Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Fraunhofer
IAIS

Fachbereich Informatik
Computer Science Department

Master Thesis

in the Master's Program
Master of Science in Computer Science

Workflow for Automatic i-Vector based Speaker Identification on German Parliament Speakers

by
Gunnar Åkermark

Primary supervisor:	Prof. Dr. Gerhard K. Kraetzschmar
Secondary supervisor:	Prof. Dr. Paul Plöger
External supervisor:	Dr. Daniel Stein

Handed in on: 04.02.2016

Eidesstattliche Erklärung

Ich erkläre an Eides Statt, dass ich meine Masterarbeit mit dem Titel *Workflow for Automatic i-Vector based Speaker Identification on German Parliament Speakers* selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe und dass ich alle Stellen, die ich wörtlich oder sinngemäß aus Veröffentlichungen entnommen habe, als solche kenntlich gemacht habe. Die Arbeit hat bisher in gleicher oder ähnlicher Form oder auszugsweise noch keiner Prüfungsbehörde vorgelegen.

Ich versichere, dass die eingereichte schriftliche Fassung der auf dem beigefügten Medium gespeicherten Fassung entspricht.

Sankt Augustin, den 04.02.2016

Gunnar Akermark

Contents

Glossary

AS audio segmentation

ASR automatic speech recognition

BIC Bayesian information criterion

DCT discrete cosine transformation

DET detection error tradeoff

DFT discrete Fourier transform

EER equal error rate

EFR eigen factor radial

EM expectation maximization

FA forced alignment

GMM Gaussian mixture model

HMM hidden Markov model

IAIS Institute for Intelligent Analysis and Information Systems

JFA joint factor analysis

LDA linear discriminant analysis

MAP maximum a posteriori

MFCC mel frequency cepstral coefficient

PDF probability density function

PLDA probabilistic linear discriminant analysis

PLP perceptual linear prediction

ROC receiver operating characteristic

SD speech detection

SID speaker identification

SN spherical normalization

TV total variability

UBM universal background model

WCCN within-class covariance normalization

List of Figures

Chapter 1

Introduction

One of the biggest challenges humanity has to face in the information age is how to deal with the enormous growth of information. Between 2010 and 2011, 48 hours of content were uploaded to Youtube every minute. Between 2011 and 2013, it were already 72 hours, and looking at the time span from 2013 to 2015, we have reached 300 hours per minute [Jam15]. This is a general trend that can be observed in every type of online media.

Proportionally to the growth of available data, the retrieval of relevant information becomes increasingly time consuming. Although many people are affected by this, journalists are among those whom this concerns the most. Their job is to aggregate relevant information and put facts into context, so that their readers can understand and be informed about the ongoings of the world. They are on tight schedules and usually cannot afford to spend hours on research.

To filter out relevant information from this sea of data, search engines provide invaluable help by offering powerful text-based filtering. However, the problem still remains for binary data like audio and video recordings, which cannot be directly accessed by text-based search engines. When dealing with very recent events, interviews are the most current and therefore possibly the most important pieces of information. Making this kind of data searchable is therefore highly desirable.

At this stage automatic speech recognition (ASR) comes into play. By automatically creating a transcript, the spoken word becomes accessible to text search, and queries for keywords are made possible. While this is an important improvement for accessibility, a lot of crucial information contained in the recording is still not captured. For example, the word „immigrant“ can have a very different connotation and gravity if used by a populist right-wing rally speaker or by a foreign minister. If a political scandal is happening, one might only be interested in statements made by the politicians involved. In case of an upcoming election, all statements of the candidates, regardless of topic, may be of interest. To provide that information, SID must be performed. While this can be done manually, resulting in very robust transcripts, this approach does not scale well. Tackling the challenge of exponentially growing data is not feasible without an automated process. SID is a research topic that has seen substantial gains in accuracy and robustness over last years, but it has not yet established itself as a helpful, large-scale tool outside the research community.

This thesis sets out to establish a workflow to provide automatic speaker identification. It's application is to help journalists searching on speeches given in the German parliament (Bundestag). This is a contribution to the News-Stream 3.0 project [new14], a BMBF[1] funded research project that addresses accessibility of various data sources for journalists.

1.1 Problem statement

While automatic speech recognition is now widely deployed in media archives and even on Youtube, SID is still not widely available. The Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS) has already implemented a SID system, which is currently in use in the News-Stream 3.0 project, but it uses an old approach that has been outperformed by newer strategies in both execution speed and recognition rate. The goal is to establish a new SID workflow using state of the art algorithms that improve recognition speed and quality using real world data and maintains compatibility with the rest of the News-Stream 3.0 framework.

This requires finding implementations of top performing algorithms, which have to be usable in the News-Stream 3.0 project both from a legal and technical point of view. Annotated training and

[1] Bundesministerium für Bildung und Forschung

testing data has to be available so that the required models can be trained and tested. The available algorithms have to be evaluated so a well suited workflow configuration can be determined.

Since the set of people in the focus of the media and public interest changes over time, it is required that new speakers can be introduced to the running system. It is desirable to incorporated new speakers into the recognition process without having to make manual adjustments to the system or to retrain existing models.

1.2 Solution approach

The workflow will be based on the i-vector approach since recent publications reported substantially decreased error rates compared to previous reports [KDS+14, GRM14, LR14]. Due to its model representation, it offers fast and flexible scoring functions [DKD+11].

In order to assemble a prototypical workflow, different available SID toolkits will be evaluated based on the technical and legal requirements of the News-Stream 3.0 project. State of the art algorithms applicable to i-vectors have to be provided, so the most promising approaches published in the literature can be compared. Algorithms of interest can be categorized into normalization and scoring methods: normalization algorithms of interest are EFR, WCCN and LDA. Scoring functions of interest are cosine scoring and PLDA.

In order to develop a workflow optimized for low error rate, an initial fully-function workflow will be established based on successful configurations in the literature. Using this as a starting point, the following steps will be performed: First, different size for the i-vectors and the background model will be compared. Second, different i-vector normalization methods will be compared, using the same scoring function. Third, cosine scoring and PLDA will be studied. Finally, execution speed measurements will be taken and all results are compared against the existing workflow.

All tests will be executed on representative data, containing 235 manually labeled speakers and 33 hours of speech taken from the german parliament. For model training, additional data from the NIST SRE 2008 corpus [Gro08] corpus is included.

Finally, a scheme will be outlined on how to implement the workflow to provide for simple introduction of new speaker models without having to manually retrain existing models.

1.3 Major outcome

A prototypical i-vector based speaker identification workflow was established using the ALIZE toolkit. The models which are required to perform speaker identification were trained using general, multilanguage speech and domain specific speech in german. The universal background model and the total variability matrix were trained on the NIST SRE 2008 training corpus and a labeled set of speeches from the German parliament. Then, different algorithms and parameters were systematically evaluated on a development data set with the goal of minimizing the classification error on the development set. This was done in parallel for cosine scoring and PLDA scoring, since both scoring functions have shown to perform well in the literature. The final workflow configuration was chosen using an independent test data set. By comparing the previous implementation to the new workflow, it was then shown that the new workflow provides both a faster classification runtime and a lower error rate than the existing implementation. Finally, a scheme for extending the set of known speakers is presented.

1.4 Structure of this document

This document is structured as following:

Chapter 2 outlines the project that this thesis is embedded in and details the requirements on the new workflow.

Chapter 3 gives an introduction to speech analysis and explains the important properties of speech and speakers.

Chapter 4 gives a thorough introduction into the SID workflow and introduces the algorithms used in the experiments.

Chapter 5 presents the results of the performed experiments, and compares them to the existing workflow.

Finally, chapter 6 summarizes the findings and gives an outlook on future work.

Chapter 2

News-Stream 3.0

This chapter introduces the News-Stream 3.0 project and lists project related requirements for this work. The project is a collaborative effort of Fraunhofer IAIS, Neofonie, DPA and Deutsche Welle, and it is funded by the BMBF. News-Stream 3.0 aims at providing tools for journalists to help them access and gather relevant information from heterogeneous data sources like blogs, twitter, news streams and media archives. The user might for example search for „financial crisis“ and „Angela Merkel“, and gets returned news articles reporting on Merkels position, blog entries discussing her statements, and speeches given by her on the topic.

Both current information, published on the internet, as well as archived material is made accessible to semantic search. The tool consist of a search engine, which contains indexed metadata and responds to search queries, and a metadata extraction pipeline which is processing archive content and online data streams. The gathered metadata is cross-references using linked open data (for example information on the speakers) which is provided by the DBpedia[1].

The architecture consists of analysis pipelines which extract metadata from data sources and a user front end which provides the interface for journalists to interact with the system. The task of extracting the metadata from audio and video files is addressed by Fraunhofer IAIS. A custom speech analysis toolkit called iFinder is used to provide transcripts using ASR and speaker ids using SID. The transcript allows for fulltext search on the spoken word, and speaker identification enables filtering for specific speakers.

IFinder is a software development toolkit (SDK) written in C++, which implements audio feature extraction, audio segmentation, ASR and SID among other features. These features are implemented as web services which are orchestrated by a custom middleware called AudioMining.

Both the transcript and the speaker ids are provided by the iFinder SDK. Whereas the ASR implementation was recently ported from the classical GMM-hidden Markov model (HMM) paradigm to the state of the art i-vector approach, the SID implementation is still GMM based. I-vectors not only promise increased recognition accuracy and faster processing, but the data model allows the speaker models to be updated easily. The GMM models on the other hand have to be relearned from scratch. It is therefore desirable to also port the SID workflow to the i-vector paradigm. In order to assemble a new SID workflow that can be implemented in this context, several requirements have to be met:

- The algorithms have to be implemented in C or C++ so that the code can be integrated into iFinder.
- All external libraries and toolkits have to provide a license that allows for commercial usage.
- Since the ASR implementation already uses i-vectors, it is desirable to reuse these vectors, so the workflow has to accept the feature file format used by iFinder.
- Although stream processing is not jet possible, it is intended to be implemented in the future. The new workflow should allow real time processing on the intended hardware environment. This also means that ASR and SID are performed in parallel, so the transcript is not available in before speaker identification.

A more detailed description of the iFinder SDK can be found in Chapter 4.9.

[1] http://wiki.dbpedia.org/

Chapter 3

Speaker identification

This chapter gives an introduction to the foundations of speaker identification. First, the scope of speaker identification is defined and put into context to related areas of research. The most prevalent application scenarios are presented and the general workflow, shared by all approaches to SID, is discussed. Then, the physiological basis of speech is explained in order to explain why utterances of different speakers sound different. Based on this, relevant properties for speaker identification are derived and interfering influences are described. In order to find both useful and compact representations of recorded speech, the human acoustic perception is described and common audio representations, used for both speaker identification and automatic speech recognition, are derived.

3.1 Differentiation

For this thesis, the analysis of speech recordings is divided into three areas: speech detection (SD), automatic speech recognition (ASR) and speaker identification (SID). Both SD and SID can be further divided into text-dependent and text-independent approaches, depending on whether a transcript of the recording is utilized or not. The combination of speaker aware speech detection (SD) and SID is referred to as Speaker Diarization. The ASR is not described in more detail since it is not relevant for this work except for its historical importance for the development of speech analysis. The term utterance is used to denote a segment of recorded speech which belongs to only one speaker.

3.1.1 Speech detection

Research on SD is driven by the question „Where does a recording contain speech?“. An audio recording has to be partitioned into segments, so that a segment border is placed whenever at least one speaker starts or stops talking. In a second step, each segment has to be labeled as speech or non-speech. When dealing with recordings that contains silence, noise, music, or multiple speakers, SD has to be performed before applying further analysis.

In text-independent SD, either a heuristic or a statistical model is used to identify speech [KEWP11]. In a heuristic model, the parameters are manually configured, while in a statistical model, the parameters are derived from training data. The latter are commonly represented by a GMM, while the transition probabilities of the „speech“ and „non-speech“-labels are either described by simple thresholds or a HMM. The features commonly used are:

- Frame energy, which is thresholded in order to remove silence and stationary noise.
- The zero crossing rate, where noise is considered to have a higher crossing rate than speech [BSS+97, TO00].
- The energy distribution across the full spectrum which is used for classification [YYDS11].

The zero crossing rate measures how often the audio waveform crosses the x axis in a fixed time frame. While SD can be addressed directly using speech and non-speech models, it can also be seen as a combination of the more general audio segmentation (AS), followed by a speech identification step. The established strategy for finding changes in the audio signal is to use the Bayesian information criterion (BIC) [CVR05]. A variable-sized window is slid across the audio stream and

a metric for signal variability is applied on the covered segment. If the variability increases, the window is shrunk until the exact position of the signal change is found and a segment border is placed. The speech identification then discards segments that contain no speech, and fuses segments that have the same label assigned. In case of SID, it is not only required to separate speech from non-speech, but also to cluster speech from different speakers. Note that this is not a prerequisite for tasks like ASR since speech recognition does not require information on the speaker identity.

When using test-dependent SD, the transcript can be used in conjunction with a phonetizer (which converts written text to a series of phones) to locate each syllable in the audio stream and thus produce a highly accurate segmentation. This process is called forced alignment (FA) and has been implemented using iterative refinement [MJVTG98], GMM-HMM models [YL08] or neural networks [SSR+15]. If the transcript contains the speaker names, the whole speaker identification task can be resolved by text mining and FA.

3.1.2 Speaker identification

The goal of speaker identification (SID) is to determine, who, out of a set of known speakers, spoke a given utterance. Approaches can be grouped into text-dependent versus text-independent as well as open-set versus closed-set categories.

Text-dependent SID either solely relies on the transcript to identify speakers based on wording and unique phrases, or it uses the transcript in addition to the recording. That way, specialized speaker models can be chosen depending on the phonemes contained in the utterance [LLML14]. Text-independent identification, which is by far the dominant category, does not use any additional information besides the audio recording. Although text-dependent SID can be superior to test-dependent approaches because the impact of the actual wording can be reduced in the channel normalization, in most applications no transcript is available.

Open-set identification means that the system is able to classify a given utterance as unknown. Closed-set identification will assign one of the known speakers to each utterance, even if the actual speaker is unknown to the system. Chapter 4.5.3 provides a more detailed explanation of the open-set and closed-set classification.

3.2 Application fields

This section introduces the mayor areas of application for SID. Since SID as a research field is relatively new compared to ASR, the mayor application fields were established by research on ASR.

As stated by Anguery et al. [AWH05], the focus of SID and speaker diarization has been largely dictated by funded research projects. Because of that, research has focused on three mayor fields of application: The first applications were focused on telephone speech. During the '90, attention shifted towards radio and television broadcast, and by the early '00 it focused on conference recordings. Each of these domain shifts introduced new challenges for segmentation and identification. The meeting scenario is considered „speech recognition complete“: all difficulties that can arise before and during ASR can be encountered and have to be dealt with. The diversification of noise and side channel influences affect both segmentation and recognition accuracy. [MBE+12]

3.2.1 Landline telephone

Speech originating from a telephone conversation provides data with very controlled modalities: the sample rate is constant, the microphone characteristics are standardized and the influence of surrounding noises and room acoustics are minimal because of the microphone placement. Telephone data has been of interest because it was the only available channel for remote conversations before the advent of Voice Over IP. A lot of research on telephone data has been done prior to the year 2000, e.g. [LH89, RR+95, RQD00].
Popular data sets are the switchboard corpus [GHM92] and the NIST SRE corpora, which all contain telephone conversations [Gro08, Gro12]. Telephone data is still part of current speaker identification challenges, with its most important application today being forensic analysis [KL10].

3.2.2 Broadcast news

Broadcast news are recorded in a studio using high quality microphones with little or no overlapping speech, so this kind of data is still very constrained and predictable. In contrast to telephone speech it can also contain segments of jingle music and ambient recordings. Notable work in this field was done by [SJRS97, GLA02, ZBMG05]. The data sets used for evaluation are the NIST Rich Transcript Evaluation corpora 2002 to 2004, which also include telephone speech [Gro02]. More recent work incorporates additional information like transcript analysis [EKLMP12], OCR on on-screen overlays [PBL+12] or face recognition [BPT+12].

3.2.3 Meeting recordings

Analyzing speech from meetings is the most targeted application field today, since it provides almost no restriction to the recording: Very different microphones can be used, ranging from clip-on to stand mounted. Not every speaker will talk directly into the microphone, so the room acoustics provide a strong coloration of the voice. In a discussion, it is also likely that multiple people talk at the same time or interrupt each other. When the whole room is recorded, all kind of background noises can be picked up, like street noises or the ventilation system. These conditions are represented in the NIST Rich Transcript corpora 2005 to 2009 [Gro05]. Notable contributions in this field are published by [AWH05, Mir07, BTHVF08].

3.3 Speaker identification workflow

This section introduces the high level structure of SID. Chapter 4 provides more concrete descriptions specific to the approaches used.

The general structure of SID, as shown in Fig. 3.1, is divided into an offline learning phase and the online recognition phase. The first steps in both phases, the preprocessing, encompasses the extraction of low level features from the audio data and performing silence removal to strap unmeaningfull data from the recordings. It is assumed that the recordings contain only speech from only one speaker. If this is not the case, segmentation has to be performed first. In the offline phase the background model and the speaker models are trained. This has to be done before the recognition can be performed in the online phase. The recognition consists of computing a similarity score for the audio recording and each of the speaker models. The way this is done depends heavily on the chosen approach. Using the scores for each speaker model, a decision has to be made, to which speaker the recording belongs.

3.3.1 Preprocessing

The preprocessing is required to transform the audio recording into a spectral representation suitable for content analysis. This representation is based on the human perception and is described in more detail in Chapter 3.6. The general approach is to divide the sample stream into overlapping windows and calculate a spectral representation of that window. The low level feature thus consists of an array of spectral values.

3.3.2 Learning phase

Before speakers can be identified, the system has to learn how to differentiate speakers based on how they sound. This is done by training speaker models. Depending on the approach, the form of these models varies greatly. The shared property is that the speaker model is a fixed-size representation, which is important later on for the scoring.

The preprocessing for the training phase consists of assembling a training data set and labeling the contained speakers. This is a time consuming task, which means that either a preexisting corpus is used, or, if the application is domain specific, only a few hours of data is available.

The first step in the learning phase is to find a projection that maps the low level features of the utterances to a fixed size representation. As a first step, many approaches utilize a universal background model (UBM), which is modeled by a GMM. For details see Chapter 4.2. The UBM represents the spectral energy distribution of speech in general. It is fitted to the test utterances from different speakers using expectation maximization (EM), as described in Chapter 4.2.2. The Gaussian mixtures provide a projection from the low level features into a higher dimensional feature space. The coefficients of the UBM are the expected mean values of speech, given that the training

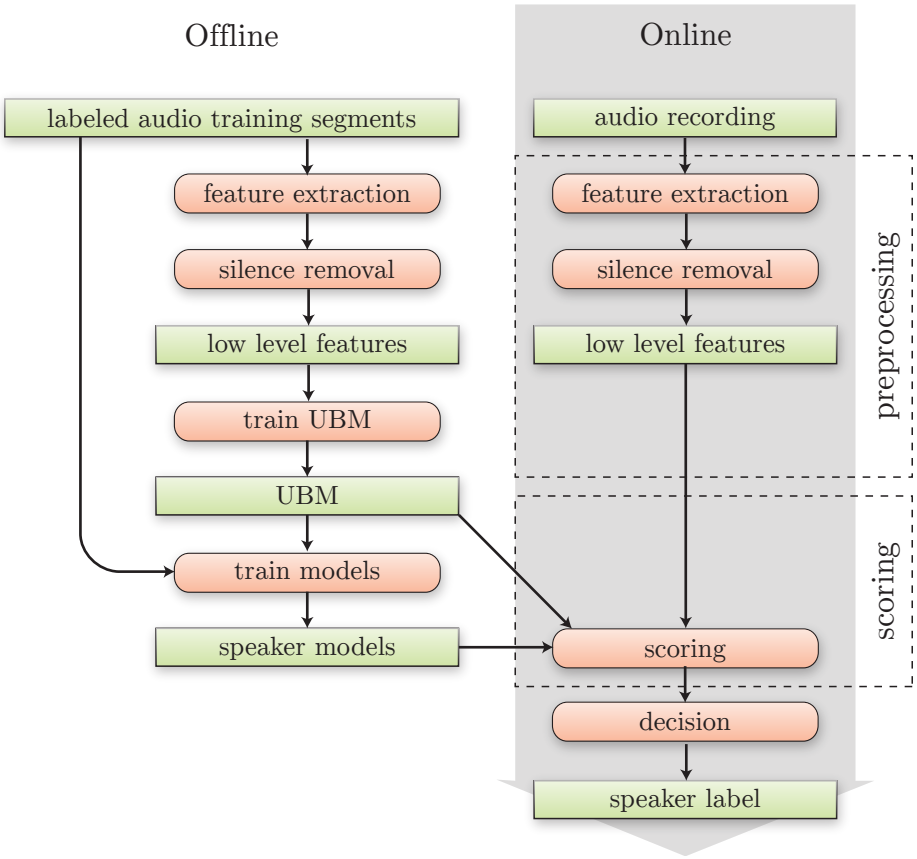


Figure 3.1: Overview of the general structure of a speaker identification workflow, independent of modeling approach. The green boxes represent artifacts, while the red round shapes represent processes. Some artifacts are omitted for clarity, if they are consumed by only one process. The arrows represent data flow.

data is large and balanced enough. In the recognition phase, this mean value is used to normalize the distribution of utterance features.

Based on this initial projection, different representations have been devised to better capture the speaker related information. The required transformations are learned by fitting the model to the labeled training utterances. Details on these methods are given in Chapter 4.

Most approaches apply a normalization step in order to redistribute the data according to the statistical assumptions of the scoring method. Normalization is also used to suppress unwanted channel information, which require a model of the content to be removed. These models are trained using different recordings for each speaker, so they represent the varying qualities between different recording sessions of the same speaker.

3.3.3 Recognition phase

During recognition, new utterances have to be projected into the UBM feature space. By applying a scoring function to the utterance and every speaker model, a similarity score is computed for each speaker. In closed-set identification, the speaker with the highest score is selected. In open-set identification, two methods can be implemented. Either a manually defined threshold is used to check if the highest score is sufficiently higher than the second-highest score. If not, then the utterance can not be classified with sufficient confidence and is declared unknown. The other strategy uses a special model for speech in general, which is trained on many different speakers. If the highest score is generated by this model, or if the highest score is not clearly larger than the general models' score (introducing the same type of threshold as in the other method), it is declared unknown. These thresholds have to be chosen manually, depending on the application.

3.4 Speech and speakers

In order to identify speakers it is important to consider what differentiates one speaker from an other. To do that, the formation of speech in the human body has to be considered. Based upon that knowledge, physiological and psychological differences can be derived and their impact on the speech itself can be estimated. These differences can be ordered from concrete, low-level features to abstract, high-level cues. Based on this categorization it is possible to derive mechanism and representations to make the information contained in these features accessible to algorithmic approaches.

3.4.1 The formation of speech

The formation of speech starts with an air pressure coming from the lungs. When the pressure is sufficiently high, the air passes through the glottis. The vocal folds can then block or restrict the airflow. When the restricted air flow causes the vocal folds to oscillate, they created the fundamental harmonic frequencies, called formants, which distinguish voiced speech from unvoiced speech. The frequencies depend on the size of the larynx. This sound excites the nasal and oral resonance chambers and is finally released through the mouth and nose. See Fig.3.2 for a schematic of the vocal tract.

Speech consists of different sounds that can be separated into consonants and vowels. Consonants are produced with the glottis partially or completely closed. They can be further categorized by place of formation and manner of articulation. Distinctive subgroups are stops, where the airflow is stopped and abruptly released, and fricatives, which are formed by forcing the air through an obstruction, formed by the lips or the tongue. Consonants have a large amount of high frequencies and have weak base frequencies.

Vowels are produced with an open vocal tract and have a prominent base frequency.

This base frequency (or the lowest resonance frequency in case of a consonant), as well as each distinctive overtone, is called a formant. Vowels can be distinguished by the frequencies of the first three formants f_0 to f_2. Higher order formants do not vary significantly across words but are more speaker dependent. The fundamental frequency f_0 of males lies in between 85 Hz to 155 Hz, for women between 165 Hz to 255 Hz, and for children in between 250 Hz to 300 Hz. The highest fundamental a trained female singer is able to sing lies around 1300 Hz. For females, f_3 can range up to 4000 Hz for stop consonants. Thus, the important frequency spectrum for speech understanding ranges from 85 Hz to 4000 Hz. [RJ11]

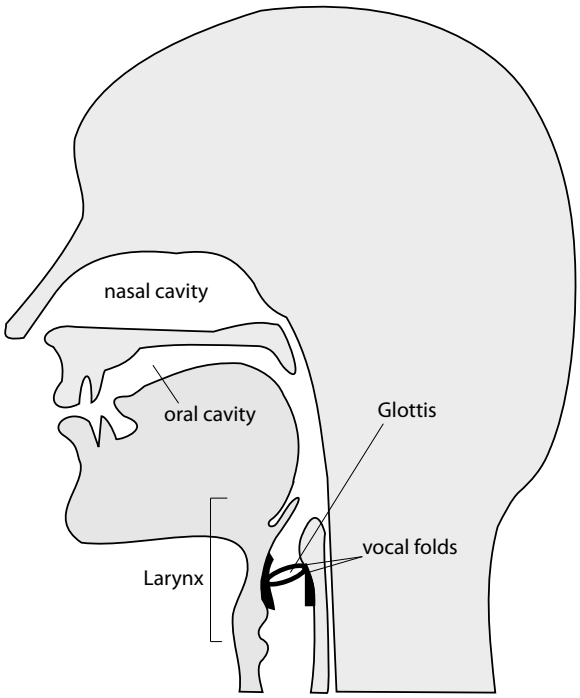


Figure 3.2: Schematic of the vocal tract, showing the physiological features that participate in the speech formation.

3.4.2 Differences between speakers

Important for SID are features that are unique and reasonably constant over time. Speaker differences, that do not contribute to identification, are either attributes shared by large groups or ones that can vary strongly over time.

What uniquely separates each and every speaker is their anatomical configuration, which defines the formants and ranges of frequency and rhythmic modulation that can be produced. Some speakers can also be uniquely described by a particular choice of words, speech melody or pronunciation. These higher level features are not physically defined but are shaped by social context and education. While these features contribute to the specific character of speech, they are possibly shared by others and can also change over time. This makes that kind of high-level speech features possibly ambiguous and unsuitable for identification, although they can still be used as secondary cues, given that the specific peculiarity is rare in the set of recognizable speakers. The most reliable features are still provided by the anatomic configuration of the speaker.

Not all speaker related features contribute to speaker identification: A prominent speech feature that is strongly influenced by membership of a group is the pitch of voice, which varies between men, women and children. Actively compensating these differences can decrease the recognition error by 0.2% [TSH00]. Another influence is the language and the wording which imprint their acoustical properties onto the speech. By leveraging the transcript of a recording, the error rate could be decreased by 1.5% [AB13]. Emotional states can impact the pitch and the tempo of speech. Targeting emotional variability in emotional speeches has lead to an error rate reduction of 1.3% [CY13].

3.5 Speech related information

Since not all information contained in an audio recording is helpful for speaker identification, it is important to define the required information before a feature extraction process can be established. A feature is an abstracted representation of the original data in a lower dimensional space. Through the projection, implicit information contained in the data is made explicit, and thus accessible for further processing steps.

In recorded speech there are multiple levels of information, ranging from low-level information like loudness to high-level information like mood of the speaker or dialect of the language that

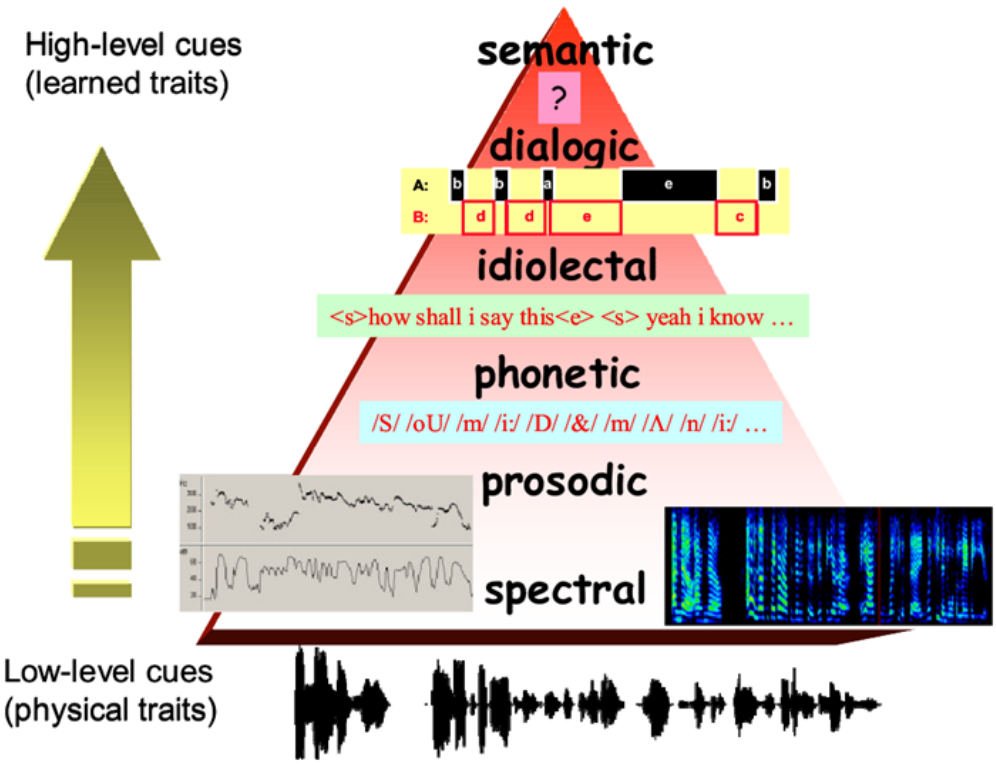


Figure 3.3: Feature pyramid for information contained in a speech recording, taken from [RCC+03].

is spoken. These features can be arranged in a pyramid as shown in Figure 3.3, where features are arranged by abstraction. The upper layers use lower-layer features as a foundation to derive their information. With each layer of feature, new domain specific knowledge is introduced that is needed to extract the desired information from the data source.

The high-level features are mainly used in ASR. Speaker identification generally relies on low-level spectral features (most prominently in the form of mel frequency cepstral coefficients (MFCCs)), on which statistical models are trained. The reason is that spoken words can be successively decomposed into levels of structure (like grammar or phonetization), so modeling these information layers can help in improving the recognition. The information needed for SID on the other does not follow a clear definition, so statistical methods are used directly on the low level features without explicitely defines hierarchies of features.

The following list describes speech related features of different levels of abstraction:

Low level cues The raw recorded data is represented by a stream of samples. This representation can already be used for silence detection (by thresholding the signal peaks) or speech detection (analyzing the zero crossing rate) [BSS+97]. For SID, the recording is usually sampled at 8 kHz. Hirsch [HHD01] has experimentally shown that higher sampling rates do not increase the recognition rate significantly.

Spectral By applying a Fourier transform on a small, windowed part of the recording a spectral representation is generated. By weighting and averaging according to a perception model, the human aural perception can be emulated as described in detail in chapter 3.6. Using either the full spectrum or a derived representation like MFCCs, basic information about the recording can be gathered: Silence detection can be done by thresholding the amount of energy, either in total or weighted by frequency. Audio segmentation can be performed by evaluating changes in the energy spectrum and placing a segment border when a difference metric reaches a threshold. The BIC (see Chapter 3.1.1) is commonly used for this [ZH05]. By incorporating a speech model, which describes the statistical energy distribution and change patterns of speech, it is possible to optimize the segmentation for speech detection [LZTZ02].

Prosodic Prosody is concerned with acoustic properties of longer parts of speech. Prosodic features can represent pitch or durations. Pitch refers to the frequency of the formants which

are identified as local maxima in the energy spectrum. Energy can also be used for impulse and rhythm detection. In combination, the length of a formant can be captured. Gender detection can be done by classifying the first formants by their pitch [MB06].

Phonetic By introducing knowledge about phones, the acoustic building blocks of speech, it becomes possible to identify phones in the the speech recording. When analyzing a large amount of speech, this can be used for language detection by comparing the distribution of phones against statistical language models. Often phones are treated as n-grams (an ordered set of n instances) so the immediate context of a phone is captured. When introducing speaker models for each phone, it becomes possible to identify the speaker [AKC$^+$02].

Idiolectal The idiolect of a segment is everything that is specific to this segment and not shared with others. One feature is the wording. To perform speech recognition, a language-depending model has to exist which provides a mapping from phone n-grams to words. These words can themselves be provided as n-grams to represent probabilities of word combinations.

Dialogic A dialogue consists of the multiple speakers talking in alteration. The change of speakers, and the length of an utterance, are features that characterize the structure of a dialogue. Having a model that provides n-grams of segment types, it may be possible to classify a speech recording by type, e.g., dialog, stage discussion or talk show.

Semantic Semantics looks at the context of the speech. Text mining can be used to derive meaning from the words spoken, so keywords can be found to summarize the contents of the speech.

3.6 Human perception modeling

The first step in speech analysis is the extraction of low level features. Several feature models have been devised specifically for speech, which all build upon the findings of psychoacoustics. The subjective perception of acoustical stimuli has been thoroughly studied, and the human ability to identify speakers and understand the content of speech, even in noisy environments, shows that high quality SID and ASR are possible when using the human perception model.

Three findings from psychoacoustics are commonly utilized by speech features:

- The perception of pitch is not linear. Stevens at al. [SVN37] introduced the Mel scale (from *melody*), defined as

$$m = 2595 \cdot log_{10}\left(1 + \frac{f}{700}\right) \quad . \tag{3.1}$$

  The function maps the perceived distance of a frequency f to the 1000 Hz reference tone from the Hertz scale onto the Mel scale. The mathematical definition is not standardized and several definitions have been proposed [FM37, Fan68].

- Perceived differences in loudness are logarithmic. Loudness is given in dB, where 0 dB is defined to be the perception threshold. Doubling the energy of a sound results in +3 dB in loudness.

- Two frequencies become indistinguishable if their frequencies are very similar. The frequency range around a reference note, in which this masking effect occurs, is called the critical band and can be modeled by the auditory filter.

The extraction of low level features loosely corresponds to the transformation from sound pressure to perceived stimulus. In this section, two different features are presented, which have been used successfully for speaker identification: mel frequency cepstral coefficients (MFCCs) and perceptual linear prediction (PLP). Both provide a compressed representation of a short frame of audio, by projecting the samples in the frame to a fixed number of feature values. They do so by leveraging different hearing and voicing properties. While MFCCs only consider the human hearing, PLP explicitly targets the formants as important speech properties. Both features (and variations of them) have been successfully used for SID [LVH$^+$11, KAM$^+$13].

Common processing steps, shared by both extraction processes, are the separation of the recording into overlapping, equally sized frames in the range of 20 to 40 milliseconds. A window function, usually a Hamming window, is applied to suppress frequency distortions by the segmentation.

Then, a discrete Fourier transform (DFT) is applied on each frame, turning the segment into a spectrum.

To make the features more robust against slowly varying noise, it is common practice to normalize the energy of each frame before generating the spectrum. Since the energy can be interesting later on, for example in silence detection, the energy is explicitly added to the feature vector after the feature values have been extracted.

To make time changes in the signal more accessible for further analysis steps, first and second order derivative values are often computed, regardless of feature type used. [DM80]

While different low-level feature representations have been used for SID, MFCCs are generally the norm and are used in the experiments of this thesis. PLP is explained for completeness and to provide a better understanding of the possibilities at this processing stage.

3.6.1 Mel frequency cepstral coefficient

MFCCs are the most commonly used low level feature in speaker and speech analysis. They are derived from psychoacoustics and model the human perception [MBE10]. The extraction process is as follows:

1. Before applying the DFT on the frames, a preemphasis is applied to increase the energy at higher frequencies. This models the increasing loudness perception for higher frequencies. The decreasing sensitivity towards the upper end of the hearing range is disregarded since it will not be considered in the following processing steps.

2. The frequency bands of the DFT are summed by a fixed number of bands using overlapping, triangular windows. The windows are distributed according to the mel scale. The number of bands can be freely chosen, 20 to 40 bands are common values. These bands correspond to the critical bands of hearing.

3. Finally, a discrete cosine transformation (DCT) is performed on the reduced and warped spectrum. By doing so, the signal is turned into a cepstrum (an anagram of spectrum, highlighting the fact that the second transformation does not project back into the time domain).

3.6.2 Perceptual linear prediction

Similar to MFCCs, PLP models the human perception when extracting the feature vector, but it also considers the voicing. Linear prediction with autoregressive modeling is used to model the most significant formants in the frame and represent the frame spectrum in terms of formant frequency and intensity.

1. Like with MFCCs, the signal is segmented into frames and a DFT is performed.

2. Also similar to MFCCs, the spectrum is summed into bands, but the window function used is not triangular but approximates of the auditory filter. The auditory filter is a band-pass filter that describes the sensitivity of a section on the basilar membrane. Instead of using the Mel scale for window spacing, the Bark scale is used:

$$B(x) = 6 \cdot \ln\left(\frac{x}{1200\pi} + (\frac{x}{1200\pi}^2 + 1)^{0.5}\right) \quad . \tag{3.2}$$

   Equal distance on the Bark scale corresponds to equal perceived pitch, whereas the Mel scale accounts for perceived pitch differences.

3. The loudness is modeled by weighting the frequency band values according to the power law of Stevens [Ste57].

4. Next, the formants are extracted by perceptual linear prediction. Linear prediction tries to estimate the signal spectrum x_n^m of a frame n, containing m spectral values, as a linear combination of previous frames. [Her90]

$$x_n^m = \sum_{i=0}^{p} a_i^m \cdot x_{n-i}^m \tag{3.3}$$

The factors a^m are the predictor coefficients and p is the order of the predictor (the number of frames considered). a^m contains a weight for each frequency band. This is called an all-pole filter of mth order. The coefficients are found by using the autocorrelation method for spectral modeling: Instead of estimating the upcoming frame based on the last ones, the current frame is „estimated“ based on itself. In other words, the filter used for modeling the predicted frame, is adapted to the current frame only and thus gives a compressed representation of it. The adaption is done using the autocorrelation matrix $x_n x_n^T$ to find values for a by minimizing an error function, typically the least squares criterion. The filter order p is chosen according to the filter properties of the vocal tract, so the filter is able to to capture the most important formant frequencies that occur in speech.

5. Finally, cepstral features are computed by applying a DCT to the a^m.

3.7 Channel information

A speech recording does not only contain information about the speaker and the speech, but also about the environment and the recording devices. These influences are called channel information since they stem from the transmission channel which provides the recording. Initially, the term was used to describe the influence of different microphones and transmission lines to the recorder, telephone recordings being a notable example for carrying a strong influence. However, the meaning of the term has been broadened to also encompass background signals like music or street noise. Depending on context, the same information is referred to as session information, since it is specific to the recording session and not the speaker [KBOD07]. Two types of channel information can be distinguished: Filtering effects, that alter the speech signal in time or frequency domain, and additive signals.

When performing SID, the channel information is unwanted since it alters the signal in a way that is uncorrelated with the speaker identity and thus distracts the classifier. Recognizing the same speaker from a microphone and a telephone recording becomes impossible without explicitly accounting for the alteration of the recordings. Developing methods for separating the impact of the speaker and the channel is the central challenge of current research in SID.

Chapter 4

Algorithms

This chapter introduces the fundamental algorithms which are used in this thesis. Chapter 4.1 starts with introducing the established quality measures and describing the fundamentals of model training. Thereafter, Chapter 4.2 describes the GMM model, and Chapter 4.3 introduces the i-vector paradigm. After introducing the i-vectors workflow in Chapter 4.3.1, the channel normalization methods and scoring functions evaluated in this thesis are explained in Chapters 4.4 and 4.5. Chapter 4.6 gives an overview of state of the art results that have been achieved in SID. In Chapter 4.7 several toolkits which provide implementations of these algorithms are compared, and Chapter 4.8 gives a detailed overview of ALIZE, the chosen toolkit for this thesis. Finally, Chapter 4.9 describes iFinder, the baseline workflow which this thesis set out to improve.

4.1 Quality measure

The task of identification consists of associating some unknown data w with one of S classes. In case of SID, it is classifying an utterance as belonging to one speaker. For each class a score $s_i|i = \{1, \ldots, S\}$ is computed, reflecting the probability of the data belonging to that class. The final classification decision uses these scores to assign the data to one of the class. When evaluating the accuracy of a classification system, a ratio of erroneous classifications to total classifications has to be computed. To do this, two sets of scores have to be available: the scores of the genuine comparisons Ω^{gen} (where the test data belongs to the tested class), and the scores of the impostors Ω^{imp} (where the input data does not belong to the tested class).

$\Omega = \Omega^{\text{gen}} \cup \Omega^{\text{imp}}$ is the set of all occurring score values, with the smallest value Ω_1 and the larges value Ω_S. The false acceptance rate (FAR), defined in Eq. 4.1, is the ratio of wrongly accepted impostors to all impostors, given some acceptance threshold τ. The false reject rate (FRR), given in Eq. 4.2, is the ratio of wrongly rejected genuine speakers to all genuine speakers, given the same threshold τ. The threshold τ is required for the decisionmaking as a reference for acceptance or rejection [SMA07].

$$\text{FAR}(\tau) = \frac{\text{FA}(\tau)}{\mid \Omega^{\text{imp}} \mid} \qquad \mid \text{FA}(\tau) = \left\{ s \mid s \in \Omega^{\text{imp}}, s > \tau \right\} \tag{4.1}$$

$$\text{FRR}(\tau) = \frac{\text{FR}(\tau)}{\mid \Omega^{\text{gen}} \mid} \qquad \mid \text{FR}(\tau) = \left\{ s \mid s \in \Omega^{\text{gen}}, s < \tau \right\} \tag{4.2}$$

Defining a quality measure for classification systems is not as straight forward as it may seem. Depending on the application, different requirements for false acceptances and false rejects may apply. For instance, when building an authentication system, false acceptances may be regarded far worse compared to a system for metadata generation, where all types of errors are equally bad. To reflect these varying requirements, different quality measures have been proposed: Traditionally, the ROC curve has been used for system quality description, but for speech analysis it was superseded by the DET curve [MDK+97]. In both cases, a single valued quality measure, the equal error rate (EER), can be derived, which is useful when comparing the performance of different systems. The EER requires a value for τ which balances the FAR and FRR. While being useful for comparing system performances, the actual application may well require a different value for τ, or even a different performance metric altogether.

A ROC graph, shown on the right in Fig. 4.1, plots the true acceptance rate (TAR) against the FAR. The TAR is defined as $\text{TAR}(\tau) = 1 - \text{FAR}(\tau)$. Its aim is to show the tradeoff between

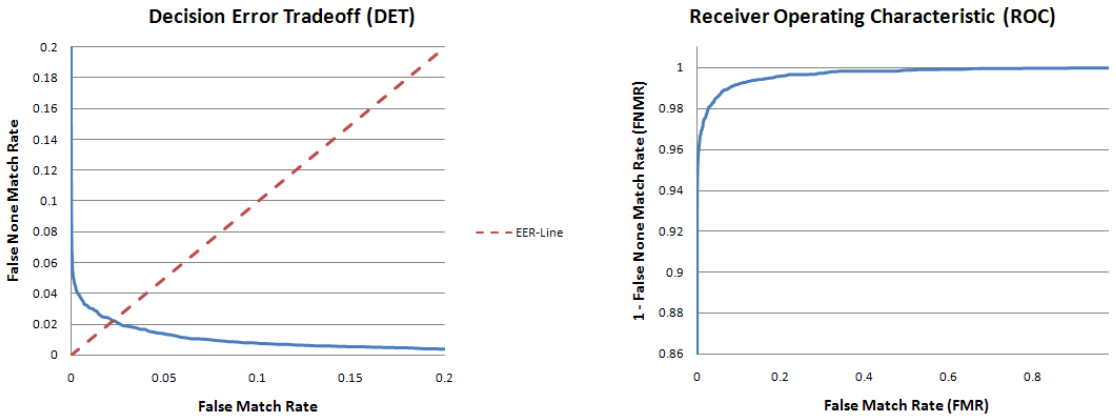


Figure 4.1: Example of a ROC and DET diagram. The image is taken from `http://biometrics.derawi.com/wp-content/uploads/2011/01/det\_roc.png`.

benefits and costs [Faw06]. The drawback of this representation is that only one of the two error types is displayed.

The DET curve, shown on the left in Fig. 4.1, was introduced by Martin et al. [MDK+97] to measure the quality of speech analysis applications. It plots both error rates against each other to highlight the balance of both error types.

The EER is the value at which the false positive rate and false negative rate are equal. In order to compute the EER, a threshold τ has to be found so that $\mathrm{FAR}(\tau) = \mathrm{FRR}(\tau)$. The EER is easily found in the DET curve diagram by intersecting the curve with the axes diagonal.

4.2 Gaussian mixture models

Gaussian mixture models are used for describing probability densities. A GMM is a weighted linear combination of C unimodal Gaussian densities

$$p(x|\theta) = \sum_{c=1}^{C} w_c \mathcal{G}[\mu_c, \Sigma_c] \tag{4.3}$$

where the weights w sum up to 1. $\mathcal{G}_c[\mu_c, \Sigma_c]$ describes the c-th Gaussian distribution with mean μ_c and $D \times D$ covariance matrix Σ_c, where D is the dimensionality of x.

$$p_c(x) = \frac{1}{(2\pi)^{D/2}|\Sigma_c|^{1/2}} \exp\Big\{ -\frac{1}{2}(x-\mu_c)^\top (\Sigma_c)^{-1}(x-\mu_c) \Big\} \tag{4.4}$$

$|\Sigma_c|$ is the determinant of the mixture covariance matrix Σ_c, and Σ_c^{-1} is the inverse. For the optimization approaches discussed in the next chapters, it is helpful to define θ to encompass all parameters of all mixtures: $\theta = \{w, \mu, \Sigma\}$.

The full covariance matrices Σ can be approximated by diagonal matrices, which makes the inversion more computationally efficient and have also shown to outperform full matrices for speaker identification [RQD00].

When training the UBM, the parameter set θ is estimated jointly by iterative EM using training data from different speakers. When training speaker models based on the UBM, the speaker model is not trained from scratch but the UBM is adapted using MAP adaption. That way a high-dimensional GMM can be adapted to a very small data set. Both procedures are described in Chapters 4.2.2 and 4.2.3.

4.2.1 GMM-based speaker identification

The structure of a GMM-based SID workflow is shown in Fig. 4.2. Compared to the general workflow description in Fig. 3.1, some new elements specific to the GMM approach are introduced.

As in the general model, the offline training phase starts with labeled training data. The first steps perform extraction of low level features from the audio recordings and silence removal. Based on the low level features the UBM is trained using EM, as explained below. After the UBM, the

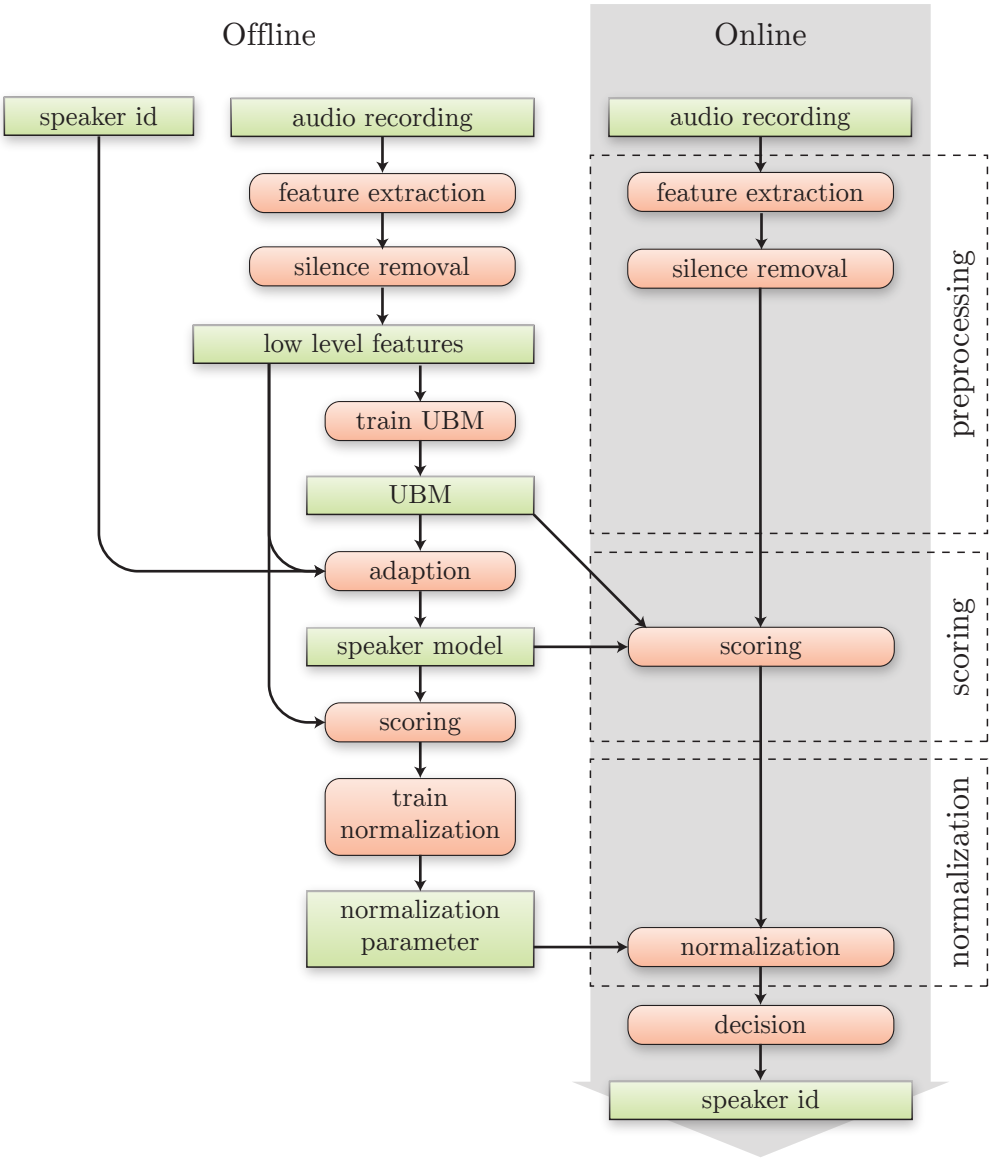


Figure 4.2: Schematic of the GMM-based SID workflow.

speaker models are created. Theses are also GMMs which are adapted from the UBM using as set of training utterances from one speaker per model. Using these models, a set of scores in computed for each training utterance. Using the scores and the original ground truth, provided in the form of utterance speaker labels, a set of score normalization parameters are computed which account for channel influences on the scoring.

Just as the offline phase, the online phase starts with extracting low level features and removing silent parts. Using the UBM, a vector representation of the recording is computed. This is compared to the speaker models by calculation a likelihood value for each model. These scores are then normalized and the classification decision is made.

4.2.2 Expectation maximization

EM is an iterative algorithm that estimates a set of hidden, latent variables on a set of measured data. It is believed that the observed data is correlated with the unobservable hidden variables. In case of GMM-based SID, the parameters of the mixtures are learned from a set of training utterances.

A speech recording $x = (y, z)$ is viewed as consisting of an observable part y (the recorded samples or extracted low level features) and an unobservable part z (the speaker characteristics). The aim is to find a parameter set θ, so that the total probability density function (PDF) $p(y, z|\theta)$ is most closely modeled, based only on the observable PDF $p(y|\theta)$. This is done by maximizing the log likelihood estimate of the observed PDF:

$$\underset{\theta}{\operatorname{argmax}} \log p(y|\theta) \tag{4.5}$$

This means finding values for the set of parameters θ that are likely to generate the whole data x, but have to be found by only using the visible part of the data y. The logarithm is applied on the likelihood function because it guarantees that the function can be derived, which is needed in order to find extrema. The logarithm ensures that the monotony of the function and the value where the function reaches its maximum are not altered. The following definitions are used in the algorithm description:

- θ is a set of parameters that describe the distribution of the data we are interested in. These parameters are learned.
- $x = (y, z)$ is the whole data, which consists of a visible part y and a hidden part z.
- $p(x|\theta)$ describes the probability of seeing some x under the condition that the parameters θ are a true description of the whole data.
- $p(x|y, \theta)$ gives the probability of some x, given the true parameters θ and the observed data y.

The observed data y, the parametric densities $p(y|\theta)$ and $p(x|\theta)$, as well as some description of x must be provided. The challenge addressed by EM is to optimize the parameters θ without knowing the data z. The EM algorithm uses two alternating steps, which are executed until the algorithm converges: the estimation step and the maximization step.

Estimation step: In the first step, an estimate of the probability distribution of the whole data is made, using some initial values for $\theta^{it}|it = 0$. Given these initial parameters and the observed data y, the conditional probability distribution of the whole data $p(x|y, \theta^{it})$ is estimated. This is an estimation of the actual probability distribution P(x|y, z) since the hidden data part z is unknown. In the next step, the expected value $E[p(x|\theta)]$ will be maximized, so a function has to be created that can be used in the maximization step and which uses the available knowledge. This function is called the Q-function:

$$Q(\theta|\theta^{it}) = \text{expected} \log p(x|\theta) = \int_X \log p(x|\theta) p(x|y, \theta^{it}) dx \quad . \tag{4.6}$$

In the Q-function, the integral is computed over the log-likelihood of all possible x. The likelihood of x is weighted by the (guessed) probability of seeing exactly that x.

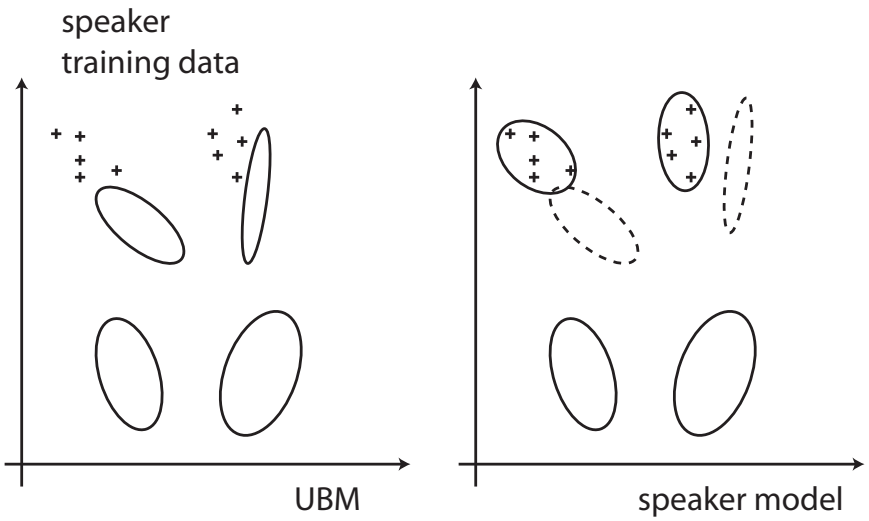


Figure 4.3: Visualization of the MAP adaption. Only mixtures that are supported by training data are adapted. [RQD00]

Maximization step: The second step tries to find a new θ which maximizes the Q-function:

$$\theta^{it+1} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta^{it}) \quad . \tag{4.7}$$

These new values are then given back to the estimation step for a new iteration. [CG10]

It is important to note that EM optimization is not guaranteed to find the global optimum because it can get „stuck“ in a local optimum. In order to account for this, it is common to perform multiple training runs with different initial values and use the best result.

4.2.3 Maximum a posteriori adaptation

Maximum a posteriori, also known as Bayesian learning, is a modification of the EM algorithm which, in the expectation step described in Eq. 4.7, maximizes the posterior instead of the likelihood:

$$\theta^{it+1} = \underset{\theta}{\operatorname{argmax}} \left(Q(\theta|\theta^{it}) + \log p(\theta)\right) \quad . \tag{4.8}$$

Thus, it is possible to introduce prior knowledge about the distribution of θ.

In the case of training speaker models, the computed values are not directly used but mixed with those of the original UBM. For each mixture, a weighting factor is computed based on the number of training data that are closest to it, as shown in Fig. 4.3. That way only those mixtures are adapted that have support by the training data. Experiments show that adapting a general model for speakers outperforms models trained from ground up using EM [Rey97].

4.3 I-vectors

For a long time, GMMs have been the standard modeling approach for SID, until Dehak et al. [DKD+11] introduced i-vector modeling, which has replaced GMM modeling as the de facto standard in speech and speaker analysis.

An i-vector, a short term for identity vector, is a fixed size, low dimensional feature vector that is extracted from the UBM space representation using joint factor analysis. Joint factor analysis (JFA) is a statistical method for decomposing the original data into one or more independent sets of latent factors, based on their variability. While JFA was initially used for channel normalization, separating useful data from distracting information, the i-vector extraction is more of a preprocessing step, in the sense that no channel influences are targeted but a general dimensionality reduction is performed which retains both speaker and channel information.

After introducing the general i-vector workflow, JFA is described as a theoretical basis from which the i-vector paradigm is derived.

4.3.1 I-vector based speaker identification

The general workflow for i-vector based SID is similar to the GMM-based workflow shown in Fig. 4.2. The two main differences to the GMM-based workflow are the extraction of i-vectors, which are then used for scoring, and the performance of channel normalization on the i-vectors before the scoring.

The offline phase starts with extracting low level features from the audio recordings and training the UBM. The important difference to the GMM-based approach is the total variability (TV) matrix, which projects from the GMM parameter space into the lower-dimensional TV space. Using the TV matrix, i-vectors are extracted from the training utterances. These i-vectors, together with the speaker ids, are then used to train one or more normalization matrices, which normalize the i-vector distribution and remove channel influences. Generating the speaker models is different from the GMM approach in that they don't have to be trained, but can simply be created by averaging over the i-vectors of utterances belonging to the required speaker. Scoring is performed using some distance metric to compare the utterance i-vector with all the speaker model i-vectors.

The online phase starts with the same preprocessing steps as the offline phase, after which the i-vector extraction is performed. Using the normalization matrices, the utterance i-vectors are redistributed and scores are computed using the speaker models. Based on these scores a classification decision is made.

4.3.2 Joint factor analysis (JFA)

Initially, JFA was introduced as a channel normalization method that tries to extract the speaker dependent information s and the channel information c from the incoming supervector w, as stated in Eq. 4.9. A supervector denotes a large vector $x \in \mathbb{R}^{ab}$ which is formed by appending multiple small vectors $y_i \in \mathbb{R}^a | i = 1, \ldots, b$ onto each other. In the case if i-vector SID, the utterance supervector $w \in \mathbb{R}^{CF}$ is formed by evaluating the C UBM mixtures on each dimension of the low level vector F. s and c are assumed to be statistically independent.

$$w = s + c \tag{4.9}$$
$$s = \mu + Vy + Dz \tag{4.10}$$
$$c = Ux \tag{4.11}$$
$$w = \mu + Vy + Dz + Ux \tag{4.12}$$

The speaker dependent information s, as described in Eq. 4.10, consists of the UBM mean vector μ, the speaker subspace projection matrix V and the representation of the utterance in that subspace y, and residual matrix D with vector z capturing all information that is not accounted for otherwise.

$\mu \in \mathbb{R}^{CF}$ describes speech in general and captures the assumed mean value of all utterances that are expected to be encountered. It corresponds to the mean value of the UBM. $V \in \mathbb{R}^{CF \times R_s}$ is the speaker related projection matrix that maps from a lower dimensional speaker space with R_s dimensions to the high dimensional input space. Its columns are referred to as eigenvoices. The vector $y \in \mathbb{R}^{R_s}$ contains the hidden speaker factors and is treated as a normally distributed random vector. $D \in \mathbb{R}^{CF \times CF}$ is a diagonal residual matrix, which accounts for speaker related information that does not contribute to speaker identity, like the language or the actual words that are spoken. This information is captured in the random vector $z \in \mathbb{R}^{CF}$. It is assumed that s is normally distributed with mean μ and covariance matrix $D^2 + VV^\top$.

The channel dependent information is described by Eq. 4.11. It consists of the channel subspace matrix $U \in \mathbb{R}^{CF \times R_c}$, which columns are referred to as eigenchannels, and the hidden channel factors $x \in \mathbb{R}^{R_c}$.

When substituting equations 4.10 and 4.11 into Eq. 4.9, the complete JFA model, Eq. 4.12, is obtained. As defined in Chapter 4.2, Σ_c is a covariance matrix for the UBM mixture component c. In JFA, these matrices are diagonal, and a super-covariance matrix $\Sigma \in \mathbb{R}^{CF \times CF}$ is defined, whose diagonal is the concatenation of the UBM covariance matrices. This construct is used later on. [KOD+08]

4.3.3 JFA model training

The matrices V, D and U must be trained using labeled data. Instead of training them jointly, a decoupled procedure is used, in which one matrix is trained after another by treating the yet

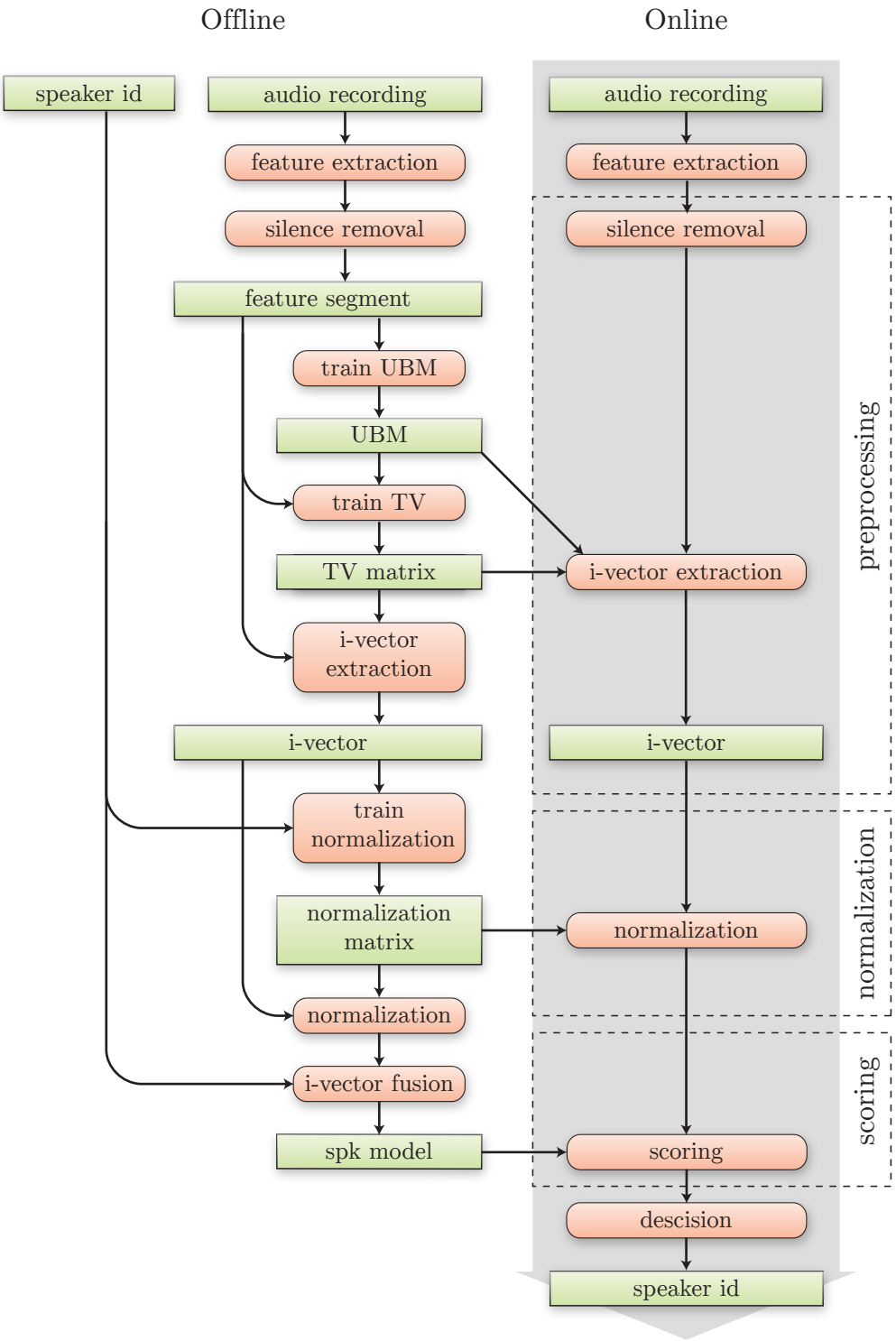


Figure 4.4: Schematic of the i-vector based SID workflow

untrained matrices as zero. This has shown to produce higher quality models than using joint estimation [KD04]. Two methods for training are used:

- Classical MAP estimation is used to train D.
- Eigenvoice MAP estimation is used to train the matrices V and U.

The matrices are trained in the following order:

1. First, the eigenvoice matrix V is trained using eigenvoice MAP estimation and assuming $D = 0$ and $U = 0$. The eigenvoice matrix covers the speaker-related principal dimensions. Assuming $D = 0$ and $U = 0$ means that we pretend that no channel influence is present and the data contains only speaker-identifying information.
2. Train the eigenchannel matrix U, using V and assuming $D = 0$. This is done by modeling all speaker related information that is till present in the data when the influence of V is removed.
3. Train the residual matrix D using V and U. This captures everything that is not covered by V or U.

The firsts step in training the V matrix is to compute a set of statics for each time frame t of speaker σ, using the components c of the UBM. The following zero order and first order statistics have to be computed:

$$N_c(\sigma) = \sum_{t \in \sigma} \gamma_t(c) \tag{4.13}$$

$$F_c(\sigma) = \sum_{t \in \sigma} \gamma_t(c) Y_t \tag{4.14}$$

$\gamma_t(c)$ is the posterior of mixture component c accounting for the feature vector $Y_t \in \mathbb{R}^F$ at time frame t. $N_c(\sigma) \in \mathbb{R}$ gives the total posterior probability of mixture component c. $F_c(\sigma) \in \mathbb{R}^F$ computes a vector that consists of the sum of all time frames of feature vector Y_t, weighted by the posterior of mixture c at each t. The next step is to normalized the first order statistic with the UBM mean value μ_c:

$$\tilde{F}_c\sigma = \sum_{t \in \sigma} \gamma_t(c)(Y_t - \mu_c) \quad . \tag{4.15}$$

In order to make these statistics usable for further computations, they have to be turned into a matrix $NN(\sigma) \in \mathbb{R}^{CF \times CF}$ and a vector $\tilde{FF}(\sigma) \in \mathbb{R}^{CF}$:

$$NN(\sigma) = \begin{bmatrix} N_1(\sigma)I & & \\ & \ddots & \\ & & N_C(\sigma)I \end{bmatrix} \tag{4.16}$$

$$\tilde{FF}(\sigma) = \begin{bmatrix} \tilde{F}_1(\sigma) \\ \vdots \\ \tilde{F}_C(\sigma) \end{bmatrix} \tag{4.17}$$

$I \in \mathbb{R}^{F \times F}$ is the F-dimensional identity matrix. Using these matrices, an estimate for the hidden speaker factors $y(\sigma)$ can be computed. $y(\sigma)$ is assumed to be normally distributed, with mean and covariance defined as

$$y(\sigma) = \mathcal{G}[l_V^{-1}(\sigma)V^\top \Sigma^{-1} \tilde{FF}(\sigma), l_V^{-1}(\sigma)] \tag{4.18}$$

$$l_V(\sigma) = I + V^\top \Sigma^{-1} NN(\sigma) V \quad . \tag{4.19}$$

$l_V^{-1}(\sigma) \in \mathbb{R}^{R_s \times R_s}$ is the covariance of the posterior distribution $y(\sigma)$. For more details on the derivation of these representations see [KBD05].

To actually use these formula to calculate V, a random initialization is performed for V. Since V is not dependent on any specific speaker σ, speaker independent statistics have to be reformulated to become independent of σ:

$$A_c = \sum_\sigma N_c(\sigma) l_V^{-1}(\sigma) \tag{4.20}$$

$$\mathcal{C} = \sum_\sigma \tilde{F}(\sigma) \left(l_V^{-1}(\sigma) V^\top \Sigma^{-1} \tilde{F}(\sigma) \right)^\top \tag{4.21}$$

$A_c \in \mathbb{R}^{R_s \times R_s}$ is the covariance of mixture component c over all speakers, weighted by the total posterior of c. $\mathcal{C} \in \mathbb{R}^{CF \times R_s}$ is the posterior mean of $y(\sigma)$, weighed by the normalized posteriors for all speakers. The matrix V is then composed as

$$V = \begin{bmatrix} V_1 \\ \vdots \\ V_C \end{bmatrix} = \begin{bmatrix} A_1^{-1}\mathcal{C}_1 \\ \vdots \\ A_C^{-1}\mathcal{C}_C \end{bmatrix} \quad . \tag{4.22}$$

Using this formulation, V can be computed using an initial approximation of V. This procedure is done iteratively, using the new values for V as an initialization of V for the next iteration.

The eigenchannel matrix U is trained in a similar way, also using the zero order and first order statistics computed over the utterances of each speaker, as given in Eq. 4.13 and 4.14. The difference to the training of V is that the first order statistics (giving the posterior values of a speaker utterance) are normalized with the expectation of the posterior of y, instead of using the UBM mean μ_c. This removes the speaker characteristics, as captured in V, and the channel influences remain.

$$F_c^U(\sigma) = \sum_{t \in \sigma} \gamma_t(c)(Y_t - E[y_c(\sigma)]) \quad . \tag{4.23}$$

The dimensionality of the posterior covariance in Eq. 4.19 becomes $R_c \times R_c$. The rest of the procedure is identical [KBOD05].

For training the residual matrix D, the first order statistics are altered in a similar way. For normalization, the s is computed according similar to Eq. 4.10, but omitting the residual:

$$s' = \mu + Vy \quad . \tag{4.24}$$

For computing s' μ, V and the mean speaker distribution $E[y(\sigma)]$ from Eq. 4.18 and used. This vector is then used for normalization in the first order statistics [KOD+08]:

$$F_c^D(\sigma) = \sum_{t \in \sigma} \gamma_t(c)(Y_t - s) \quad . \tag{4.25}$$

The dimensionality of the posterior covariance in Eq. 4.19 becomes $CF \times CF$ since D captures properties of the whole feature space.

4.3.4 Total variability space

The i-vector approach simplifies the JFA model, given in Eq. 4.12, to model only the speaker related information:

$$w = \mu + Ty \quad . \tag{4.26}$$

The total variability matrix $T \in \mathbb{R}^{CF \times d_{iv}}$ is a non-quadratic matrix of low rank which projects a i-vector $y \in \mathbb{R}^{d_{iv}}$ into the original feature space. The computation of T is the same as the computation of V in JFA, except that the zero order and first order statistics in Eq. 4.13 and 4.14 are not computed over utterances from the same speaker, but using utterances from different speakers, thus turning the speaker specific statistics into general speech statistics. This means that the i-vectors still contain all channel influences, and additional channel normalization has to be performed on the i-vectors. Methods for doing so are presented in Chapter 4.4.

In order to compute the i-vector y_u for a given utterance u, the statistics are computed similar to Eq. 4.13 and 4.14:

$$N_c = \sum_t \gamma_t(c) \tag{4.27}$$

$$F_c = \sum_t \gamma_t(c) Y_t \quad . \tag{4.28}$$

Using the TV matrix T and the inverted covariance matrix Σ^{-1}, the i-vector y_u is computed as

$$y_u = \left(I + T^\top \Sigma^{-1} N(u) T\right)^{-1} T^\top \Sigma^{-1} \tilde{F}_u \tag{4.29}$$

where $I \in \mathbb{R}^{d_{iv} \times d_{iv}}$ is the identity matrix [DKD+11].

In thei-vector approach, the speaker model is also represented by an i-vector. There are two ways to generate a single i-vector from an array of utterances:

- Appending the audio recordings and extracting one i-vector from the combined recording.
- Generate one i-vector per utterance and compute a mean i-vector afterwards.

4.4 I-vector Normalization

I-vector normalization is performed before the speaker scoring. This step has two purposes: normalizing the distribution of the data in the total variability space so that statistical assumptions of the following scoring function are met, and removing unwanted channel information. In case of cosine scoring, the data is assumed to be distributed normally, according to $\mathcal{G}[0, I]$. This assumption is usually not fulfilled by real world data and has to be established explicitly.

Channel normalization refers to the reduction of information that is unrelated to the speaker identity. This can either be done by redistributing the i-vectors in the total variability space, or by projecting them into a lower dimensional subspace. These projections are trained using either the global covariance $\Sigma \in \mathbb{R}^{d_{iv} \times d_{iv}}$, the within-class covariance $\Sigma_w \in \mathbb{R}^{d_{iv} \times d_{iv}}$ and/or the between-class covariance $\Sigma_b \in \mathbb{R}^{d_{iv} \times d_{iv}}$, which are defined as

$$\Sigma_w = \sum_{\sigma=1}^{S} \frac{\Psi_\sigma}{\Psi} (\bar{y}_\sigma - \bar{y})(\bar{y}_\sigma - \bar{y})^\top \tag{4.30}$$

$$\Sigma_b = \frac{1}{\Psi} \sum_{\sigma=1}^{S} \sum_{u=1}^{\Psi} (y_{u,\sigma} - \bar{y}_\sigma)(y_{u,\sigma} - \bar{y}_\sigma)^\top \tag{4.31}$$

$$\Sigma = \Sigma_w + \Sigma_b \quad . \tag{4.32}$$

Ψ is the number of all test utterances, while Ψ_σ is the number of utterances associated with speaker σ. $\bar{y}_\sigma \in \mathbb{R}^{d_{iv}}$ is the mean vector of all utterances from speaker σ, while $\bar{y} \in \mathbb{R}^{d_{iv}}$ is the mean vector of speech in general. $y_{u,\sigma}$ is the i-vector of utterance u, which contains speech by speaker σ [BLM+12].

In order to establish normal distribution, the mean value has to be subtracted from the data and a projection has to be found which turns the total covariance matrix Σ into the identity matrix. In order to increase the influence of speaker differences on the scoring function the between-class covariance has to be maximized. To reduce the influence of channel information, the withing-class covariance has to be minimized. The following chapters describe some of the most successful algorithms that are used in SID.

4.4.1 Within-class covariance normalization (WCCN)

WCCN was introduced by Hatch et al. [HKS06] as a normalization method for support vector machine-based recognition. The idea behind WCCN is to construct upper bounds for the FAR and FRR (as defined in Chapter 4.1) and then find a linear projection that minimizes these bounds. This projection is the inverse within-class covariance matrix Σ_w^{-1}, hence the name of this method [DKD+11, HKS06].

To define the normalization projection $\varphi(w)$, the within-class covariance must be decomposed using Cholesky decomposition:

$$\varphi(w) = B^\top w \tag{4.33}$$

$$\Sigma_w^{-1} = BB^\top \quad . \tag{4.34}$$

4.4.2 Eigen factor radial (EFR) normalization

EFR normalization was proposed by Bousquet et al. [BMB11, BLM+12] as a preparation for cosine scoring. Its purpose is to solve two requirements: providing mean and covariance normalization, and remove the channel influence which is believed to be a nonlinear dilatation („radial“ effect). This is to be done without dimensionality reduction. The first requirement is dealt with by calculating the mean and covariance of the i-vectors and then subtracting the mean and use the inverse covariance matrix as a normalization projection. By using the decomposition from Eq. 4.34, the i-vector y can be transformed into y' by

$$y' = B^\top (y - \bar{y}) \quad . \tag{4.35}$$

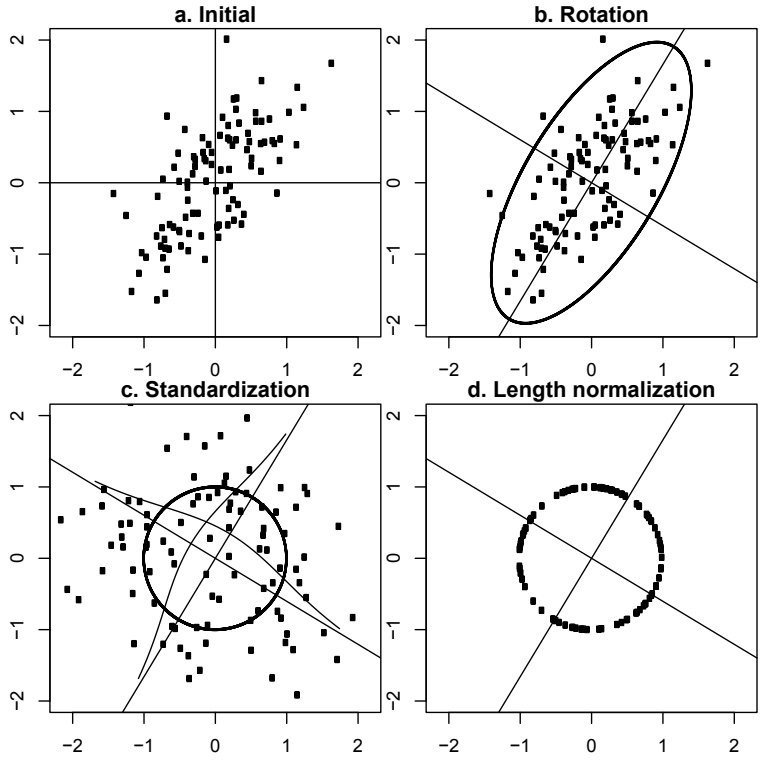


Figure 4.5: 2D visualization of the EFR normalization algorithm, taken from [BMB11].

The second goal is to remove channel influences. It is believed that the channel influences do effect mostly the length of the vectors. The countermeasure to this is length normalization. By dividing by the vector magnitude, the transformation function is obtained:

$$y' = \frac{B^\top (y - \bar{y})}{\|B^\top (y - \bar{y})\|} \quad . \tag{4.36}$$

4.4.3 Linear discriminant analysis (LDA)

LDA is a dimensionality reduction technique that projects high dimensional vectors into a subspace that minimizes the within-class covariance and maximizes the between-class covariance, as defined in Eq. 4.30 and 4.31. The directions v of this projection are found by maximizing the Rayleigh coefficient

$$J(v) = \frac{v^\top \Sigma_b v}{v^\top \Sigma_w v} \quad . \tag{4.37}$$

For a d_{lda}-dimensional subspace, the d_{lda} directions with the highest Rayleigh quotient are used. Fig. 4.6 gives a 2D example for $d_{\text{lda}} = 1$.

4.4.4 Spherical normalization (SN)

SN was developed by Bousquet et al. [BLM$^+$12] as an extension of EFR. Whereas EFR normalization is intended to be used with cosine scoring, spherical normalization (SN) adapts the same strategy for LDA or PLDA (described below). Both methods assume that the speaker related variability is low dimensional, so it can be projected into a subspace where noise influences are minimal. The goal of SN is to redistribute the i-vectors on the unit sphere surface so that the data becomes more easy to project into the speaker subspace. This is done by increasing the Rayleigh coefficient, given in Eq. 4.37. It was found that, because all i-vectors are situated on the unit sphere, the classical way of computing the within-class covariance Σ_w did not represent the actual data distribution very well. This is visualized in Fig. 4.7: The covariance of each speaker cluster is perpendicular to the unit circle so the within-class covariance Σ_w is not able to cover these covariances equally well. To resolve this problem, a spherical within-class covariance is introduced.

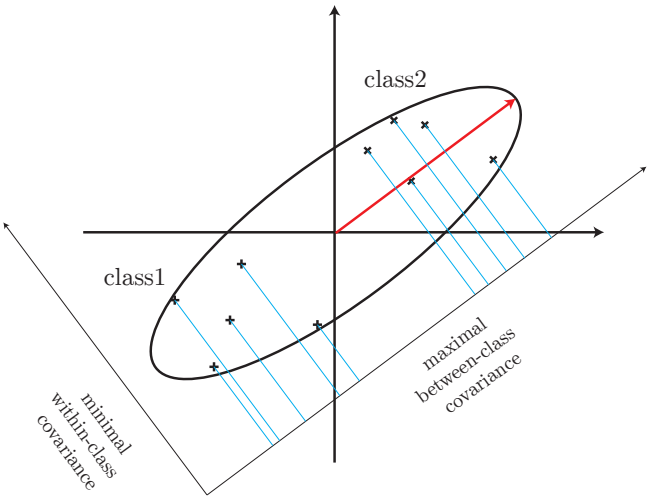


Figure 4.6: 2D visualization of the LDA projection. The red arrow represents the first subspace direction, which maximizes the between-class covariance and minimizes the within-class covariance.

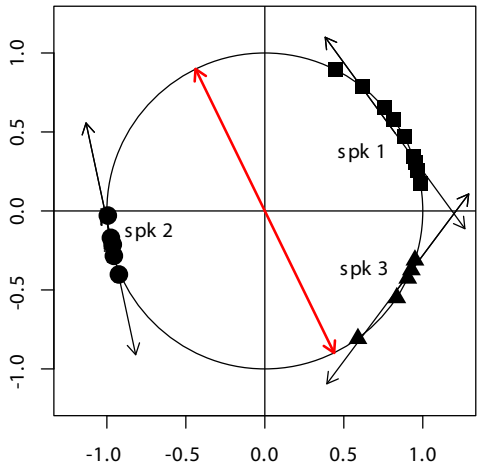


Figure 4.7: Within-class covariance of i-vectors on the unit sphere. The red arrow represents the first principal axis of the within-class covariance Σ_w, while the black arrows show the actual covariances of the three clusters. The image is taken from [BLM+12].

This is done by modifying the EFR algorithm in Eq. 4.36: EFR normalization makes Σ tending towards $\frac{I}{d_{iv}}$ (the identity matrix, divided by the number i-vector dimensions), which is spherical. To make the covariance Σ_w spherical, the same transformation can be used, only replacing the decomposed total covariance $\Sigma \to PDP^\top$ with the within-class covariance Σ_w:

$$y' = \frac{\Sigma_w^{-1/2}(y - \bar{y})}{\|\Sigma_w^{-1/2}(y - \bar{y})\|} \quad . \tag{4.38}$$

4.4.5 Probabilistic linear discriminant analysis (PLDA)

Similar to LDA, PLDA uses the within-class covariance and between-class covariance, but extends the data model by introducing Gaussian distributions. An input feature vector $w \in \mathbb{R}^{d_{iv}}$, containing speech from speaker σ, is assumed to be decomposable into a speaker dependent component $\mu + Vy_\sigma$, a channel component Ux and a residual noise component ϵ:

$$w_\sigma = \mu + Vy_\sigma + Ux + \epsilon \quad . \tag{4.39}$$

While $y_\sigma \in \mathbb{R}^{d_{\text{voice}}}$ only depends on the speaker, $x \in \mathbb{R}^{d_{\text{channel}}}$ and $\epsilon \in \mathbb{R}^{d_{iv}}$ are different for each recording. ϵ is defined to be Gaussian with diagonal covariance matrix Σ. $V \in \mathbb{R}^{d_{iv} \times d_{\text{voice}}}$ and $U \in \mathbb{R}^{d_{iv} \times d_{\text{channel}}}$ are of lower rank than the utterance. Usually diagonal matrices are used, when full covariance matrices are employed the method is referred to as two covariance modeling. Gaussian distributions are assigned to w_σ, y_σ and x, which are defined as follows:

$$Pr(w_\sigma | y_\sigma, x, \theta) = \mathcal{G}_w[\bar{y} + Vy_\sigma + Ux_{\sigma t}, \Sigma] \tag{4.40}$$

$$Pr(y_\sigma) = \mathcal{G}_y[0, I] \tag{4.41}$$

$$Pr(x) = \mathcal{G}_x[0, I] \quad . \tag{4.42}$$

Eq. 4.40 describes a conditional probability (which is the distribution of the input vectors w) as dependent on the speaker vector y_σ, the channel vector x and the parameter set $\theta = \{\bar{y}, V, U, \Sigma\}$. y_σ and x are assumed to be normally distributed. θ has to be trained on a data set using the EM algorithm described in Chapter 4.2.2. The PLDA scoring is described in Chapter 4.5.2 [PE07, BLM+12].

4.5 I-vector Scoring

Scoring functions provides a similarity score between an utterance i-vector and a speaker model (which is also an i-vector). Based on these similarity scores a classification decision is then made. Different scoring functions have been proposed, which can be distinguished into distance-based and likelihood-based functions. The scoring functions used in this work are the cosine distance function and the likelihood-based PLDA scoring.

4.5.1 Cosine scoring

Cosine similarity returns the cosine of the angle between two vectors as a measure of similarity. The return values are bound to $[-1, 1]$. It is computed by the dot product, which makes it a very fast scoring function:

$$\text{score}(w_1, w_2) = \frac{w_1 w_2}{\|w_1\| \, \|w_2\|} \quad . \tag{4.43}$$

4.5.2 PLDA scoring

For computing a similarity score of two i-vectors w_1, w_2, the likelihood of w_1 and w_2 containing speech by the same speaker has to be found. To do this, two hypotheses are introduced: H_{same},

stating that both vectors share the same speaker, and H_{diff}, stating that the speakers are different.

$$\text{score}(w_1, w_2) = \log \frac{Pr(w_1, w_2|H_{\text{same}})}{Pr(w_1, w_2|H_{\text{diff}})} \tag{4.44}$$

$$= \log \frac{Pr(w_1, w_2|H_{\text{same}})}{Pr(w_1|H_{\text{diff}})Pr(w_2|H_{\text{diff}})} \tag{4.45}$$

$$\begin{aligned} &= \log \mathcal{G}\left[\begin{bmatrix} {w'}_1 \\ {w'}_2 \end{bmatrix}, \begin{bmatrix} \Sigma + VV^\top & VV^\top \\ VV^\top & \Sigma + VV^\top \end{bmatrix}\right] \\ &\quad - \log \mathcal{G}\left[w'_1, \Sigma + VV^\top\right] - \log \mathcal{G}\left[w'_2, \Sigma + VV^\top\right] \end{aligned} \tag{4.46}$$

$$\begin{aligned} &= \begin{bmatrix} {w'}_1^\top & {w'}_2^\top \end{bmatrix} \begin{bmatrix} \Sigma + VV^\top & VV^\top \\ VV^\top & \Sigma + VV^\top \end{bmatrix}^{-1} \begin{bmatrix} w'_1 & w'_2 \end{bmatrix} \\ &\quad - {w'}_1^\top \left[\Sigma + VV^\top\right]^{-1} w'_1 \\ &\quad - {w'}_2^\top \left[\Sigma + VV^\top\right]^{-1} w'_2 + C \end{aligned} \tag{4.47}$$

Eq. 4.44 gives the relation of these two probabilities. The probability in the denominator can be separated, as stated in Eq. 4.45, since w_1 and w_2 are assumed to not share the same speaker factors y. Following [RAHK14, PE07], Eq. 4.46 formulates the probabilities as Gaussian distributions, where $w' = w - \bar{w}$ denotes the normalized input vector. The actual score computation is described in Eq. 4.47, where all constant terms are incorporated in C, which and can then be omitted.

4.5.3 Classification decision

Based on the similarity scores $\Omega = \{s_1, \ldots, s_S\}$ for each of the S speaker models, the final classification decision for one utterance is made. Depending on the application, different decision schemes can be used. A general distinction has to be made between open-set and closed-set identification: in the first case, every utterance will be assigned to one of the classes $f(s_1, \ldots, s_S) \to \{1, \ldots, S\}$, while open-set identification has the option to declare the input as unknown $f(s_1, \ldots, s_S) \to \{1, \ldots, S, O\}$. In closed-set identification the only meaningful option is to assign the speaker with the highest score:

$$f(s_1, \ldots, s_S) = i \quad | \, \Omega_S = s_i \quad . \tag{4.48}$$

Ω_S denotes the S'th order statistics, which, in case of a set containing S values, is the largest value. In open-set identification it is possible to classify an utterance as unknown, which means it is not sufficient to find the speaker with the highest score, but the classifier has to be confident enough to assign that speaker. This is represented by a confidence function p:

$$f(s_1, \ldots, s_S) = \begin{cases} i \mid \Omega_S = s_i & p(s_1, \ldots, s_S) > 0 \\ O & \text{otherwise} \end{cases} \quad . \tag{4.49}$$

Every closed-set identifier can be transformed into an open-set identifier by introducing a function p and introducing the Unknown output label. It has shown to be helpful to introduce a reference score s_O, which is generated using an average-speaker model [RAHK14]. This model is generated by averaging over all speaker models. The classification function now looks like this:

$$f(s_1, \ldots, s_S, s_O) = \begin{cases} i \mid \Omega_S = s_i & p(s_1, \ldots, s_S, s_O) > 0 \\ O & \text{otherwise} \end{cases} \quad . \tag{4.50}$$

The idea is to compare the highest score to the average score and accept the highest score only if it is sufficiently higher than the average. The function p can be defined as follows:

$$p(s_1, \ldots, s_S) = \Omega_S - s_O \alpha \tag{4.51}$$

The difference is calculated between the highest score and the weighted average score s_O. The weighting factor α is a tuning dial to influence how optimistic the system is in assigning a known speaker to an utterance. This is especially relevant in verification systems.

4.6 State of the art results

In this brief chapter an overview is given over the current state of text-independent speaker identification research.

The state of the art for modern speaker identification is „widely dominated“ [GRM14] by the use of i-vectors in combination with either PLDA as normalization and scoring function [KDS+14], or the cosine distance scoring function combined with WCCN [Deh09, FC14]. Dehak et al. [DKD+11] introduced the i-vector approach and compared it against support vector machines for speaker classification. By comparing LDA, WCCN and nuisance attribute projection as normalization steps before performing cosine distance scoring, LDA in combination with WCCN were found to produce the lowest EER of 14.44% on the NIST SRE 2008 male 10sec trial. They used a UBM with 2048 mixtures, 600 dimensions for the i-vectors and 250 dimensions for LDA. Bousquet et al [BMB11] improved the workflow of Dehak by introducing EFR and radial NAP normalization. Evaluating the NIST SRE 2008 short2 male trial with a UBM size 512 and i-vector size 400 and Mahalanobis scoring, an EER of 5.24%. Kanagasundaram et al. [KDS+14] investigated different normalization techniques for i-vectors. They proposed an extension of LDA called weighted LDA by introducing class-dependent weights for compensation the influence of different class distances, and an alternative to LDA called weighted maximum margin criterion (WMMC) which uses weight to balance the importance of within-class covariance and between-class covariance compensation. For scoring they used Gaussian PLDA (GPLDA). Based on the NIST SRE 2008 short2 trial they achieved an EER of 3.61%.

Working on recordings of TV shows, Fredouille and Charlet [FC14] applied i-vector based SID to the french television corpus REPERE. They evaluated PLDA, EFR, SN and WCCN for normalization and PLDA and cosine distance as scoring functions. The Corpus consists of 571 speakers 47 hours of recordings. They use 512 mixtures in the UBM, 200 dimensions for the i-vectors and, for PLDA scoring, rank 200 for channels and 100 for speakers. PLDA was found to be worse than all other combinations, with the best EER of 2.5% achieved with WCCN and cosine scoring. Glembek at al. [GMM+14] apply WCCN and LDA normalization, combined with PLDA scoring, to the DARPA RATS task. The task focuses on SID under unknown channel conditions. It contains 2 hours of data, 1000 speakers and 8 different channel types. By using PLP for low level features, 2048 UBM mixtures, 400 i-vector dimensions, 200 LDA dimensions and full rank PLDA channels, an EER of 6.26% was achieved. Garcia and McCree [GRM14] focus on adapting a SID workflow for domain specific use with only a small amount of training data. PLDA is used for normalization and scoring. By using 2048 UBM mixtures, 600 i-vector dimensions and 400 dimensions for PLDA, an EER of 2.32% was achieved on the Switchboard corpus. The NIST SRE 04, 05, 06, and 08 corpora were used for initial training.

While it is unclear if cosine scoring (with optimized normalization like WCCN) is superior to PLDA or not, all authors highlight the criticality of within-class covariance normalization and the importance of representative data for session normalization training.

4.7 Toolkit comparison

Several toolkits have been developed that provide algorithms and data structures for SID, using different programming languages, licenses and supporting different approaches. This sections introduces three promising toolkits that have been considered for this thesis and highlights their strengths and drawbacks in respect to the requirements of the News-Stream 3.0 project. The requirements a toolkit has to meet are:

- Offering a license that permits commercial usage.
- Providing state of the are algorithms like i-vectors and PLDA.
- Be written in either C or C++ to be integrable into the News-Stream 3.0 pipeline.
- Still being maintained.

Several widely used toolkits like SPRO [1] or HTK [2] are excluded since they focus on ASR and have to be modified to perform SID. Nonetheless do they deserve a mention since they provide low level feature extraction which can be used in conjunction with the SID toolkits. In case of ALIZE it is even necessary since ALIZE does not offer low level feature extraction natively.

[1] http://www.irisa.fr/metiss/guig/spro/
[2] http://htk.eng.cam.ac.uk/

Name	License	Language	Modeling	Normalization
ALIZE	LGPL	C++	GMM, IV, SVM	EFR, SphN., LDA, WCCN
Kaldi	Apache 2.0	C++	GMM, IV, neural n.	LDA
LIUM	GNU	Java	GMM, IV	EFR

Table 4.1: Feature matrix of SID toolkits

4.7.1 ALIZE

ALIZE is an open source platform for speaker recognition that is developed by the University of Avignon. It is written in C++, licensed under LGPL and has been tested on Linux, Mac OS and Windows. It is distributed through a SVN repository[3], with the latest commit dating back to 2014. No precompiled versions are available.

ALIZE is specialized in speaker identification and provides GMM, SVM and i-vector modeling for speaker representation. For the i-vector approach, EFR, spherical normalization, LDA and WCCN are implemented for channel normalization. The scoring functions offered by the toolkit include mahalanobis distance, cosine scoring and PLDA. ALIZE does not provide low level feature extraction, so an external tool must be used for that. [LBF+13]

4.7.2 KALDI

Kaldi is an open source toolkit for speech and speaker recognition that is maintained primarily by Daniel Povey at Johns Hopkins University, Baltimore. The name references a mythological Ethiopian goatherder who is said to have discovered the coffee plant. It is written in C/C++ and licensed under Apache v2.0. It is tested under Unix-like systems and Windows. The project is distributed through a GitHub repository[4], which is actively maintained and extended. No precompiled versions are available.

The design goals are the usage of a finite-state transducer framework, support for fast linear algebra libraries like LAPACK and having a non-restrictive license. Building upon the OpenFst library, the code is highly modularized and flexible.

Low level feature extraction supports both MFCC and PLP. For modeling, GMMs, i-vectors and neural networks are supported. For normalization, low level mean and covariance normalization is provided as well as two versions of LDA for high level features. [PGB+11]

4.7.3 LIUM

The LIUM diarization toolkit is named after the *Laboratoire d'Informatique de l'Université du Maine*, where it is developed. It is written in Java, published under the GNU license and distributed as a downloadable jar file. Version 2, the latest release, was created in 2013.

LIUM was created for speaker diarization, so it includes audio clustering, speech detection and low level MFCC extraction. Speaker models can be created using either GMMs or i-vectors. EFR is offered for normalization and mahalanobis distance is used for i-vector comparison. [RDG+13]

4.7.4 Comparison

When comparing the three toolkits, as done in Fig. 4.1, LIUM is first to be discarded since it is not written in a C-like language and does not allow commercial application. Both Kaldi and ALIZE fulfill the requirements, but since Kaldi is a general speech analysis tool and ALIZE is specialized on SID, ALIZE offers more options for normalization and scoring, like PLDA, and was thus chosen for training and evaluation.

4.8 The ALIZE toolkit

ALIZE is a toolkit that aims for providing baseline implementations of successful algorithms for SID, but by using different low level features it can be used for any classification task. Being a toolkit means that the project provides building blocks which the user has to chain into a workflow. This ensures a high degree of freedom in the application. The ALIZE toolkit actually consists of

[3] svn://alize.univ-avignon.fr/svn/ALIZE
[4] https://github.com/kaldi-asr

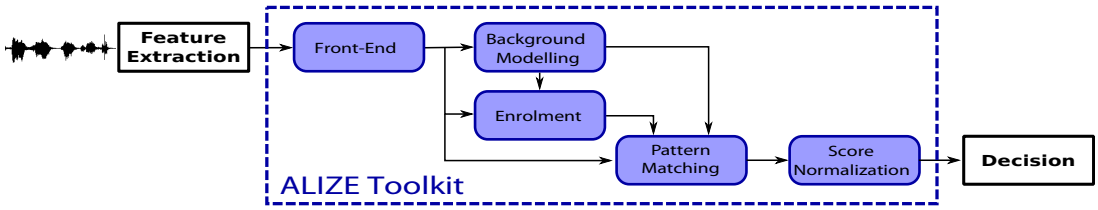


Figure 4.8: The general structure of the ALIZE toolkit, taken from [LBF+13].

two projects which build upon another: the core library which is called ALIZE, and the executables which are provided in the LIA_RAL package. The ALIZE core library contains the data structures and algorithms which are used by the executables.

These executables are grouped into steps, which are shown in Fig. 4.8. It is important to note that the low level feature extraction and the final decisionmaking are not part of the scope that ALIZE covers, so this has to be done by other tools. Each step has different executables for the three modeling approaches that are supported: GMM-based modeling, SVM-based modeling and i-vector based modeling. For each approach, different algorithms are provided. This chapter will only cover the i-vector approach since the other approaches were not used in this work.

The executables are command line driven and can be configured either by command line arguments or through a configuration file. All data is stored in files, so running a workflow requires that potentially large files (up to 4 GB) are repeatedly read and written to disk. When the prototypical workflow established here is turned into a production workflow, it is advisable to merge the code into one executable so no unnecessary file operations are executed.

The following list describes the executables of the ALIZE toolkit, grouped by processing step:

Feature extraction: As already stated, no feature extraction is provided by ALIZE itself, but file format support is provided for the SPRO[5] and HTK[6] generated feature files. A third option is using raw floating point data blobs for general features vectors. In that case, the data definition has to be provided through the configuration file.

Frontend: The frontend processing provides three functions: FeatureNormalization, EnergyDetection and LabelFusion. Feature normalization performs mean removal on the feature vectors. This can be done separately for each dimension which is helpful for silence removal. In that case, only the energy (or log energy) is normalized and then used by the energy detector to mark segments as silent. The energy detector creates a label file for each feature file which contains segments labeled as speech or non-speech. Label fusion can be used to smoothen the segment boundaries. This step is the same for all approaches.

Background modeling: Background modeling consists of training the UBM and the TV matrix. TrainWorld uses EM to train a GMM on a training data set (see Chapter 4.2.2 and 4.2), and TotalVariability uses JFA to train the total variability matrix (see Chapter 4.3.3).

Enrollment: The term enrollment is more intuitive in the GMM paradigm than in i-vector based classification. It refers to the speaker model training, which, in the GMM paradigm, is done by adapting the UBM using utterances from one speaker. Thus, these utterances are „rolled into“ the model. In the i-vector approach, the model is just an i-vector, which is extracted using IvExtractor. Neither utterance fusion nor i-vector averaging, as described in Chapter 4.3.4, are offered by ALIZE, but both are simple to implement. Training the normalization matrices can be part of the enrollment step. ALIZE allows for offline and online normalization training. This applies to all implemented normalization algorithms: EFR, spherical normalization, LDA, WCCN and PLDA (see Chapter 4.4 for descriptions). All normalization algorithms are provided by the IvNorm executable.

Pattern matching: This is the online scoring process. The IvTest executable processes a trial file, which defines pairs of i-vector files to compare. IvTest can be configured to use different scoring methods, like cosine distance, Mahalanobis distance, two-covariance scoring and

[5]https://gforge.inria.fr/projects/spro
[6]http://htk.eng.cam.ac.uk/

PLDA (see Chapter 4.4.5). It also provides the possibility to train and run any of the normalization methods online. The results are written into an ASCII file containing the i-vector names and the floating point score for each test.

Score normalization: Score normalization is used in GMM-based classification as a means of bias compensation. In the i-vector approach, this is dealt with in the normalization step before scoring.

Decisionmaking: The decisionmaking is not provided by ALIZE and must be tailored to the application at hand. A common decision making process is described in Chapter 4.5.3.

4.9 IFinder

IFinder is a software development kit for speech analysis, developed at Fraunhofer IAIS. It is written in C++ and can be used either as command line tool or through a soap webservice interface. Among others, the SDK contains modules for MFCC extraction, audio segmentation, ASR and SID.

IFinder internally uses KALDI for MFCC extraction. The exact MFCC parameters are given in Chapter 5.2. Audio segmentation is performed for finding homogenous segments of speech. The BIC criterion segmentation, as described in Chapter 3.1.1, is used. The implementation is based on [TG99, CWF10]. ASR is done using i-vector based recognition provided by KALDI.

The current SID implementation is based on [RQD00, RR$^+$95] and uses the classic GMM-UBM approach. The UBM is trained using EM, the speaker models are then adapted using MAP. The training data set for the UBM consists of 14 hours of news broadcasts and the UBM was trained with 1024 mixtures. The speaker models are adapted using 2 minutes of speaker from each speaker.

The evaluation trial used in the experiments was run on the iFinder system and produced an EER of 8.05% and took 18.15 sec. on average to analyze a 2 minutes test file.

Chapter 5

Experiments

This chapter describes the process of developing the optimized SID workflow for the Parliament corpus and discusses the results.

First, Chapter 5.1 introduces the data sets that are used to train, develop and test the workflow. Then, Chapter 5.2 presents the development strategy and starting conditions of the workflow optimization. Optimization goal is the minimization of the EER for the parliament data set. In Chapter 5.2.1 and 5.2.2, the most influential parameters for UBM and TV matrix training are explored. Then, Chapter 5.2.3 and 5.2.4 evaluate the normalization options offered for cosine and PLDA scoring. This is done separately for each scoring function, since cosine scoring itself is not parameterizable while PLDA offers parameters that strongly affect its performance. Also, ALIZE offers WCCN normalization only for cosine scoring. For both scoring functions a complete workflow is developed, so that the best workflow can be selected in the final evaluation. Chapter 5.3 looks at the runtime behavior of both workflows. The decision on the final workflow is presented in chapter 5.4. In Chapter 5.6, the final workflow is compared to the baseline implementation from the iFinder SDK. Chapter 5.7 discusses the findings and provides an outlook on future work. Finally, a scheme for providing speaker set extendability is proposed in Chapter 5.8. The EER values and DET curves used for evaluation are generated using DETware, the NIST evaluation tool[1].

5.1 Data Sources

The quality of the models trained for general speech and speakers, as well as the normalization and dimensionality reduction matrices, are heavily dependent on the characteristics of the training data. It is therefore important to verify that the data sets are balanced regarding the trained characteristics, and that the range of expected input during runtime is well reflected in the training set.

A corpus contains a possibly large number of manually labeled data. In case of SID, the labels contain at least the id of the speakers and their gender, but in case of the NIST SRE corpora, which are described below, more information like recording method, language and speaker nationality are provided. In order to gain comparability among methods, it is common to provide data set definitions and trials. Typical data set definitions are for training data, test data and evaluation data. The training set is used only for offline training and the test set is used in online mode when running the trials. In case of a system comparison, as it is done in this thesis, it is important to have a third set for final evaluation. This is a measure against overfitting on the evaluation set, just as the evaluation set is required to prevent overfitting on the training set.

This thesis uses two corpora: the NIST speech recognition evaluation 2008 and the German Parliament corpus. The following sections describe their properties and motivate their usage.

5.1.1 NIST speech recognition evaluation

The speech recognition evaluation corpus was developed by the Multimodal Information Group of the Linguistic Data Consortium (LDC) and the National Institute of Standards and Technology. Although it is not the only speech analysis related corpus released by the LDC, it is the de facto standard data set in ASR and SID. An updated version of the SRE corpus is released yearly, so a

[1] `http://www.itl.nist.gov/iad/mig/tools/DETware_v2.1.targz.htm`

unique LDC catalog number is given to each data set in order to clearly distinguish the corpora. This thesis uses the SRE 2008 training data sets LDC2011S05 and LDC2011S07. Also part of the SRE 2008 corpus are a test set and additional evaluation data, which were not utilized. In the experiments, the term NIST SRE 2008 will refer only to the mentioned training sets.

The whole SRE 2008 corpus contains 565 hours of telephone speech and 75 hours of interviews, using high quality microphones. 640 hours of telephone speech are multilingual, all other speech is in English.

The training set is separated into several conditions:

10-sec contains 10 seconds long recordings of stereo telephone conversations, with each speaker on one channel.

short2 contains either 5 minutes of stereo telephone conversation, one channel for each speaker, or 2 minutes of mono interview recordings.

3conv contains stereo telephone recordings that are composed of three different conversations. Each conversation contains the target speaker and a different conversational partner.

8conv is the same as 3conv, but contains eight different telephone conversations.

long contains eight minutes mono recordings of one speaker only.

3summed is similar to 3conv, but instead of featuring channel separation between the speakers, all information is merged into one channel.

For each utterance, a machine generated transcript is provided. For multispeaker recordings, all speaker identities are provided and the main speaker is marked. [Gro08]

The test set is separated into fever, but similar conditions: **10-sec**, **short3**, **long** and **3summed**. The definitions are the same as in the training conditions. For 13 possible combinations of training and test conditions, a separate trial is provided. Each trial consists of roughly 2000 speakers to identify and about 100 000 tests. All trials have an equal proportion of male and female speakers. The data is provided as 8-bit files, using the uncompressed sphere format. A tool for reading the sphere format is provided[2].

5.1.2 Speakers in the German parliament

The German Parliament corpus was assembled at Fraunhofer IAIS, based on publicly available recordings of speeches given in the German parliament (Bundestag). It is an internal training and benchmarking corpus that represents German planned speech, recorded with high quality equipment.

It consists of 235 speakers, 163 male and 75 female. While this is imbalanced, it reflects the gender distribution in the parliament. For each speaker there are 1 to 6 utterances of 2 minutes length. They are stored as 16 bit mono wav files. Through varying number of utterances per speaker a gender balance in the recordings is provided.

The corpus is divided into three disjoint data sets: training, development and testing. The training set consists of 1176 utterances, which corresponds to 39.2 hours of speech. The development set consists of 993 utterances, and the test set contains 413 utterances.

5.1.3 Usage in this work

Both the NIST SRE corpus and the Parliament corpus are used in the experiments. The training of the UBM and the TV matrix is done using the training sets of both corpora. One reason is that the NIST corpus provides a greater variety of languages and channel conditions, which makes the model more general. Although a data-specific model would be more appropriate in this data specific use case, it was explicitly desired to have more general UBM and TV models in order to be able to reuse the models in different projects and contexts. Another reason is that training a large GMM requires a lot of training data, which is often a problem in domain specific applications where not enough training data is available. The impact of different training data on the error rate is explored in Chapter 5.2.1.

To be able to compare MFCCs, and subsequently i-vectors, it is very important that all data shares the same bit rate and sampling rate. Although some MFCC extraction tools like SPRO

[2] ftp://jaguar.ncsl.nist.gov/pub/sphere-2.7-20120312-1513.tar.bz2

provide a bit rate parameter, it has proved to be necessary to perform bit rate conversion before MFCC extraction. Therefore, all utterances from the Parliament corpus were converted to 8 bit.

Besides the UBM and TV matrix training, all training and evaluation was done using the Parliament corpus.

5.2 Classification quality optimization

The i-vector SID workflow, as described in Chapter 4.3.1, consist of several steps that build upon each other to classify the speaker of an incoming utterance. The quality of the classification depends both on the quality of each individual step, as well as the combination of algorithms. Although evaluating the fitness of a trained model (how well has this model captured the required information to distinguish speakers) outside a fully functional workflow is possible, using likelihood values or by quantifying the data covariance, the synergy of different algorithms is not captured. The chosen approach for evaluating different algorithms and parameters is to start with a fully functional workflow, based on parameters that have been shown to be successful in the literature, and optimizing the workflow from start to finish. The base configuration, on which the following experiments build upon, consists of:

- MFCCs with 60 dimensions, including delta and double delta values. Parameters used for extraction are a preemphasis of 0.97, frame length of 20 ms, frame shift of 10 ms, 24 filter banks, 19 cepstral coefficients (plus log energy) and a liftering factor of 22. Silence removal and per-frame energy normalization is performed across all dimensions.
- A UBM with $d_{\text{gmm}} = 1024$ mixtures.
- A TV matrix with $d_{\text{iv}} = 400$ dimensions.
- EFR normalization.
- I-vector averaging for speaker model generation.
- Cosine scoring.

The configuration for the low level feature extraction is the same as in the analysis system used in the News-Stream 3.0 project. The TV matrix and the EFR matrices are trained using the parliament training data set, the evaluation is done on the parliament evaluation set. The recognition quality is given as EER in percentage. Throughout the experiments, all UBM models, TV matrices and normalization matrices are trained with 6 iterations.

5.2.1 Model data evaluation

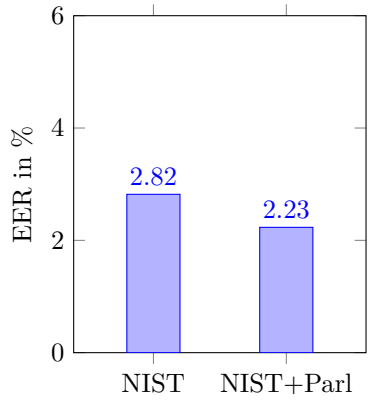
The first step in establishing an i-vector based SID workflow is to train a UBM. The important parameters for training a GMM are:

- The data set, which has to be balanced (with regard to gender and channel types) and large enough to support the number of mixtures.
- The number of mixtures.

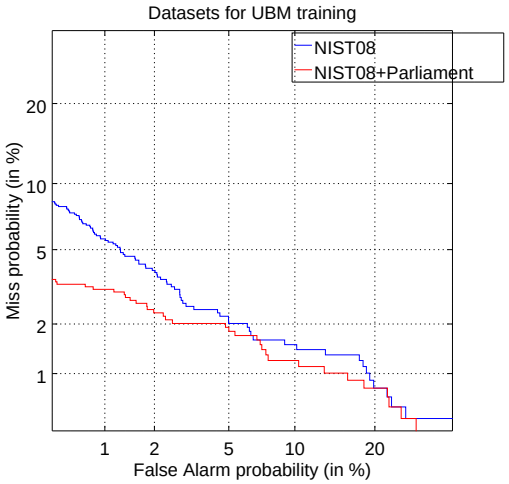
Additionally, practical concerns when using EM-based algorithms are the initial configuration and the number of iterations. Since the EM-based algorithms are only guaranteed to find a local maximum, it is important to find a good initial guess for the mixture values and a sufficiently high iteration count. Each UBM was trained three times and the best model (with lowest log likelihood) was used.

In the literature, it is common to see the usage of two gender dependent UBMs. This makes sense in the context of the NIST SRE corpora, where the trials distinguish between the genders, but in practice the gender is not known in before, so a mixed gender UBM is used.

A general problem in domain specific SID applications is the lack of sufficient amounts of data to support the mixtures, so it is common practice to combine multiple data sets for training the UBM [FC14]. We could confirm this experimentally by training a UBM on the parliament training data set. It resulted in an EER of 45.62%, which is close to random guessing. When using the NIST SRE'08 training data set, an EER of 2.82% was achieved. By using the parliament and NIST training data sets in conjunction, the EER was decreased to 2.23%. Fig. 5.1 shows the EER and DET curves. The combined data sets show a noticeable improvement in recognition rate over the model trained solely on out-of-domain data.



(a) EER using the NIST and parliament+NIST UBMs.



(b) DET curve of baseline workflow using NIST and parliament+NIST UBMs.

Figure 5.1: Comparison of the error rate using baseline workflow with two different UBMs, trained on the NIST SRE'08 training data set and NIST+parliament training data sets.

5.2.2 Model size evaluation

After establishing the data set for training the UBM, the optimal number of mixtures d_{gmm} has to be found. The following numbers of mixtures were tested: 32, 64, 128, 256, 512, 1024. The next step in the workflow consists of training the TV matrix. The most important parameter of this process is the number of dimensions d_{iv}. For each UBM size, a TV matrix was trained with 10, 20, 25, 50, 100, 200 and 400 dimensions, respectively. Fig. 5.2a shows the EER depending on d_{gmm} and d_{iv}. The influence of d_{iv} is very prominent while increasing d_{gmm} provides only slight improvement. The lowest EER of 2.21% was achieved by using $d_{\text{gmm}} = 512$ and $d_{\text{iv}} = 200$. As shown in both Fig. 5.2a and Fig. 5.2b, the differences between high-dimensional UBM and TV matrices are minor.

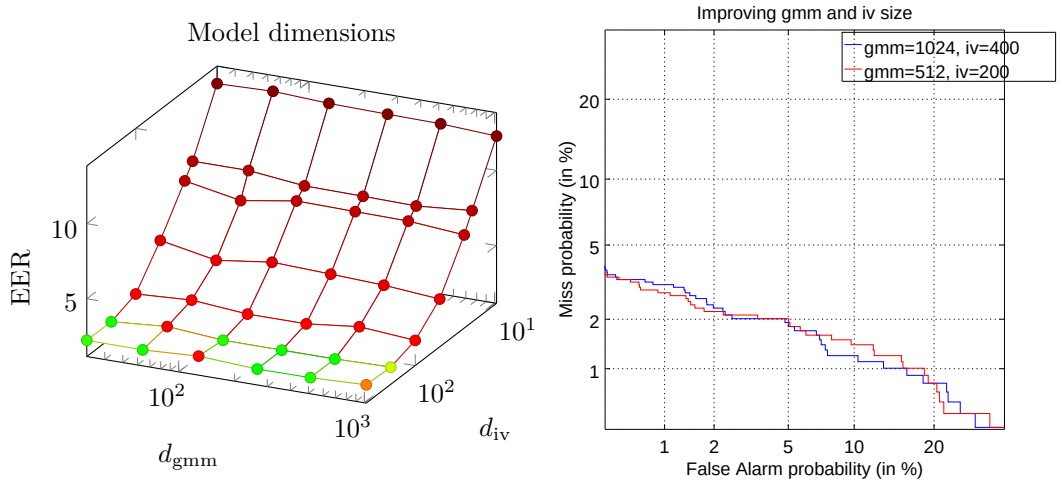
In the following, the $d_{\text{gmm}} = 512$ UBM and the $d_{\text{iv}} = 200$ TV matrix established here are used as a baseline to compare the performance of normalization and scoring methods examined in the following sections.

5.2.3 Normalization for cosine scoring

Cosine scoring, as implemented in ALIZE, offers more combinations for channel normalization algorithms compared to ALIZE's implementation of PLDA. The possible combinations are:

- Plain cosine scoring
- LDA
- WCCN
- EFR
- EFR + LDA
- EFR + WCCN
- EFR + WCCN + LDA
- Spherical normalization
- Spherical normalization + LDA
- Spherical normalization + WCCN
- Spherical normalization + WCCN + LDA.

WCCN and LDA can always be added as an additional processing step after the initial normalization. In the following evaluation, LDA was applied using $d_{\text{lda}} = 100$, which is half the dimensions of the TV space. The UBM and TV matrix established in the previous section were used. As illustrated in Fig. 5.3, LDA provides an improvement in every combination, while performing best in combination with EFR. To further improve the performance of this combination, the optimal rank for LDA is investigated. Fig. 5.4a shows the performance in relation to d_{lda}.



(a) The EER of cosine scoring with EFR, depending on d_{gmm} and d_{iv}.

(b) DET curve using $d_{\text{gmm}} = 512, d_{\text{iv}} = 200$ (red), and the baseline curve of Fig. 5.1b (blue).

Figure 5.2: Visualization of the influence of UBM and TV matrix size on the error rate, using the baseline system and the NIST+parliament UBM. While increasing d_{gmm} does only provide slightly decreased error rates, increasing d_{iv} provides very noticeable improvements which saturate around $d_{iv} = 400$. gmm $= 512, d_{\text{iv}} = 200$ provides the lowest EER of 2.21%.

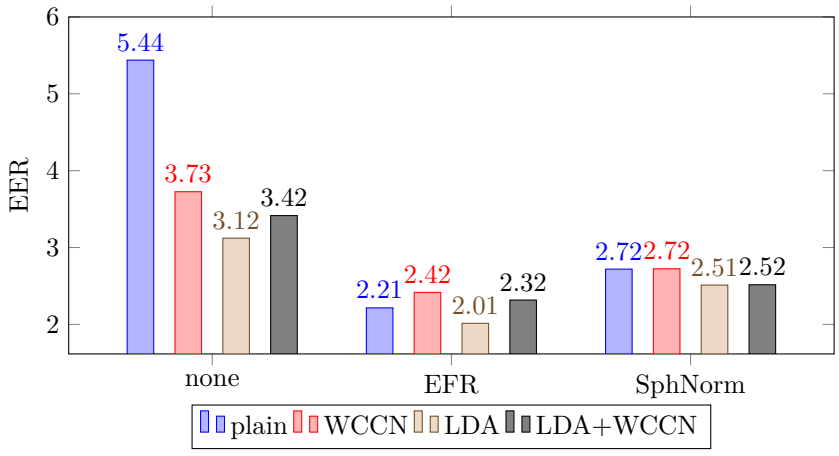
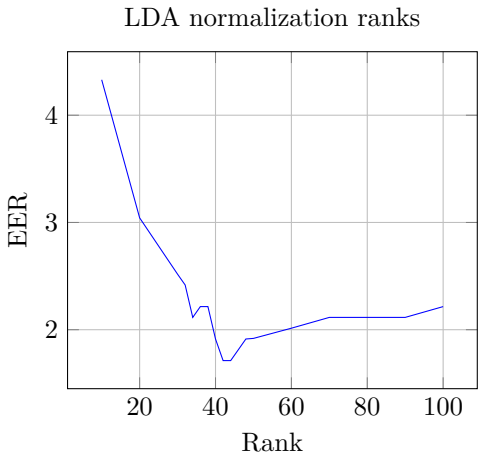
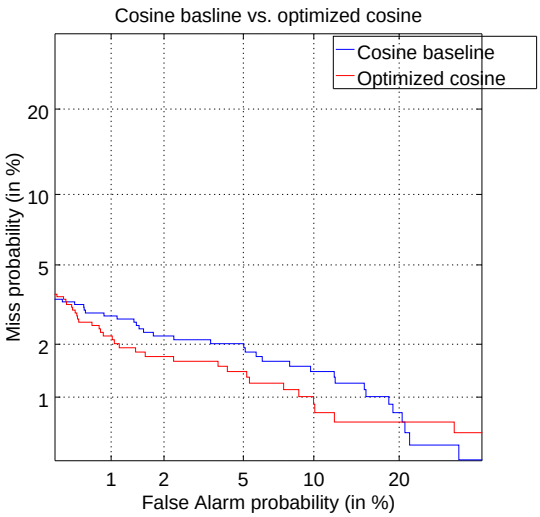


Figure 5.3: EER of cosine scoring, using different combinations of normalization algorithms. In the NONE plot, WCCN and LDA are used as stand alone normalization procedures. In the other two plots, EFR and SphNorm are the primary normalization algorithms whereas WCCN and LDA are applied in a second step. In ALIZE, WCCN is always applied last. PLAIN refers to no normalization. Note that the previous tests already used EFR normalization.

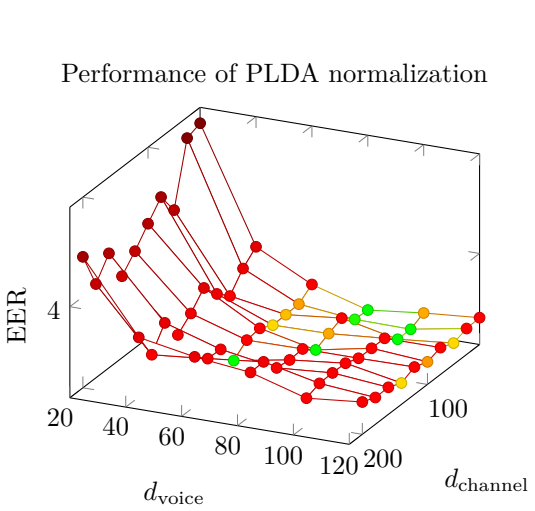


(a) EER of the cosine workflow, depending on LDA rank d_{lda}. The smallest EER of 1.71% is reached at $d_{\text{lda}} = 44$.

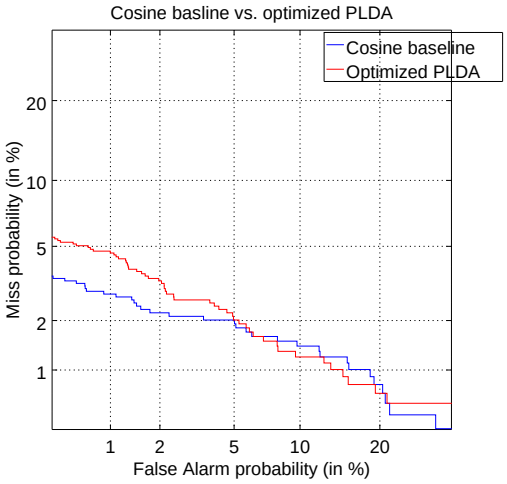


(b) DET curve of cosine scoring using EFR and LDA with $d_{\text{lda}} = 44$. As baseline, the curve from Fig. 5.2b is used.

Figure 5.4: Error rate of cosine scoring, using EFR and LDA as normalization steps, exploring the dependency of the EER on d_{lda}.



(a) The EER of PLDA normalization and scoring in relation to d_{voice} and d_{channel}. The smallest EER of 2.61% was obtained at $d_{\text{voice}} = 80, d_{\text{channel}} = 100$.



(b) DET curve for PLDA normalization with $d_{\text{voice}} = 80$ and $d_{\text{channel}} = 100$ (red). The curve from Fig.5.2b (blue), with EER of 1.71%, is used for comparison. Note that the PLDA workflow does not use additional normalization steps.

Figure 5.5: Error rate of PLDA normalization and scoring in relation to different values for d_{voice} and d_{channel}. While increasing values for $d_{channel}$ provide slight improvements, increasing values for d_{voice} decreases the EER noticeably.

5.2.4 Normalization for PLDA scoring

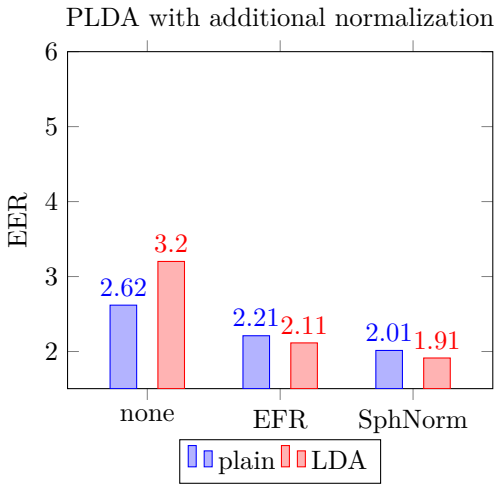
While PLDA was initially introduced to SID with an emphasis on its normalization capabilities, in this thesis it is treated as a scoring function with built-in normalization, since, as it is implemented in ALIZE, it is not possible to combine it with an other scoring function.

The normalization performance of PLDA is primarily influenced by the number of eigenvoices d_{voice} and eigenchannels d_{channel}. To find the suitable values for both parameters, the parameter space was evaluated in increments of 20, ranging from 20 to 200 for d_{voice} and 20 to 120 for d_{channel}. The EER for different values is visualized in Fig. 5.5a. As shown in Fig. 5.5b, plain PLDA does not improve the error rate over the cosine baseline for this data set.

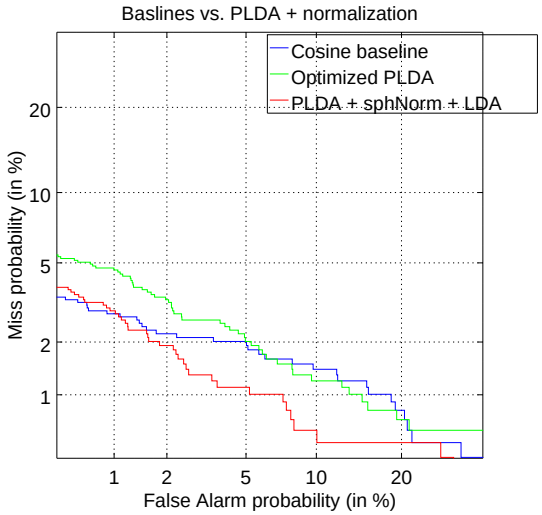
PLDA can be combined with additional normalization methods, namely EFR, spherical normalization and LDA. All additional processing takes place before PLDA normalization. Fig. 5.6a compares the performances of different combinations, using the same default parameters as for Fig. 5.3.

The best performance is achieved when using both spherical normalization and LDA. Since spherical normalization does not provide parameters to tune, only the LDA rank can be further optimized. The error rate, depending on rank, is displayed in Fig. 5.7.

While LDA provides an improvement regardless of rank, its performance is heavily fluctuating, with numerical instability occurring below $d_{\text{lda}} = 80$. The spike at rank 150 is considered an outlier since it is not supported by neighboring measurements, and the minimum at rank 80 is too close to numerical instability to be considered a safe parameter choice. The best supported minimum lies at $d_{\text{lda}} = 166$, providing an EER of 1.71%.

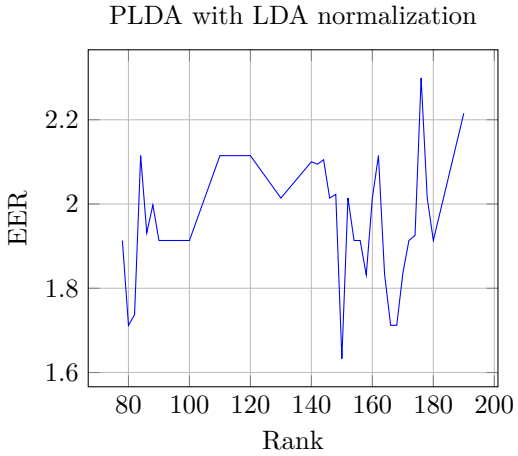


(a) Additional normalization steps for PLDA normalization and scoring. An LDA rank of $d_{\text{lda}} = 100$ was used.

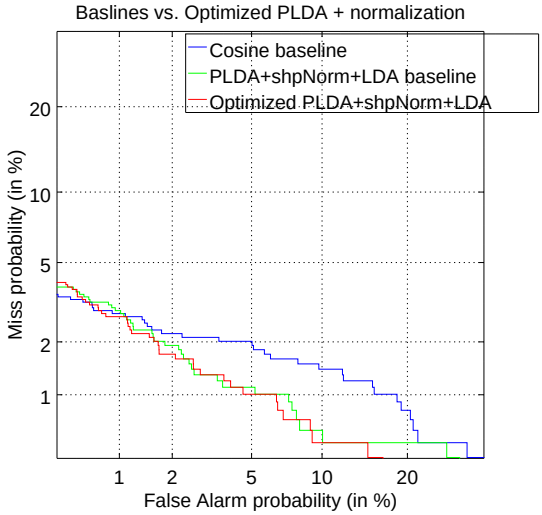


(b) DET curve of PLDA using spherical normalization and LDA (red), where an EER of 1.91% is achieved. As baselines, the cosine scoring from Fig. 5.2b (blue) and plain PLDA from Fig. 5.5b (green) are used.

Figure 5.6: Comparison of different normalization algorithms in addition to PLDA.

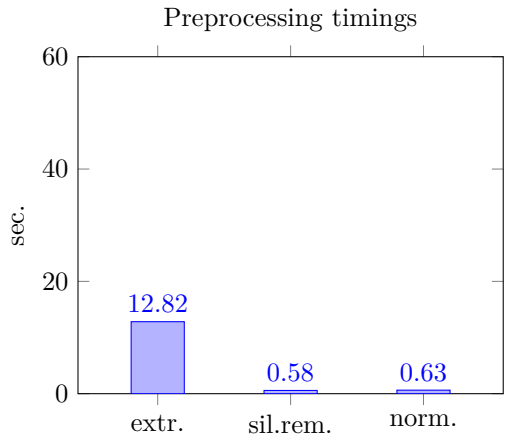


(a) Performance of LDA as secondary normalization to PLDA, depending on rank. While the smallest EER was achieved at $d_{\text{lda}} = 150$, it is considered an outlier. The chosen value of $d_{\text{lda}} = 166$ provides an EER of 1.76%.

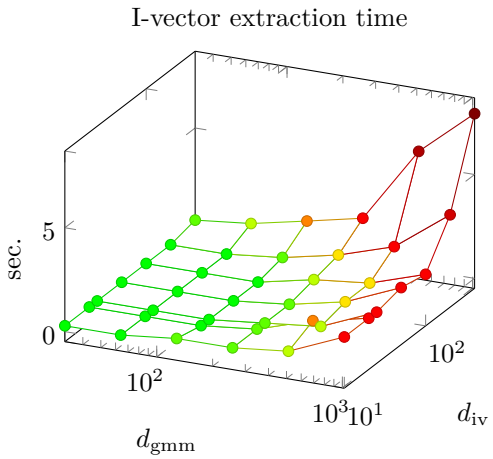


(b) DET curve of PLDA using spherical normalization and LDA with $d_{\text{lda}} = 166$ (red). As baselines, the curves from Fig. 5.6b and Fig. 5.2b are used.

Figure 5.7: Error rate for PLDA in conjunction with spherical normalization and different values for d_{lda}. The performance of different ranks for LDA fluctuates strongly. Optimizing the rank value produces only a slight improvement over the randomly chosen rank of 100.



(a) Timing of the low level preprocessing steps that generate the MFCCs: MFCC extraction, silence removal and MFCC normalization.



(b) The i-vector extraction timing, depending on the model sizes d_{gmm} and d_{iv}.

Figure 5.8: Feature extraction timing for one 2 min. utterance.

5.3 Execution speed evaluation

The execution speed is evaluated only for the online processing steps, for offline training only rough numbers are given since it is not important for this evaluation. A 2 min. mono wave file from the parliament test set is used as input, and all 235 speaker from the evaluation set are tested against. The normalization matrices are evaluated both precomputed and online. The timing experiments are done on a 2.4 GHz dual core machine with 4 GB ram.

Offline training was done using a 32 core cluster using 2 GHz clock frequency and 128 GB ram. Model training was done single threaded, since ALIZE has proved not to be thread safe although multithreading can be enabled. Training a GMM on the parliament training set took around 1 day for 32 mixtures and 8 days for 1024 mixtures. Training the TV matrix took 1 day for 10 dimensions and 6 days for 400 dimensions. Training was done using 6 iterations.

Fig.5.8a visualizes the execution times of the MFCC extraction and preprocessing steps. The i-vector extraction times, as shown in Fig. 5.8b, are below 1 sec. for small and mid-sized UBMs and TV matrices. Only at $d_{\text{gmm}} = 512$ and $d_{\text{iv}} = 100$ the extraction process of 2 min. of audio takes more than 1 second to terminate. At the highest dimensions, requiring about 9 seconds, it comes close to the MFCC extraction duration. This means that execution speed is no deciding factor when comparing the cosine-based and PLDA-based workflows.

The processing speed of the normalization and scoring is evaluated for both the cosine-based and PLDA-based configurations. The implementation of the normalization algorithms, provided by ALIZE, allows for two different modes of operation: training the normalization matrices online, or using precomputed matrices. Both possibilities are compared in Fig. 5.9. It can be seen that using online normalization, PLDA takes about three times longer than cosine scoring. When using precomputed normalization matrices, they are about equal in execution speed.

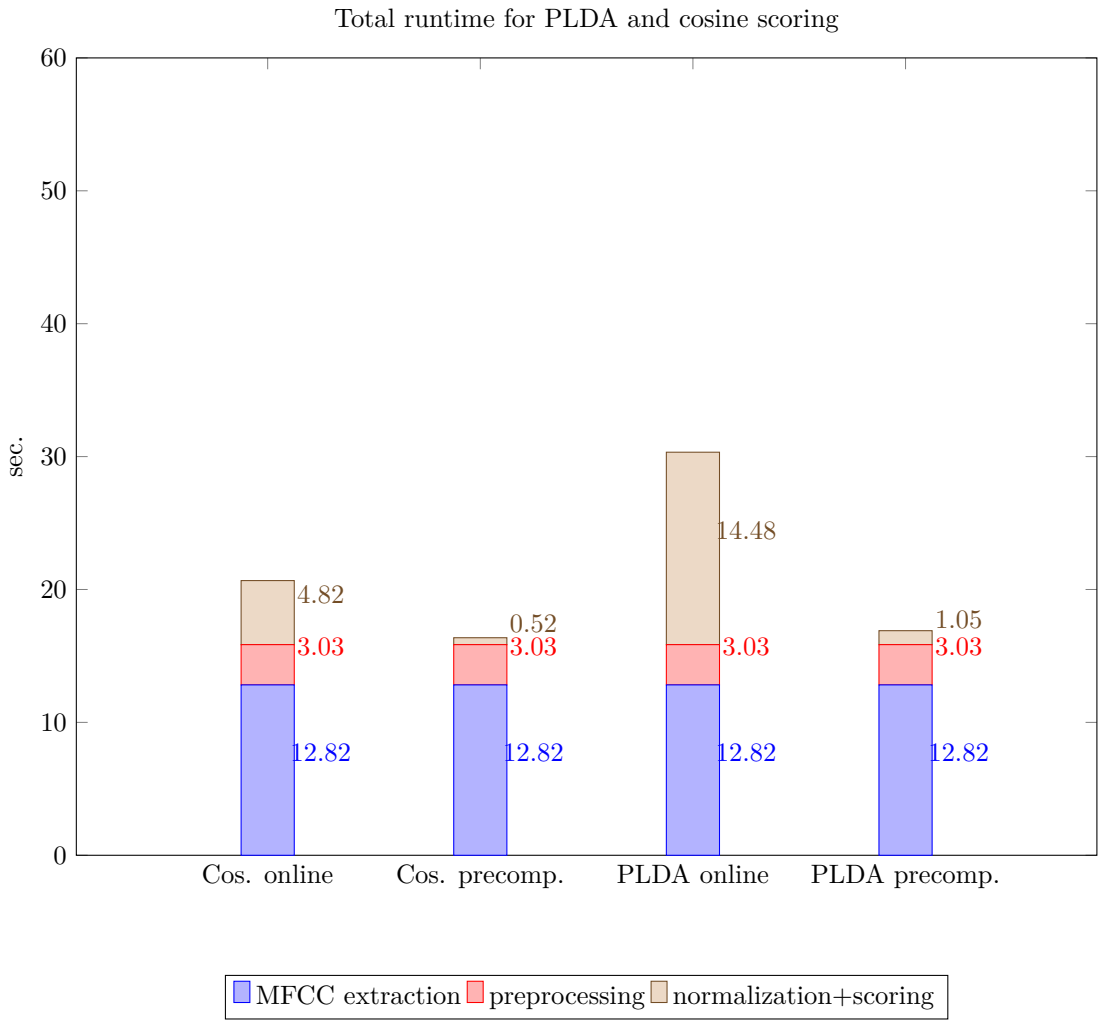
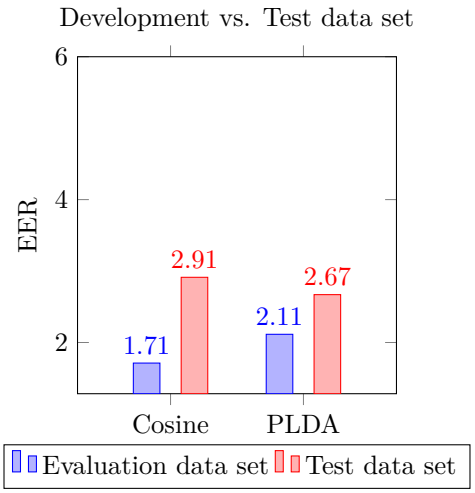
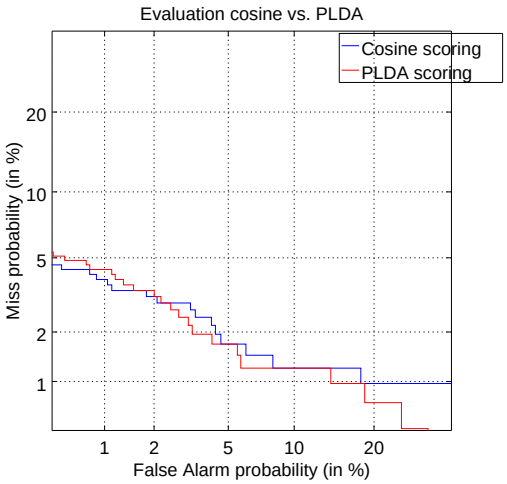


Figure 5.9: Runtime comparison of cosine-based and PLDA-based workflow configurations, both for precomputed normalization matrices and online computed matrices. Precomputed means that the normalization matrices are computed offline using a training set. Online mode performs the same operation, but it is done for each trial, which allows for updating the training set on the fly. The measurements are an averaged timing for one 2 min. utterance.



(a) Comparison of the EER for the PLDA and cosine workflow configurations, evaluated on the evaluation set and the test data set.



(b) DET curves for the cosine and PLDA configurations, using the test data set.

Figure 5.10: Evaluating the cosine based and PLDA based workflows using the test data set. While both workflows perform worse on the test set, PLDA proved to be more consistent when facing different data.

5.4 Configuration selection

While the underlying models are trained on the training data set, the workflow configurations are manually optimized on the test data set. The final decision on which configuration to use is performed on an independent test data set, so overfitting to the evaluation set is avoided.

As shown in Fig.5.10a, the cosine configuration performs noticeably worse on the test set compared to the evaluation set. PLDA, on the other hand, shows a less dramatic increase in error rate. The PLDA-based configuration shows more stability over the cosine configuration, which is probably overfitted to the test data set. Thus, the PLDA workflow is selected as final configuration.

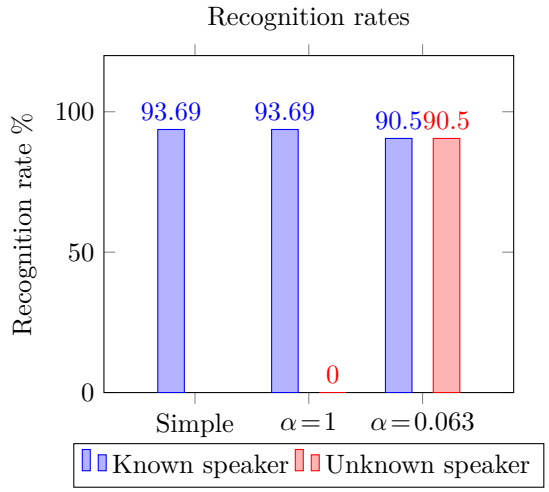
5.5 Decisionmaking

Both closed-set and open-set decisionmaking, as discussed in Chapter 4.5.3, are tested on the final workflow. Closed-set decisionmaking, using the simple highest-score selection method, performs reasonably well by providing a recognition rate of 93.689% (classifying 93.689% of the utterances correctly). For open-set decisionmaking, two cases are evaluated: classifying utterances that belong to a known speaker, and classifying utterances that belong to an unknown speaker. To emulate the unknown case, the correct speaker model is removed from the set of known speakers for each utterance. The unknown classification is performed by using an average speaker model y_O.

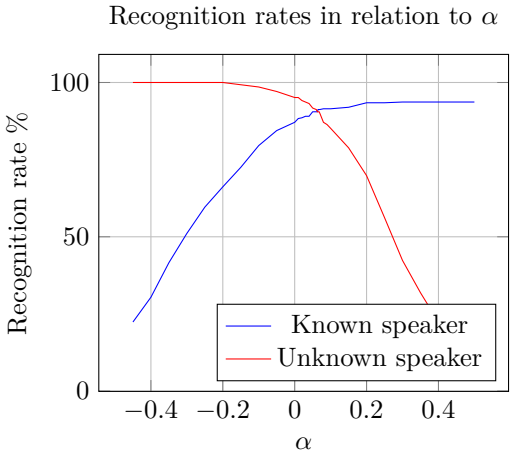
When using the unscaled model y_O ($\alpha = 1$), the system performs overly optimistic, as shown in Fig. 5.11a: in the first case, 93.689% of the known utterances are classified correctly, the rest being classified as unknown. No utterance is assigned to a wrong speaker. When removing the correct speaker model for each utterance, no utterance is recognized as unknown but gets assigned a wrong speaker. To balance this discrepancy in recognition rates, a value for α was found which balances the recognition rates for known and unknown speakers, as shown in Fig. 5.11b. $\alpha = 0.063$ causes the recognition rates to be come equal and leads to an overall recognition rate of 90.511%, where 3.163% of the known utterances are assigned a wrong speaker and the test are wrongly classified as unknown.

5.6 Comparison to iFinder

The goals of the new workflow are to improve the recognition rate and online execution time of the current implementation. To assess the current performance, an instance of the iFinder speaker classifier was installed on the test machine and evaluated using the parliament test set. Fig. 5.12

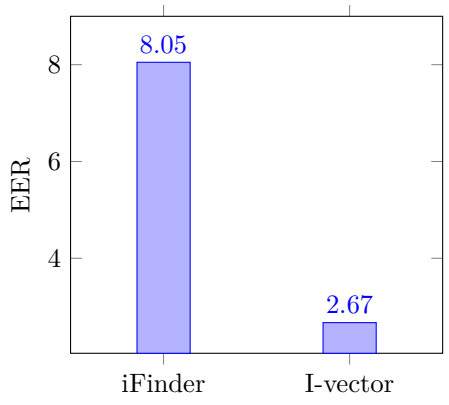


(a) Comparing the recognition rate for the closed-set assumption, using the simple decision scheme, and the open-set scheme for $\alpha = 1$ and $\alpha = 0.063$.

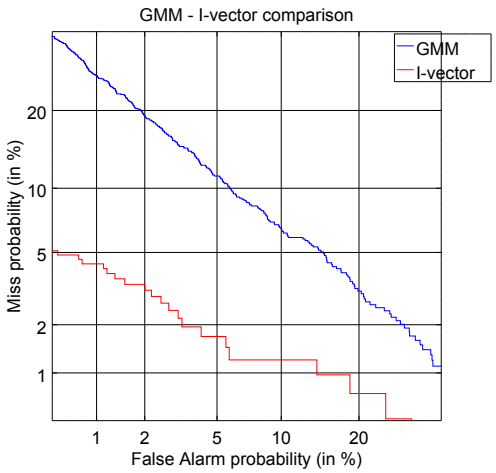


(b) Recognition rates for open-set decisionmaking, showing the recognition percentage for known and unknown utterances, depending on α. The curves intersect at $\alpha = 0.063$, where the classification error is equally distributed among those two cases.

Figure 5.11: Evaluation of decisionmaking schemes, using the scores of the PLDA system. Simple best-score selection is used for closed-set assumption and confidence-based classification is used for the open.sett assumption. α is a data-dependent factor which balances recognition rates for known and unknown utterances.



(a) EER of the GMM-based iFinder implementation and the new I-vector based workflow.



(b) DET curves for the old and new workflow.

Figure 5.12: Error rate comparison of the GMM-based and I-vector-based classifiers.

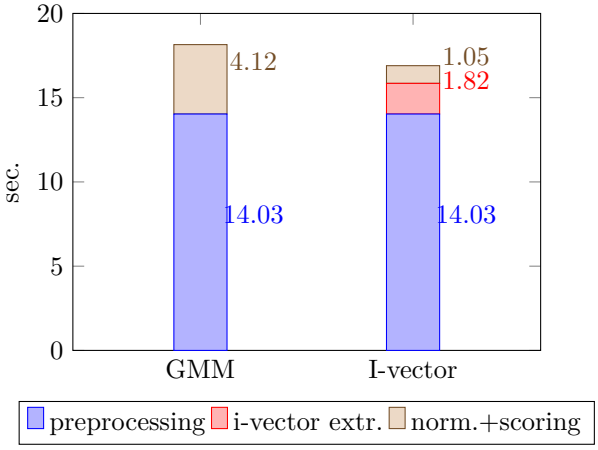
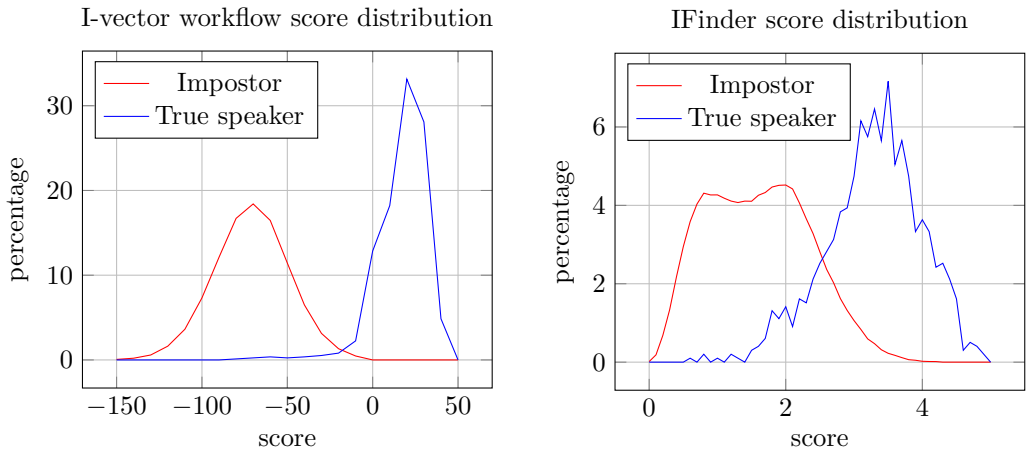


Figure 5.13: Comparing runtime of the GMM-based workflow and the precomputed PLDA workflow for a 2 min. recording and all 235 speaker models. The preprocessing steps, shown in Fig. 5.8a are merged for clarity. The GMM-based iFinder implementation takes 18.15 sec. total to analyze the test file while the PLDA workflow takes 16.9 sec. total.



(a) Normalized score distributions of the i-vector based workflow.

(b) Normalized score distributions of the iFinder implementation.

Figure 5.14: Comparison between the normalized score distributions in the new i-vector based workflow and the iFinder reference workflow. The False curve represents the scores given to impostors and the True curve represents the scores of genuine speakers.

visualizes the error rate. The new workflow clearly outperforms the iFinder workflow. Fig. 5.14 visualizes the score distributions of the new PLDA-based system and the iFinder implementation. The graphs of the new system show a reduced variance in the distributions, compared to the old system, which provides for better separability during decisionmaking.

To evaluate the execution time, the duration of the test is taken and divided by test performed on each input file (one input file tested against all 235 speakers). Fig. 5.13 compares the workflow specific timings. While the i-vector based workflow is faster than the iFinder implementation, it is only a slight improvement.

The main gain of the new PLDA-based workflow is a substantial decrease in error rate and a small decrease in execution speed. Thus, both system improvement goals are reached.

5.7 Discussion

While the overall results are in line with recent publications, in terms of reaching a similar EER for domain specific data set as [GRM14] and confirming the use of PLDA as a well-performing and robust normalization and scoring function, some findings give interesting insights for practical implementations.

5.7.1 UBM and TV matrix

Fig. 5.2a shows the impact of varying numbers of mixtures in the UBM and sizes of the TV space. The number of mixtures have a low impact on the recognition rate, compared to the number of TV dimensions. Since larger numbers of mixtures still perform better than smaller numbers, it is desirable to use large UBMs unless the i-vector extraction speed is a critical factor. In regards to execution speed, the i-vector extraction is actually more expensive than the normalization and comparisons, as shown in Fig. 5.9. Since the extraction speed is equally affected by both UBM and TV matrix size, as shown in Fig. 5.8b, a tradeoff in favor of execution speed can be made with only small losses in recognition rate.

5.7.2 Cosine scoring

Fig. 5.3 shows several interesting results regarding the performance of normalization for cosine scoring. When comparing EFR and SN, EFR shows better performance than SN in every combination, even in combination with LDA. This is unexpected since SN is an extension of EFR which is explicitly modified to work in conjunction with LDA. The linear, non-spherical within-class covariance matrix seems to have a stronger impact on the angular distribution of the i-vectors than the spherical covariance.

A second observation is the performance of WCCN, which underperforms in almost all combinations. Paired with SN, WCCN does not provide for a noticeable impact. The performance of the combination of WCCN and EFR is especially surprising because both methods perform within-class covariance normalization. WCCN seems to negate the normalization effects of Σ_w compensation and length normalization performed by EFR with its linear projection. The same effect is noticeable in combination with LDA.

LDA shows the best normalization performance, which is to be expected since it is the most sophisticated algorithm, balancing Σ_w and Σ_b. When comparing the LDA ranks, as shown in Fig.5.4a, a clear optimum of $d_{\text{lda}} = 44$ can be found. This shows that the speaker-related information can be well represented in a small-dimensional subspace, which is in line with the underlying assumption on the separability of speaker related information. It also shows that between-class covariance normalization helps in making the speakers more separable.

5.7.3 PDLA scoring

The normalization performance of PLDA, as shown in Fig.5.5 of different ranks for speaker and channel information, has shown to be mainly depending on the speaker rank. The optimal speaker and channel rank do not differ much. In light of the assumption of low-dimensional, separable subspaces of speaker and channel information, it can be argued that both spaces are similarly large.

Interestingly, when comparing the LDA rank of 44 in the cosine workflow with the rank of 166 in the PLDA workflow, the speaker space used in the latter is more than twice as large. This,

together with the observation of the seemingly irregular performance across rank values, can be interpreted as a lack of information to be removable by LDA. As PLDA itself is an extension of LDA, both methods are able to remove similar information patterns.

5.8 Speaker set extension

It is desirable to improve the identification system by extending the set of known speaker or improving existing speakers models with newly gathered utterances. In the context of the News-Stream 3.0 project, it is also desirable to apply these improvements without shutting down the production system.

As described in Chapter 4.8, ALIZE uses configuration files to define the trials and to map the speaker identifiers to the paths of the files which contain the model data. These configuration files are reloaded for each incoming utterance, so it is possible to update both the current trial file (which lists all speaker model files to compare against) and the speaker models themselves in between runs.

The speaker models can be updated by allowing the user to select speech segments in the user interface and assign them to an existing speaker. The speaker model can then be updated by including the new utterance into the averaging process. The same way, new speakers can be generated.

Chapter 6

Conclusion

The central challenge for any SID workflow is to distinguish speaker related information from unwanted information in order to provide robust classification. Finding a suitable procedure to exclude or lessen the impact of unwanted information cannot be done in general, but must be tailored to the data at hand. In this thesis an i-vector based speaker identification workflow was developed in the application context of the News-Stream 3.0 project. A domain specific data set was used to assemble a SID system with a configuration optimized for the given data. The new workflow outperforms the previous implementation in both recognition rate and execution speed. Combinations of a set of algorithms provided by the ALIZE toolkit, as well as important parameters, have been explored for each processing step:

The number of mixtures of the UBM and the rank of the total variability matrix have been investigated for their impact on recognition quality and execution speed. While the UBM size has no strong influence on the error rate, a total variability rank of at least 100 dimensions provided error rates of 3% or lower. The i-vector extraction speed depends approximately quadratic on both values.

Cosine scoring and PLDA have been used as scoring functions, and normalization configurations and parameter values have been optimized for each of them using the domain specific evaluation data. In both cases, EFR and spherical normalization have been used as normalization procedures, while LDA (and WCCN in case of cosine scoring) have been applied as additional processing steps. Cosine scoring, combined with EFR and LDA, showed a clear optimum in the evaluated range for LDA rank and provided an EER of 1.71% on the development set. PLDA itself showed good results for $d_{\text{voice}} > 60$. In combination with spherical normalization and LDA, an EER of 2.11% was reached on the development set. For different d_{LDA}, although providing an overall improvement, the error rate varied seemingly at random which shows that most channel information was already captured by PLDA.

When compared on a test set, the PLDA configuration was chosen over the cosine-based workflow, since it showed more stable performance when evaluated on the testing data set. It achieves an EER of 2.67% on the test set, and takes 8 sec. to classify an i-vector. The cosine-based configuration, while having a lower error rate on the development set, showed a strong decrease in performance on the test set which indicates overfitting. Finally a scheme for updating the models and introducing new speaker models into the running workflow was proposed, which enables the running system to adapt to the data it is analyzing.

Both closed-set and open-set decisionmaking have been evaluated on the PLDA workflow. The closed-set classification achieved a recognition rate of 93.69%, while the open-set classification achieved a recognition rate of 90.51%.

Overfitting has shown to be an important consideration when developing an optimized classification workflow. Future work should look at how to prevent overfitting when only a limited amount of data is available. Randomly generated subsets of the evaluation data could be used for parameter evaluation, aiming for the lowest error rate and minimal variance over the test sets.

It would also be interesting to look deeper into the unexpected performance of combinations of normalization methods. By evaluating different projections for the within-class covariance and between-class covariance, more insight into the actual distribution of the data and how to counteract it may be gained.

Bibliography

[AB13] Hagai Aronowitz and Oren Barkan. On leveraging conversational data for building a text dependent speaker verification system. In *INTERSPEECH*, pages 2470–2473, 2013.

[AKC+02] Walter D Andrews, Mary A Kohler, Joseph P Campbell, John J Godfrey, and Jaime Hernández-Cordero. Gender-dependent phonetic refraction for speaker recognition. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 1, pages I–149. IEEE, 2002.

[AWH05] Xavier Anguera, C Woofers, and Javier Hernando. Speaker diarization for multi-party meetings using acoustic fusion. In *Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on*, pages 426–431. IEEE, 2005.

[BLM+12] Pierre-Michel Bousquet, Anthony Larcher, Driss Matrouf, Jean-François Bonastre, and Oldrich Plchot. Variance-spectra based normalization for i-vector standard and probabilistic linear discriminant analysis. In *Speaker and Language Recognition Workshop (IEEE Odyssey)*, 2012.

[BMB11] Pierre-Michel Bousquet, Driss Matrouf, and Jean-François Bonastre. Intersession compensation and scoring methods in the i-vectors space for speaker recognition. In *INTERSPEECH*, pages 485–488, 2011.

[BPT+12] Hervé Bredin, Johann Poignant, Makarand Tapaswi, Guillaume Fortier, Viet Bac Le, Thibault Napoleon, Hua Gao, Claude Barras, Sophie Rosset, Laurent Besacier, et al. Fusion of speech, faces and text for person identification in tv broadcast. In *Computer Vision–ECCV 2012. Workshops and Demonstrations*, pages 385–394. Springer, 2012.

[BSS+97] Adit Benyassine, Eyal Shlomot, Huan-Yu Su, Dominique Massaloux, Claude Lamblin, and Jean-Pierre Petit. Itu-t recommendation g. 729 annex b: a silence compression scheme for use with g. 729 optimized for v. 70 digital simultaneous voice and data applications. *Communications Magazine, IEEE*, 35(9):64–73, 1997.

[BTHVF08] Kofi Boakye, Beatriz Trueba-Hornero, Oriol Vinyals, and Gerald Friedland. Overlapped speech detection for improved speaker diarization in multiparty meetings. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 4353–4356. IEEE, 2008.

[CG10] Yihua Chen and Maya Gupta. Em demystified: An expectation-maximization tutorial. *UWEE Technical Report Number UWEETR-2010-0002*, 2010.

[CVR05] Mauro Cettolo, Michele Vescovi, and Romeo Rizzi. Evaluation of bic-based algorithms for audio segmentation. *Computer Speech & Language*, 19(2):147–170, 2005.

[CWF10] Shih-Sian Cheng, Hsin-Min Wang, and Hsin-Chia Fu. Bic-based speaker segmentation using divide-and-conquer strategies with application to speaker diarization. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(1):141–157, 2010.

[CY13] Li Chen and Yingchun Yang. Emotional speaker recognition based on i-vector through atom aligned sparse representation. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7760–7764. IEEE, 2013.

[Deh09] Najim Dehak. *Discriminative and generative approaches for long-and short-term speaker characteristics modeling: application to speaker verification.* Ecole de Technologie Superieure (Canada), 2009.

[DKD+11] Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(4):788–798, 2011.

[DM80] Steven B Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 28(4):357–366, 1980.

[EKLMP12] Elie El Khoury, Antoine Laurent, Sylvain Meignier, and Simon Petitrenaud. Combining transcription-based and acoustic-based speaker identifications for broadcast news. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 4377–4380. IEEE, 2012.

[Fan68] Gunnar Fant. Analysis and synthesis of speech processes. *Manual of phonetics*, 2:173–277, 1968.

[Faw06] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[FC14] Corinne Fredouille and Delphine Charlet. Analysis of i-vector framework for speaker identification in tv-shows. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[FM37] Harvey Fletcher and Wilden A Munson. Relation between loudness and masking. *The Journal of the Acoustical Society of America*, 9(1):78–78, 1937.

[GHM92] John J Godfrey, Edward C Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, volume 1, pages 517–520. IEEE, 1992.

[GLA02] Jean-Luc Gauvain, Lori Lamel, and Gilles Adda. The limsi broadcast news transcription system. *Speech communication*, 37(1):89–108, 2002.

[GMM+14] Ondrej Glembek, Jiaxin Ma, Pavel Matejka, Bing Zhang, Oldrich Plchot, Lukas Burget, and Spyros Matsoukas. Domain adaptation via within-class covariance correction in i-vector based speaker recognition systems. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 4032–4036. IEEE, 2014.

[GRM14] Daniel Garcia-Romero and Alan McCree. Supervised domain adaptation for i-vector based speaker recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 4047–4051. IEEE, 2014.

[Gro02] NIST Multimodal Information Group. Nist rt 2002 evaluation plan. `http://www.itl.nist.gov/iad/mig/tests/rt/2002/docs/rt02_eval_plan_v3.pdf`, 2002. Accessed: 2015-10-31.

[Gro05] NIST Multimodal Information Group. Nist rt 2005 evaluation plan. `http://www.itl.nist.gov/iad/mig/tests/rt/2005-spring/rt05s-meeting-eval-plan-V1.pdf`, 2005. Accessed: 2015-10-31.

[Gro08] NIST Multimodal Information Group. Nist sre 2008 evaluation plan. `http://www.itl.nist.gov/iad/mig/tests/sre/2008/sre08_evalplan_release4.pdf`, 2008. Accessed: 2015-10-18.

[Gro12] NIST Multimodal Information Group. Nist sre 2012 evaluation plan. `http://www.nist.gov/itl/iad/mig/upload/NIST_SRE12_evalplan-v17-r1.pdf`, 2012. Accessed: 2015-10-31.

[Her90] Hynek Hermansky. Perceptual linear predictive (plp) analysis of speech. *the Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

[HHD01] HG Hirsch, K Hellwig, and Stefan Dobler. Speech recognition at multiple sampling rates. In *INTERSPEECH*, pages 1837–1840, 2001.

[HKS06] Andrew O Hatch, Sachin S Kajarekar, and Andreas Stolcke. Within-class covariance normalization for svm-based speaker recognition. In *Interspeech*, 2006.

[Jam15] Josh James. Data never sleeps 3.0. `https://www.domo.com/blog/2015/08/data-never-sleeps-3-0/`, 2015. Accessed: 2015-10-25.

[KAM+13] Tomi Kinnunen, Md Jahangir Alam, Pavel Matejka, Patrick Kenny, Jan Cernockỳ, and Douglas D O'Shaughnessy. Frequency warping and robust speaker verification: a comparison of alternative mel-scale representations. In *INTERSPEECH*, pages 3122–3126, 2013.

[KBD05] Patrick Kenny, Gilles Boulianne, and Pierre Dumouchel. Eigenvoice modeling with sparse training data. *Speech and Audio Processing, IEEE Transactions on*, 13(3):345–354, 2005.

[KBOD05] Patrick Kenny, Gilles Boulianne, Pierre Ouellet, and Pierre Dumouchel. Factor analysis simplified. In *ICASSP (1)*, pages 637–640. Citeseer, 2005.

[KBOD07] Patrick Kenny, Gilles Boulianne, Pierre Ouellet, and Pierre Dumouchel. Joint factor analysis versus eigenchannels in speaker recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(4):1435–1447, 2007.

[KD04] Patrick Kenny and Pierre Dumouchel. Experiments in speaker verification using factor analysis likelihood ratios. In *ODYSSEY04-The Speaker and Language Recognition Workshop*, 2004.

[KDS+14] Ahilan Kanagasundaram, David Dean, Sridha Sridharan, Mitchell McLaren, and Robbie Vogt. I-vector based speaker recognition using advanced channel compensation techniques. *Computer Speech & Language*, 28(1):121–140, 2014.

[KEWP11] Jonathan Kola, Carol Espy-Wilson, and Tarun Pruthi. Voice activity detection. *MERIT BIEN*, pages 1–6, 2011.

[KL10] Tomi Kinnunen and Haizhou Li. An overview of text-independent speaker recognition: From features to supervectors. *Speech communication*, 52(1):12–40, 2010.

[KOD+08] Patrick Kenny, Pierre Ouellet, Najim Dehak, Vishwa Gupta, and Pierre Dumouchel. A study of interspeaker variability in speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(5):980–988, 2008.

[LBF+13] Anthony Larcher, Jean-François Bonastre, Benoit GB Fauve, Kong-Aik Lee, Christophe Lévy, Haizhou Li, John SD Mason, and Jean-Yves Parfait. Alize 3.0-open source toolkit for state-of-the-art speaker recognition. In *INTERSPEECH*, pages 2768–2772, 2013.

[LH89] Kai-Fu Lee and Hsiao-Wuen Hon. Speaker-independent phone recognition using hidden markov models. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(11):1641–1648, 1989.

[LLML14] Anthony Larcher, Kong Aik Lee, Bin Ma, and Haizhou Li. Text-dependent speaker verification: Classifiers, databases and rsr2015. *Speech Communication*, 60:56–77, 2014.

[LR14] Liang Lu and Steve Renals. Probabilistic linear discriminant analysis for acoustic modelling. *Signal Processing Letters, IEEE*, 2014.

[LVH+11] A Lawson, Pavel Vabishchevich, M Huggins, P Ardis, Brandon Battles, and A Stauffer. Survey and evaluation of acoustic features for speaker recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5444–5447. IEEE, 2011.

[LZTZ02] Qi Li, Jinsong Zheng, Augustine Tsai, and Qiru Zhou. Robust endpoint detection and energy normalization for real-time speech and speaker recognition. *Speech and Audio Processing, IEEE Transactions on*, 10(3):146–157, 2002.

[MB06] Kamran Mustafa and Ian C Bruce. Robust formant tracking for continuous speech with speaker variability. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(2):435–444, 2006.

[MBE10] Lindasalwa Muda, Mumtaj Begam, and I Elamvazuthi. Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques. *arXiv preprint arXiv:1003.4083*, 2010.

[MBE+12] Xavier Anguera Miro, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. Speaker diarization: A review of recent research. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(2):356–370, 2012.

[MDK+97] Alvin Martin, George Doddington, Terri Kamm, Mark Ordowski, and Mark Przybocki. The det curve in assessment of detection task performance. Technical report, DTIC Document, 1997.

[Mir07] Xavier Anguera Miro. *Robust speaker diarization for meetings*. Universitat Politècnica de Catalunya, 2007.

[MJVTG98] Pedro J Moreno, Christopher F Joerg, Jean-Manuel Van Thong, and Oren Glickman. A recursive algorithm for the forced alignment of very long audio segments. In *ICSLP*, volume 98, pages 2711–2714, 1998.

[new14] newsstreamproject.org. Neue big-data-infrastruktur fÃ¼r journalisten. `http://newsstreamproject.org/wp-content/uploads/2014/12/Presseinformation.pdf`, 2014. Accessed: 2015-10-18.

[PBL+12] Johann Poignant, Hervé Bredin, Viet-Bac Le, Laurent Besacier, Claude Barras, and Georges Quénot. Unsupervised speaker identification using overlaid texts in tv broadcast. In *Interspeech 2012-Conference of the International Speech Communication Association*, page 4p, 2012.

[PE07] Simon JD Prince and James H Elder. Probabilistic linear discriminant analysis for inferences about identity. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[PGB+11] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB.

[RAHK14] Padmanabhan Rajan, Anton Afanasyev, Ville Hautamäki, and Tomi Kinnunen. From single to multiple enrollment i-vectors: Practical plda scoring variants for speaker verification. *Digital Signal Processing*, 31:93–101, 2014.

[RCC+03] Douglas Reynolds, Joe Campbell, Bill Campbell, Bob Dunn, Terry Gleason, Doug Jones, Tom Quatieri, Carl Quillen, Doug Sturim, and Pedro Torres-Carrasquillo. Beyond cepstra: exploiting high-level information in speaker recognition. In *Proceedings of the Workshop on Multimodal User Authentication*, pages 223–229, 2003.

[RDG+13] Mickael Rouvier, Grégor Dupuy, Paul Gay, Elie Khoury, Teva Merlin, and Sylvain Meignier. An open-source state-of-the-art toolbox for broadcast news diarization. Technical report, Idiap, 2013.

[Rey97] Douglas A Reynolds. Comparison of background normalization methods for text-independent speaker verification. In *Eurospeech*, 1997.

[RJ11] Henning Reetz and Allard Jongman. *Phonetics: Transcription, production, acoustics, and perception*, volume 34. John Wiley & Sons, 2011.

[RQD00] Douglas A Reynolds, Thomas F Quatieri, and Robert B Dunn. Speaker verification using adapted gaussian mixture models. *Digital signal processing*, 10(1):19–41, 2000.

[RR+95] Douglas Reynolds, Richard C Rose, et al. Robust text-independent speaker identification using gaussian mixture speaker models. *Speech and Audio Processing, IEEE Transactions on*, 3(1):72–83, 1995.

[SJRS97] Matthew A Siegler, Uday Jain, Bhiksha Raj, and Richard M Stern. Automatic segmentation, classification and clustering of broadcast news audio. In *Proc. DARPA speech recognition workshop*, volume 1997, 1997.

[SMA07] Michael E Schuckers, Yordan Minev, and Andy Adler. Curvewise det confidence regions and pointwise eer confidence intervals using radial sweep methodology. In *Advances in Biometrics*, pages 376–385. Springer, 2007.

[SSR+15] Hasim Sak, Andrew Senior, Kanishka Rao, Ozan Irsoy, Alex Graves, Françoise Beaufays, and Johan Schalkwyk. Learning acoustic frame labeling for speech recognition with recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4280–4284. IEEE, 2015.

[Ste57] Stanley S Stevens. On the psychophysical law. *Psychological review*, 64(3):153, 1957.

[SVN37] Stanley Smith Stevens, John Volkmann, and Edwin B Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

[TG99] Alain Tritschler and Ramesh A Gopinath. Improved speaker segmentation and segments clustering using the bayesian information criterion. In *Eurospeech*, volume 99, pages 679–682, 1999.

[TO00] S Gökhun Tanyer and Hamza Ozer. Voice activity detection in nonstationary noise. *IEEE Transactions on speech and audio processing*, 8(4):478–482, 2000.

[TSH00] Remco Teunen, Ben Shahshahani, and Larry P Heck. A model-based transformational approach to robust speaker recognition. In *INTERSPEECH*, pages 495–498, 2000.

[YL08] Jiahong Yuan and Mark Liberman. Speaker identification on the scotus corpus. *Journal of the Acoustical Society of America*, 123(5):3878, 2008.

[YYDS11] Dongwen Ying, Yonghong Yan, Jianwu Dang, and Frank K Soong. Voice activity detection based on an unsupervised learning framework. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(8):2624–2633, 2011.

[ZBMG05] Xuan Zhu, Claude Barras, Sylvain Meignier, and Jean-Luc Gauvain. Combining speaker identification and bic for speaker diarization. In *INTERSPEECH*, volume 5, pages 2441–2444, 2005.

[ZH05] Bowen Zhou and John HL Hansen. Efficient audio stream segmentation via the combined t 2 statistic and bayesian information criterion. *Speech and Audio Processing, IEEE Transactions on*, 13(4):467–474, 2005.