| Geometric mean | 3.9 | 3.1 | 2.9 | 1.0 | 0.8 | 0.8 | 9.1 | 8.8 | 8.7 |
| Arithmetic mean | 3.8 | 2.4 | 1.7 | 0.9 | 0.6 | 0.7 | 9.6 | 8.2 | 8.3 |
| Maximum | 3.6 | 3.0 | 4.6 | 1.8 | 1.0 | 0.6 | 10.0 | 9.9 | 9.3 |

repeat the procedure 100 times. In this way, we get an idea about the average performance and variance. The average detection error tradeoff (DET) curves [33] for the three normalization methods are shown in Fig. 6 for different cohort sizes. For comparative purposes, we also show the baseline (no score normalization) and the case where all speakers are included in the cohort. Table 2 shows the standard deviations for the three application scenarios and cohort sizes $M = 5, 10, 20$.

We observe that increasing the cohort size improves accuracy for all methods, except for cohort size $M = 1$, for which the baseline gives similar or better results. However, the performance increases rapidly with increasing cohort size in both “secure” and “user-convenient” ends of the curve for all three methods. Increased cohort size reduces also variance, which is due to the fact that larger cohorts include more and more tge same models as the models are selected among the targets.

Regarding the three methods, the ordering is consistent: geometric mean performs the worst and maximum the best on average. However, the variance of the arithmetic mean is smallest, and thus it is expected to be most robust. Because of larger variance, we expect that the geometric mean and maximum methods require more careful selection of the cohort.

Geometric mean and maximum operators are in a sense opposites to each other. Geometric mean gives high nontarget score if the test data yields high likelihood for *all* cohort models (“AND” operator). In contrast, maximum method indicates high nontarget score if there is a single cohort model that has high likelihood (“OR” operator). The arithmetic mean is in between the two extremes, and all the three formulae are special cases of *generalized mean* [34].

Even though performance increases with the cohort size, it must be remem-

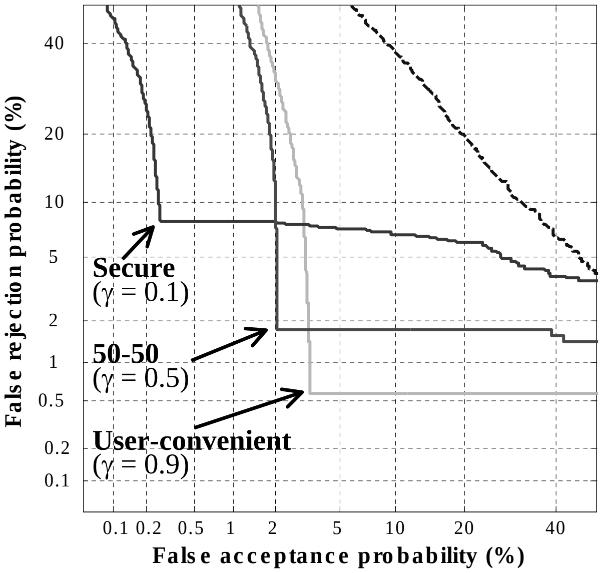


Figure 7: Examples of DET curves obtained by GA (arithmetic mean, cohort size $M = 5$).

bered that large cohort size implies a large number of likelihood calculations and it becomes computationally unfeasible. For this reason, we are interested in smaller cohort sizes.

Table 3: Verification thresholds optimized by GA (log likelihood ratio domain).

| | **Secure**<br>$\gamma = 0.1$ | | | **50-50**<br>$\gamma = 0.5$ | | | **User convenient**<br>$\gamma = 0.9$ | | |
|---|---|---|---|---|---|---|---|---|---|
| Cohort size | 5 | 10 | 20 | 5 | 10 | 20 | 5 | 10 | 20 |
| Geometric mean | 1.37 | 1.39 | 1.4 | 0.89 | 0.95 | 0.97 | 0.27 | 0.37 | 0.42 |
| Arithmetic mean | 1.09 | 1.11 | 1.11 | 0.73 | 0.75 | 0.77 | 0.12 | 0.19 | 0.21 |
| Maximum | 0.56 | 0.41 | 0.27 | 0.25 | 0.00 | 0.00 | -0.35 | -0.50 | -0.64 |

Table 4: Results for geometric mean normalization.

| | **Secure** | | | **50-50** | | | **User-convenient** | | |
|---|---|---|---|---|---|---|---|---|---|
| | FRR @ FAR = 3 % | | | EER | | | FAR @ FRR = 3 % | | |
| Baseline | 69.4 | | | 20.2 | | | 56.1 | | |
| UBM | 17.2 | | | 8.4 | | | 45.8 | | |
| Cohort size | 5 | 10 | 20 | 5 | 10 | 20 | 5 | 10 | 20 |
| Random | 38.6 | 29.6 | 24.3 | 12.1 | 10.5 | 9.7 | 45.9 | 43.2 | 43.5 |
| CI | 20.8 | 16.7 | 14.8 | 9.7 | 8.1 | 7.7 | 41.6 | 31.6 | 39.5 |
| MSC | 20.7 | 16.6 | 14.5 | 9.2 | 8.3 | 7.9 | 42.6 | 36.6 | 35.5 |
| MSCF | 34.7 | 32.1 | 27.2 | 12.1 | 11.0 | 10.3 | 49.3 | 52.7 | 50.3 |
| UCN | 60.9 | 55.4 | 47.8 | 17.6 | 15.8 | 14.6 | 52.7 | 50.2 | 44.7 |
| GA reference | 3.7 | 2.6 | 4.3 | 3.1 | 2.8 | 3.1 | 13.4 | 5.0 | 19.1 |

Table 5: Results for arithmetic mean normalization.

| | **Secure** | | | **50-50** | | | **User-convenient** | | |
|---|---|---|---|---|---|---|---|---|---|
| | FRR @ FAR = 3 % | | | EER | | | FAR @ FRR = 3 % | | |
| Baseline | 69.4 | | | 20.2 | | | 56.1 | | |
| UBM | 17.2 | | | 8.4 | | | 45.8 | | |
| Cohort size | 5 | 10 | 20 | 5 | 10 | 20 | 5 | 10 | 20 |
| Random | 27.3 | 18.5 | 14.8 | 10.1 | 8.9 | 8.3 | 44.2 | 41.9 | 40.6 |
| CI | 17.5 | 13.6 | 11.3 | 8.8 | 7.8 | 7.4 | 40.8 | 36.4 | 40.1 |
| MSC | 15.1 | 11.4 | 10.2 | 8.1 | 7.9 | 7.2 | 41.1 | 35.4 | 32.8 |
| MSCF | 18.4 | 13.2 | 11.1 | 9.2 | 8.0 | 7.9 | 43.2 | 48.2 | 49.3 |
| UCN | 56.1 | 48.8 | 39.5 | 15.9 | 14.3 | 12.7 | 51.1 | 49.0 | 48.7 |
| GA reference | 3.9 | 2.6 | 4.0 | 3.1 | 2.7 | 4.0 | 12.0 | 2.7 | 30.2 |

### 4.3 Selection Algorithms

Next, we compare the following heuristic approaches:

| | |
|---|---|
| Random | Random cohort |
| CI | Closest impostors selected using (12) |
| MSC | Maximally spread close [3] |
| MSCF | Maximally spread close + far [3] |
| UCN | Unconstrained cohort normalization [8] |

Genetic algorithm is optimized for the test data, and its purpose is to provide a lower bound to the error rates reachable by MFCC/GMM combination. It

Table 6: Results for maximum normalization.

| | **Secure**<br>FRR @ FAR = 3 % | | | **50-50**<br>EER | | | **User-convenient**<br>FAR @ FRR = 3 % | | |
|---|---|---|---|---|---|---|---|---|---|
| Baseline | 69.4 | | | 20.2 | | | 56.1 | | |
| UBM | 17.2 | | | 8.4 | | | 45.8 | | |
| Cohort size | 5 | 10 | 20 | 5 | 10 | 20 | 5 | 10 | 20 |
| Random | 24.7 | 18.4 | 19.9 | 10.9 | 9.0 | 7.9 | 44.9 | 43.3 | 44.1 |
| CI | 13.9 | 11.7 | 10.4 | 9.2 | 8.3 | 7.7 | 42.8 | 40.8 | 49.4 |
| MSC | 13.8 | 11.8 | 10.8 | 8.9 | 8.6 | 7.9 | 40.5 | 51.5 | 49.5 |
| MSCF | 19.4 | 14.1 | 11.7 | 9.9 | 8.8 | 8.6 | 42.2 | 50.5 | 58.4 |
| UCN | 50.4 | 39.6 | 29.0 | 14.5 | 14.0 | 11.3 | 51.2 | 46.4 | 48.0 |
| GA reference | 2.8 | 2.0 | 3.6 | 2.9 | 2.2 | 3.6 | 2.9 | 5.3 | 24.8 |

presents an “oracle selection” scheme - the oracle knows exactly what the targets are going to say during verification trial and selects the optimal cohorts for future.

GA finds a single operating point from the error tradeoff curve and is suboptimal in the other regions, see Fig. 7. Examples of thresholds optimized found by GA are listed in Table 3. It can be observed that the threshold increases when moving towards secure applications, which is expected.

The “corner” points in Fig. 7 are the minimum cost function operating points. We set $\gamma = 0.1$, $\gamma = 0.5$, and $\gamma = 0.9$ for the secure, 50-50, and the user-convenient scenarios, respectively. After preliminary experimentation, we fixed the GA parameters as follows: population size 100, the number of generations 500, mutation probability 0.01, and crossover probability 0.5.

The results for the three normalization methods are given in Tables 4-6. The results for baseline (no score normalization) and the UBM [5] are also shown as a reference. Several observations can be made. Firstly, arithmetic mean and maximum are more accurate than geometric mean. Secondly, comparing the heuristic methods, CI, MSC and MSCF are similar in performance, whereas UCN is worse. Thirdly, comparing the cohort and UBM approaches, UBM outperforms random cohort, MSCF and UCN in most cases, whereas CI and MSC outperform UBM.

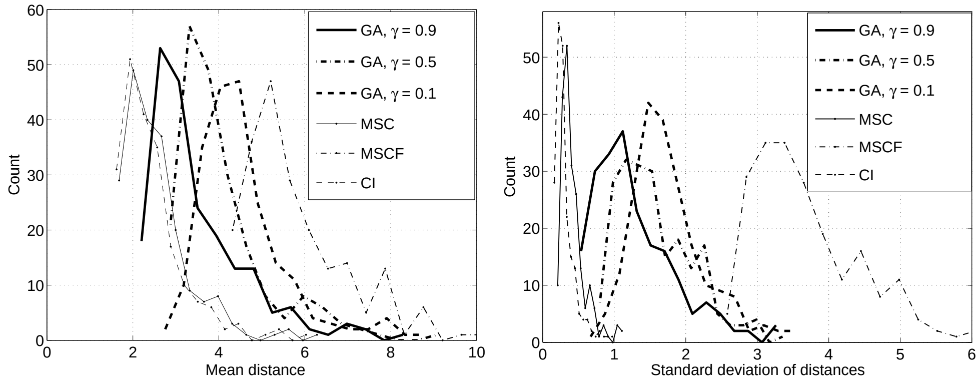
Some interesting observations can be made regarding the application scenario and UBM versus cohort approaches. In the 50-50 case, the differences are small between the methods. However, in the secure and user-convenient scenario, the cohort approach clearly outperforms UBM. In the secure end, UBM reaches an FRR of 17.2 %, whereas the best heuristic cohort selection method reaches 10.2 % (MSC with arithmetic mean, cohort size 20). In the user-convenient end, UBM reaches a FAR of 45.8 %, whereas the best heuristic cohort method reaches 31.6 % (CI with geometric mean, cohort size 10). These observations stress the importance of comparing methods using not only on the EER operating point which is an arbitrary choice.

The reference performance given by GA shows that there is much room to improve cohort selection algorithms. In particular, all the studied methods are poor at the user-convenient end. The GA suggests that it would be possible to reach a FAR of 2.7 % at FRR = 3.0 % if the cohorts were selected optimally. The best heuristic reaches as poor as 31.6 % FAR, an order of magnitude worse than GA suggests. Notice however that for GA, increased cohort size reduces the performance, which is contradictory to the results for the heuristic methods. A possible explanation for this is that the parameter space is larger for increased cohort size and GA might not have converged yet. We did not make further attempts in optimizing the number of generations as the simulations take rather long time.

## 4.4 Analysis of Selected Cohorts

Next, we analyze the cohort sets selected by the genetic algorithm, with the hope to gain understanding on the selection procedure. The GA was optimized for the test data, and now we are interested to see if optimal selection could be predicted from the training conditions only. We use the distance (12) for analyzing the model proximities. We also experimented with the similarity measure (11), and the results were similar.

The distribution of means and standard deviations of the distances from the target to his cohort models are shown in Fig. 8 for the arithmetic mean method and cohort size $M = 20$. The CI, MSC and MSCF are also shown for comparison. We make the following observations. Regarding the distribution of means, the models selected using CI and MSC are closer to target models than for other methods as expected. The models selected using MSCF are further away, and the GA selected models in between. The order of the standard deviations is the same, and holds for all the three application scenarios. These observations suggest that the optimal cohort should contain not "too close" or "too far" models but something in between. Similarly, the optimal cohort should not be too concentrated or too spread but something in between.



(a) Means

(b) Standard deviations

Figure 8: Distributions of mean and standard deviation of cohort model distances from the target.

Table 7: Number of cases (%) where speaker belongs to his own cohort

| | **Secure**<br>$\gamma = 0.1$ | | | **50-50**<br>$\gamma = 0.5$ | | | **User convenient**<br>$\gamma = 0.9$ | | |
|---|---|---|---|---|---|---|---|---|---|
| Cohort size | 5 | 10 | 20 | 5 | 10 | 20 | 5 | 10 | 20 |
| Geometric mean | 11.0 | 20.0 | 24.0 | 25.0 | 38.0 | 46.0 | 86.0 | 74.0 | 74.0 |
| Arithmetic mean | 19.0 | 33.0 | 48.0 | 49.0 | 66.0 | 76.0 | 93.0 | 95.0 | 95.0 |
| Maximum | 0.00 | 0.00 | 0.48 | 0.00 | 99.0 | 99.5 | 95.0 | 95.0 | 97.0 |

According to Fig. 8, in user-convenient scenario, the cohort models should be selected closer to the target than in the secure scenario. Table 7 gives further evidence of this by showing the the number of cases, in which speaker belongs to his own cohort. We observe that in the user-convenient scenario, speaker belongs to his own cohort in 74 % - 97 % of the cases, and the number decreases when moving towards the secure end.

This result might seem counterintuitive at the first glance. In a user-convenient application, it is important that the correct speaker is not rejected; thus, it seems logical to assume that competing models should not be located “too close” to the target. However, by including close models to the cohort, the denominator of the LR will be accurately presented when a genuine speaker is present (likelihood of $X$ for both target and cohorts is accurately computed). In the extreme case of cohort size $M = 1$ and speaker in his own cohort, LR for a genuine speaker will be always close to 1 and the threshold is set easily around this value by GA (see Table 3).

By excluding the target from his cohort in the secure scenario, the score for a genuine speaker will be in general larger, which has the effect of shifting the genuine distribution right. On the other hand, (casual) impostor data is far away from the target model in general, and it does not matter if the target is included in the denominator or not - the impostor data will far away from the target model and not be affected by it much. Thus, the impostor distribution will be relatively unchanged regardless of whether target is or is not included in the cohort. Because the genuine distribution shifts up, the distributions will be better separated.

In conclusion, the effect of including target in his own cohort in a user-convenient application makes the genuine distribution centered around LR = 1, and setting of threshold is easier. In the secure application, leaving the speaker out from the cohort has the effect of shifting genuine distribution right while retaining impostor distribution relatively unchanged.

## 5 Discussion and Conclusions

We have presented a step towards non-heuristic cohort selection based on minimizing a detection cost function. We find the following observations the most interesting ones:

1. UBM and cohort approaches perform similar in 50-50 and user-convenient scenario, whereas cohort is clearly better in the secure scenario.

2. There is lots of room for studying score normalization, especially in the

user-convenient end of the DET curve. The results of GA suggest that the MFCC features can reach both low FAR and FRR if the cohorts are well-selected.

The experiments suggest the following design rules for the cohort normalization approach:

1. Randomly selected cohort is better than no cohort. In this case, the cohort size should be as large as possible.

2. In general, larger cohort is better because it reduces the variance of the nontarget scores.

3. Arithmetic mean normalization is most robust and consistent over different selection methods, and we recommend to use it by default.

4. Maximum normalization has the best potential according to the GA reference, but the difference with the arithmetic mean is not large.

5. Of the heuristic methods compared, CI and MSC are both good choices.

6. In a user-convenient and 50-50 applications, it is advantageous to include nearby models into the cohort. In particular, the speaker's own model.

From a practical point of view, we must ask how useful the cohort normalization is in real applications. Sometimes cohort approach is criticized for its computational complexity and memory requirements, which is true if cohort size is large or the cohort models are selected from an external population. However, the results of GA suggest that good cohorts can be selected among the other registrants; in this case, we need to store only the lookup tables for the cohort indices in addition to the models. The results also suggest that small error rates could be reached if we knew how to select the cohorts; the methodology in this study presents an "oracle selection" scheme where the oracle knows exactly what the targets are going to utter during verification trial and selects good cohorts.

We have used GA here merely as an analysis tool. However, it might be used also as a practical cohort selection method. We believe in its potential, because it jointly optimizes the cohort sets and the verification threshold; usually these two are designed independent from each other, although FAR and FRR errors depend on both of them.

To apply GA as a practical cohort selection method, there are two principal issues that need to be studied. Firstly, as seen from Fig. 7, the algorithm optimizes a single point on the tradeoff curve. However, from the system administrator's perspective, it would be good to have the whole tradeoff curve optimized, from which the desired optimal threshold can be selected. For this, the objective function should be modified to minimize the total area under the DET curve for example. The second challenge relates to computational complexity: the simulations made in this study were time- and memory-consuming.

Finally, we wish to emphasize that the optimization was carried out entirely in the score space by having fixed acoustic features and models. The result of the optimization is a set of indices that merely tells against which models the features are to be matched during the verification process. Similar optimization can be carried out for any biometric authentication problem, in which severe mismatches are expected between training and testing.

## References

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

[2] K.-P. Li and J.E. Porter. Normalizations and selection of speech segments for speaker recognition scoring. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1988)*, pages 595–598, New York, 1988.

[3] D.A. Reynolds. Speaker identification and verification using gaussian mixture speaker models. *Speech Communication*, 17:91–108, 1995.

[4] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas. Score normalization for text-independent speaker verification systems. *Digital Signal Processing*, 10:42–54, 2000.

[5] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10(1):19–41, 2000.

[6] A.E. Rosenberg, J. DeLong, C.-H. Lee, B.-H. Juang, and F.K. Soong. The use of cohort normalized scores for speaker recognition. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1992)*, pages 599–602, Banff, Canada, October 1992.

[7] C.-S. Liu, H.-C. Wang, and C.-H. Lee. Speaker verification using normalized log-likelihood score. *IEEE Trans. on Speech and Audio Processing*, 4(1):56–60, 1996.

[8] A.M. Ariyaeeinia and P. Sivakumaran. Analysis and comparison of score normalization methods for text dependent speaker verification. In *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech 1997)*, pages 1379–1382, Rhodes,Greece, 1997.

[9] T. Isobe and J. Takahashi. Text-independent speaker verification using virtual speaker based cohort normalization. In *Proc. 6th European Conference on Speech Communication and Technology (Eurospeech 1999)*, pages 987–990, Budapest, Hungary, 1999.

[10] Y. Zigel and A. Cohen. On cohort selection for speaker verification. In *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, pages 2977–2980, Geneva, Switzerland, 2003.

[11] T. Kinnunen, E. Karpov, and P. Fränti. Efficient online cohort selection method for speaker verification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2004)*, volume 3, pages 2401–2402, Jeju Island, Korea, 2004.

[12] P. Rose. *Forensic Speaker Identification.* Taylor & Francis, London, 2002.

[13] D.A. Reynolds and R.C. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Trans. on Speech and Audio Processing*, 3:72–83, 1995.

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

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

[16] F.K. Soong and A.E. Rosenberg. On the use of instantaneous and transitional spectral information in speaker recognition. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 36(6):871–879, 1988.

[17] R. Duda, P. Hart, and D. Stork. *Pattern Classification.* Wiley Interscience, New York, second edition, 2000.

[18] A. Higgins, L. Bahler, and J. Porter. Speaker verification using randomized phrase prompting. *Digital Signal Processing*, 1:89–106, 1991.

[19] D.A. Reynolds. Comparison of background normalization methods for text-independent speaker verification. In *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech 1997)*, volume 2, pages 963–966, Rhodes,Greece, 1997.

[20] D. Tran and M. Wagner. Fuzzy C-means clustering-based speaker verification. In *Proc. Advances in Soft Computing (AFSS 2002)*, pages 318–324, Calcutta, India, February 2002.

[21] P. Sivakumaran, J. Fortuna, and A.M. Ariyaeeinia. Score normalization applied to open-set, text-independent speaker identification. In *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, pages 2669–2672, Geneva, Switzerland, 2003.

[22] L.F. Lamel and J.L. Gauvain. Speaker verification over the telephone. *Speech Communication*, 31:141–154, 2000.

[23] D. Tran and M. Wagner. Noise clustering-based speaker verification. In *Proc. Advances in Soft Computing (AFSS 2002)*, pages 325–331, Calcutta, India, February 2002.

[24] K.P. Markov and S. Nakagawa. Text-independent speaker recognition using non-linear frame likelihood transformation. *Speech Communication*, 24:193–209, 1998.

[25] R.A. Finan, A.T. Sapeluk, and R.I. Damper. Impostor cohort selection for score normalization in speaker verification. *Pattern Recognition Letters*, 18:881–888, 1997.

[26] N. Mirghafori and L. Heck. An adaptive speaker verification system with speaker dependent a priori decision thresholds. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 2002)*, pages 589–592, Denver, Colorado, USA, 2002.

[27] T. Pham and M. Wagner. Fuzzy-integration based normalization for speaker verification. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP 1998)*, pages 3273–3276, Sydney, Australia, 1998.

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

[29] T. Isobe and J. Takahashi. A new cohort normalization using local acoustic information for speaker verification. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 1999)*, volume 2, pages 841–844, Phoenix, Arizona, USA, 1999.

[30] M. Przybocki and A. Martin. NIST speaker recognition evaluation chronicles. In *Proc. Speaker Odyssey: the Speaker Recognition Workshop (Odyssey 2004)*, pages 15–22, Toledo, Spain, 2004.

[31] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer Verlag, Berlin, 3rd revised and extended edition edition, 1996.

[32] A. Martin and M. Przybocki. The NIST 1999 speaker recognition evaluation - an overview. *Digital Signal Processing*, 10:1–18, 2000.

[33] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech 1997)*, pages 1895–1898, Rhodes,Greece, 1997.

[34] L.I.Kuncheva. *Fuzzy Classifier Design.* Physica Verlag, Heidelberg, 2000.

# Dissertations at the Department of Computer Science

**Rask, Raimo.** Automating Estimation of Software Size during the Requirements Specification Phase - Application of Albrecth's Function Point Analysis Within Structured Methods. Joensuun yliopiston luonnontieteellisiä julkaisuja, 28 - University of Joensuu. Publications in Sciences, 28. 128 p. Joensuu, 1992.

**Ahonen, Jarmo.** Modeling Physical Domains for Knowledge Based Systems. Joensuun yliopiston luonnontieteellisiä julkaisuja, 33 - University of Joensuu. Publications in Sciences, 33. 127 p. Joensuu, 1995.

**Kopponen, Marja.** CAI in CS. University of Joensuu, Computer Science, Dissertations 1. 97 p. Joensuu, 1997.

**Forsell, Martti.** Implementation of Instruction-Level and Thread-Level Parallelism in Computers. University of Joensuu, Computer Science, Dissertations 2. 121 p. Joensuu, 1997.

**Juvaste, Simo.** Modeling Parallel Shared Memory Computations. University of Joensuu, Computer Science, Dissertations 3. 190 p. Joensuu, 1998.

**Ageenko, Eugene.** Context-based Compression of Binary Images. University of Joensuu, Computer Science, Dissertations 4. 111 p. Joensuu, 2000.

**Tukiainen, Markku.** Developing a New Model of Spreadsheet Calculations: A Goals and Plans Approach. University of Joensuu, Computer Science, Dissertations 5. 151 p. Joensuu, 2001.

**Eriksson-Bique, Stephen.** An Algebraic Theory of Multidimensional Arrays. University of Joensuu, Computer Science, Dissertations 6. 278 p. Joensuu, 2002.

**Kolesnikov, Alexander.** Efficient Algorithms for Vectorization and Polygonal Approximation. University of Joensuu, Computer Science, Dissertations 7. 204 p. Joensuu, 2003.

**Kopylov, Pavel.** Processing and Compression of Raster Map Images. University of Joensuu, Computer Science, Dissertations 8. 132 p. Joensuu, 2004.

**Virmajoki, Olli.** Pairwise Nearest Neighbor Method Revisited. University of Joensuu, Computer Science, Dissertations 9. 164 p. Joensuu, 2004.

**Suhonen, Jarkko.** A Formative Development Method for Digital Learning Environments in Sparse Learning Communities, Dissertations 10. 154 p. Joensuu, 2005.

**Xu, Mantao.** K-means Based Clustering and Context Quantization, Dissertations 11. 162 p. Joensuu, 2005.

**Kinnunen, Tomi.** Optimizing Spectral Feature Based Text-Independent Speaker Recognition, Dissertations 12. 156 p. Joensuu, 2005.

Julkaisija<br>
Publisher

Joensuun yliopisto<br>
Tietojenkäsittelytieteen laitos<br>
University of Joensuu<br>
Department of Computer Science

Vaihdot

Joensuun yliopiston kirjasto / Vaihdot<br>
PL 107, 80101 Joensuu<br>
Puh. 013-251 2677, fax 013-251 2691<br>
e-mail: `vaihdot@joensuu.fi`

Exchanges

Joensuu University Library / Exchanges<br>
P.O. Box 107, FI-80101 Joensuu, FINLAND<br>
Tel. +358-13-251 2677, fax +358-13-251 2691<br>
e-mail: `vaihdot@joensuu.fi`

Myynti

Joensuun yliopiston kirjasto / Julkaisujen myynti<br>
PL 107, 80101 Joensuu<br>
Puh. 013-251 4509, fax 013-251 2691<br>
e-mail: `joepub@joensuu.fi`

Sales

Joensuu University Library / Sales of Publications<br>
P.O. Box 107, FI-80101 Joensuu, FINLAND<br>
Tel. +358-13-251 4509, fax +358-13-251 2691<br>
e-mail: `joepub@joensuu.fi`