Discriminative Classifiers for Speaker Recognition

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

von Dipl.-Ing.(FH) Marcel Katz
geb. am 01. Oktober 1975 in Bonn

genehmigt durch die
Fakultät für Elektrotechnik und Informationstechnik
der Otto-von-Guericke-Universität Magdeburg

Gutachter: **Prof. Dr. rer.nat. Andreas Wendemuth**
Prof. Dr.-Ing. Tim Fingscheidt

Promotionskolloquium am 05. März 2008

To my parents
and my wife

Zusammenfassung

Bedingt durch den wachsenden Bedarf an Sicherheitsanwendungen ist die Erkennung einer Person anhand der Stimme zu einem Interessenschwerpunkt im Bereich der Authentifikationsverfahren geworden. Traditionell basieren Verfahren zur Sprechermodellierung in der Sprechererkennung schwerpunktmäßig auf generativen Klassifikationsmethoden, wie beispielsweise den Gauss'schen Mischverteilungs-Modellen (GMMs). In der jüngeren Forschung werden jedoch auch alternative Klassifikationsmethoden wie zum Beispiel die Support Vector Machine (SVM) erfolgreich in verschiedenen Gebieten der Mustererkennung eingesetzt. Begründet in der statistischen Lerntheorie zeichnen sich diese alternativen Klassifizierer durch eine sehr gute Generalisierungsfähigkeit aus. Aufgrund dessen werden diese so genannten diskriminativen Methoden auch gezielt im Bereich der Sprechererkennung als vielversprechender Ansatz zur Verbesserung der Erkennungsleistung diskutiert.

Im Rahmen dieser Forschung liegt der Schwerpunkt der vorliegenden Arbeit in der Entwicklung und Integration verschiedener diskriminativer Klassifizierer in den Bereich der Sprechererkennung.

Als eine Alternative zur SVM wird in dieser Arbeit die Sparse Kernel Logistic Regression (SKLR), eine ausgedünnte, nichtlineare Erweiterung der bekannten logistischen Regression, vorgestellt. Im Gegensatz zur SVM modelliert die SKLR direkt die a posteriori Wahrscheinlichkeit einer Klassenzugehörigkeit und liefert daher auch auf natürliche Weise eine Wahrscheinlichkeitsausgabe. Zu diesem Zweck wird eine neue Sprechererkennungsumgebung konzipiert und implementiert, die zwei unterschiedliche Erkennungsansätze enthält, einen für kleine sowie einen für große Lernstichproben.

Im ersten Erkennungsansatz werden diskriminative Klassifizierer direkt auf den Merkmalsvektoren der parametrisierten Sprache angewendet und es wird gezeigt, dass sowohl die SVM als auch die SKLR der traditionellen GMM-Methode in der Erkennungsleistung überlegen sind.

Im zweiten Ansatz wird ein, dem neuesten Stand der Technik entsprechendes, Sprechererkennungssystem für große Lernstichproben konzipiert, welches Gauss'sche Mischverteilungen mit diskriminativen Klassifizierern, insbesondere der SKLR, kombiniert.

Darüber hinaus werden in dieser Arbeit verschiedene Methoden der Merkmalsextraktion aus Sprachsignalen für die Sprechererkennung auf großen Lernstichproben untersucht. Es wird gezeigt, dass der Einsatz von Fusionsansätzen die mehrere Teilsysteme kombinieren zu einer signifikanten Verbesserung der Erkennungsleistung gegenüber dem Einsatz von einzelnen Teilsystemen führen.

Alle präsentierten Verfahren werden auf international anerkannten Sprachdatenbanken getestet und wurden in einschlägigen internationalen Medien publiziert. Im Vergleich mit anderen Arbeiten auf dem aktuellen Forschungsstand erreichen unsere vorgestellten Sprechererkennungssysteme gleichwertige oder bessere Erkennungsleistungen.

Abstract

Due to the growing need for security applications speaker recognition as the biometric task of authenticating a claimant by voice has currently become a focus of interest. Traditionally approaches in the area of speaker recognition were mainly based on generative classifiers like Gaussian Mixture Models (GMMs). However, more recently other classifiers like Support Vector Machines (SVMs) have been successfully applied to several fields of pattern recognition. These discriminative classifiers which are theoretically derived from statistical learning theory obtain a high generalization ability. Therefore these so called discriminative methods have also been discussed as a promising approach to specifically improve performance of speaker recognition systems.

Following this train of thought, this work focuses on the development and integration of different discriminative classifiers into the field of speaker recognition. As an alternative to the SVM we present the Sparse Kernel Logistic Regression (SKLR), a sparse non-linear expansion of the well known Logistic Regression. In contrast to Support Vector Machines the SKLR directly models the posterior probability of class membership and therefore naturally provides a probability output. For this reason a new speaker recognition environment is designed and implemented which includes two different recognition approaches, one for limited and one for large (the so called extended) training data.

In the first recognition approach the discriminative classifiers are applied directly on feature vectors from parameterized speech frames and it is shown that both, SVM as well as SKLR outperform traditional GMM methods.

In the second approach a state-of-the-art speaker recognition system for large amount of training data is designed that combines Gaussian Mixture Models (GMM) with discriminative classifiers and integrates the SKLR into this system.

Furthermore, we investigate different feature extraction methods for speaker recognition on large amount of training data. It is shown that the application of fusion schemes which combine these subsystems yield a significant improvement of the recognition performance in comparison to the application of single subsystems.

All presented approaches are evaluated on internationally recognized corpora and were published in appropriate international media. The comparison of our speaker recognition systems with other state-of-the-art systems revealed equal or significantly better recognition performance.

Acknowledgments

I would like to express my sincere gratitude to Prof Andreas Wendemuth for his support and helpful advise over the last years. He made my research possible by including me to his group at the Otto-von-Guericke university.

I wish to thank my colleagues, especially Martin Schafföner, Sven Krüger and Edin Andelic for countless valuable and helpful discussions. Martin, thank you so much for worldwide cluster support!

I am grateful to Prof Günter Meier for introducing me to the world of speech recognition. His scientific enthusiasm has given me encouragement to do this work.

I deeply thank my friends and whole family for supporting me throughout last years and without whom this thesis would not have been possible. Special thanks to my colleagues of the improvisation theater groups Kunstrasen e.V. and Hechtsprung. Further, my warmest thanks to my parents for their support and confidence in many ways. And finally my dearest thanks to my wife Andrea for her encouragement, patience, listening, pushing and trust. Now it's your turn!

Thanks are due to the Otto-von-Guericke university and the federal state Sachsen Anhalt for financial support.

Contents

List of Figures

List of Tables

List of Symbols

$\|\cdot\|$ ..... norm

$\mathbf{R}$ ..... expected risk

$\mathbf{R}_{emp}$ ..... empirical risk

d ..... dimensionality of the input space

h ..... VC dimension

k ..... Mercer kernel

$\boldsymbol{x}_i$ ..... the i'th training vector (sample)

y_i ..... the i'th training label

$\boldsymbol{X}$ ..... matrix of training vectors

$\boldsymbol{K}$ ..... kernel matrix

$\boldsymbol{I}$ ..... identity matrix

Φ ..... map into the feature space

$\boldsymbol{\Phi}$ ..... matrix of mapped vectors

$\mathcal{F}$ ..... set of functions

Δ ..... delta coefficients

$\Delta\Delta$ ..... delta-delta coefficients

λ ..... regularization constant

$\boldsymbol{\Sigma}$ ..... covariance matrix

$\boldsymbol{\mu}$ ..... mean vector

$\boldsymbol{\sigma}$ ..... variance vector

ξ ..... slack variable

$\mathbb{N}$ ..... set of natural numbers

N ..... number of training examples

M ..... number of selected/support vectors

$\mathbb{R}$ ..... set of reals

$\mathcal{S}$ ..... training subset

$\mathcal{Y}$ ..... set of training labels

$\mathcal{X}$ ..... set of training samples

H ..... Hilbert Space

$\mathcal{N}(\mu, \Sigma)$ .... Gaussian Distribution

$L(\beta)$ ..... (log)-likelihood function

$l(f(x), y)$ ... loss function

List of Abbreviations

AR Auto Regressive

DCF Detection Cost Function

DCT Discrete Cosine Transformation

DET Detection Error Tradeoff

DFT Discrete Fourier Transformation

DNA Deoxyribonucleic Acid

EER Equal Error Rate

EM Expectation Maximization

FA False Accept

FFT Fast Fourier Transformation

FR False Reject

FT Fourier Transformation

FVS Feature Vector Selection

GLDS Generalized Linear Discriminant Sequence

GMM Gaussian Mixture Model

HTER Half Total Error Rate

IER Identification Error Rate

IFFT Inverse Fast Fourier Transformation

IID Independently and Identically Distributed

IRLS Iteratively Re-Weighted Least Squares

KFD Kernel Fisher Discriminant

KKT Karush-Kuhn-Tucker

KL Kullback-Leibler

KLR Kernel Logistic Regression

LASSO Least Absolute Shrinkage and Selection Operator

LDA Linear Discriminant Analysis

LDC Linguistic Data Consortium

LP Linear Prediction

LPC Linear Predictive Coding

LPCC Linear Predictive Cepstral Coefficients

LR Logistic Regression

LRT Likelihood Ratio Test

MAP Maximum A Posteriori

MFCC Mel Frequency Cepstral Coefficients

ML Maximum Likelihood

MLE Maximum Likelihood Estimation

MLLR Maximum Likelihood Linear Regression

MPI Message Passing Interface

NAP Nuisance Attribute Projection

NIST National Institute of Standards and Technology

NLL Negative Log-Likelihood

NN Neural Network

PAV Pair-Adjacent Violators

PCA Principle Component Analysis

PDF Probability Density Function

PIN Personal Identification Number

PLP Perceptual Linear Prediction

RBF Radial Basis Function

RKHS Reproducing Kernel Hilbert Space

ROC Receiver Operating Characteristic

SKLR Sparse Kernel Logistic Regression

SRE Speaker Recognition Evaluation

SRM Structural Risk Minimization

SVM Support Vector Machine

TNorm Test-Normalization

UBM Universal Background Model

VC Vapnik Chervonenkis

ZNorm Zero-Normalization

1 Introduction

In general the field of **biometrics** (ancient Greek: bios = "life" and metron = "measure") deals with measurements of creature's specific characteristics. In particular biometrics addresses methods of identifying a creature by one or more of these characteristics, thus recognizing a subject by physiological or behavioral traits. An overview of different traits, divided in behavioral and physiological traits is given in figure 1.1. Well known biometric traits include fingerprints, iris, face and deoxyribonucleic acid (DNA) as well as signature and voice. For a more detailed introduction to biometrics see [Jain et al., 2004].

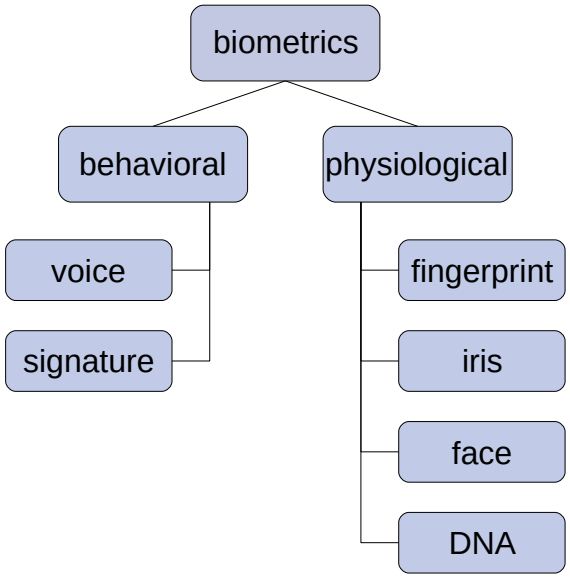

Figure 1.1: Overview of the most important biometric characteristics, classified into physiological and behavioral traits.

One of the oldest measurements of biometrics is the fingerprint which was introduced as an identification measure by Francis Galton in 1888, [Galton,

1888a] and [Galton, 1888b]. The fingerprint is still one of the most conventional method of identifying criminals.

Nowadays it is possible to realize automatic identification and verification methods with high accuracy, mainly due to the increase of computational power during last century.

Therefore the use of biometrics is more convenient than other traditional identification techniques today. Either a person needs to show a passport or a credit card or has to remember a password. In other cases one needs both, e.g., the bank card and the **personal identification number** (PIN) for the cash machine. The drawbacks are obvious: passwords can be forgotten, cards can be stolen and pin numbers can be observed by thieves when entered into a cash machine.

Instead of these methods the key idea of biometrics is to use the subject itself. Especially through the combination of different biometrics an improvement of security is expected [Kung et al., 2004].

Not only as a consequence of September 11, 2001, security issues became crucial for society in this century.

Authentication applications are highly relevant for networking, communication and mobility ranges from border or access control, credit card payments to telephone-based applications like telephone banking or telephone conferences.

This thesis is focused on automatic authentication or recognition of a subject by speech. Traditionally, security routines in telephone applications like customer services are based on simple queries like customer's name, date of birth or address. But a further authentication by customer's speech yields a higher security level for both, the company and the customer.

The main aim of this thesis is to examine and compare different kinds of classification methods in the field of speaker recognition. Additionally, competing feature extraction techniques in the acoustic space are investigated regarding the choice of classifiers. Furthermore strategies of combining information from competing feature streams and different classification methods are assessed.

Chapter 2 gives a brief overview of speaker recognition and describes various realizations like speaker identification and speaker verification. Moreover, we address current problems of state-of-the-art recognition systems.

The extraction of speaker specific or characterizing features from the speech

signal is a crucial step of speaker recognition. We review the main points of speech production in chapter 3 and introduce feature extraction methods used in this thesis. In chapter 4 we will discuss the theoretical background of **Statistical Learning Theory** and address the principles of **Structural Risk Minimization** with the aim to lay the foundation for the discriminative classifiers we will present in this thesis.

While generative classifiers have been studied in the field of speech and speaker recognition over years, more currently discriminative methods became a focus of interest. In chapter 5 we discuss different discriminative classification approaches like **Support Vector Machines** (SVMs) and **Kernel Logistic Regression** (KLR).

A novel speaker recognition system especially designed for limited training data is presented in chapter 6. This chapter provides speaker identification and verification experiments on the use of generative and discriminative classifiers.

Chapter 7 offers a detailed description of the speaker verification system designed for large speaker recognition evaluations and includes comparative experiments of the presented classification methods. In the following results on different fusing approaches for combining competitive feature extraction and classification methods are presented.

Finally, chapter 8 summarizes the results of this thesis and gives an outlook for future research.

2 Speaker Recognition

Speaker Recognition as part of the identification of behavioral traits in biometrics is a method of recognizing a speaker by voice. While other biometric techniques like fingerprint or retina patterns are based on physiological characteristics, human speech is a behavioral pattern and only moderately influenced by physiological characteristics of the speech production system.

Use and application of speaker recognition is already widespread, e.g., to give access to a private area or as identification feature in telephone based applications. In telephone banking systems it is very important that a person who prompts a credit card number verifies herself as owner of the card. Consequently, speaker authentication methods improve security aspects in such applications.

The field of speaker recognition consists of two tasks, namely **speaker identification** and **speaker verification**.

As in other biometric systems both speaker recognition methods can be divided into two phases, the **enrollment** and the **test phase**. During enrollment or training, the system stores the characteristic information of the speaker. On the one hand, this is done in order to use them as voice reference, on the other hand to estimate a stochastic model based on these information. However, during the test phase characteristics of a new speech example are compared to stored or estimated ones. In the following sections we give an overview of different fields of speaker recognition and review state-of-the-art classification and decision methods (see also [Campbell, 1997, Bimbot et al., 2004]).

2.1 Speaker Identification

A speaker identification task consists of a set of known speakers or **clients**. The problem of this task is to decide which person from the set is talking, see figure 2.1.

Closed sets only consider known speakers. Consequently, speech input has to be classified to one speaker of the set, even if the speech does not

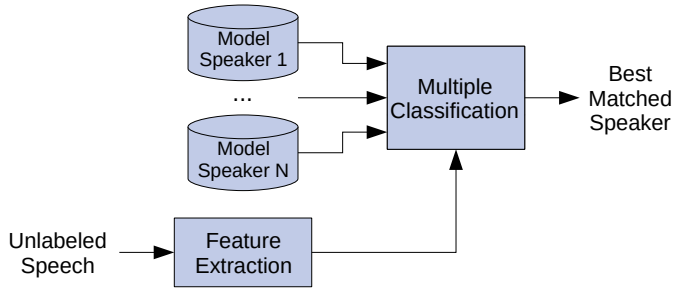

Figure 2.1: Illustration of a closed set speaker identification system.

originate from any speaker of the set. Alternatively, the **open set** speaker identification contains an additional rejection outcome to consider such cases. The open set speaker identification is more complex because the system must not only decide which speech input matches which speaker but also whether the characteristics of the speech input are close enough to any speaker in the set.

Related applications of speaker identification are **speaker tracking** and **speaker change detection**. While speaker tracking follows a client during a conversation, change detection systems detect changes of speakers during a conversation. These methods are essential in applications like automatic transcription of conversational speech, used in meetings or conferences.

In the scope of biometrics, speaker identification is also used in forensic speaker identification, e.g., to convict a suspect in a criminal investigation.

2.2 Speaker Verification

In the area of speaker verification the recognition system verifies if a person is the one he claims to be. Therefore, the main difficulty in this setup is the challenge that the system has not only to deal with the client but also with all kinds of **impostors**, who pretend to be a client. As illustrated in figure 2.2 the verification system asks for a binary (yes/no) decision instead of the multiple classification in speaker identification. In this case the speaker has to claim his identity before the onset of the core classification process. Speaker verification is also referred to as speaker detection.

Applications of speaker verification are ideal for security issues, such as

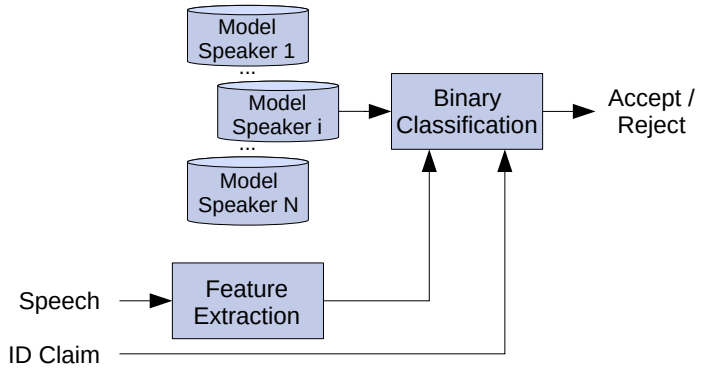

Figure 2.2: Illustration of a speaker verification process with to inputs. The speech and the identity claim of a speaker.

speaker authentication in commercial phone-based systems. In general it is more convenient and natural to use human voice for authentication than to remember a PIN or other kinds of passwords as necessary in traditional methods.

2.3 Text Dependence

A significant difference between speaker recognition systems is the relation to spoken text used during enrollment and test. The main text-styles of spoken text are **text-dependent**, **text-prompted** and **text-independent**, for a more detailed description see [Bimbot et al., 1994].

Text-dependent systems use the same pass-phrase during enrollment and testing. It is possible to use the same pass-phrase for all users or to allow fixed user pass-phrases, in which the user is free to choose his personal pass-phrase during enrollment and test.

In text-prompted systems the client is prompted to speak specific words or sentences during enrollment and testing. For example the speaker speaks several digits or digit sequences during the enrollment and for every digit a single speaker-specific model is created. During testing the user is prompted to speak a sequence of a few selected random digits.

In contrast, in text-independent systems the client can speak anything during enrollment and test, even in different languages. The accuracy of such systems might be limited because it is difficult to consider special speaker specific

characteristics, e.g., pronunciation of phonemes. To overcome this drawback it is possible to combine speech and speaker recognition systems [Stolcke et al., 2007]. The input speech is then recognized by the speech recognizer and a quasi text-dependent speaker recognizer could be enabled.

2.4 Speaker Variability and Robustness

The goal of speaker recognition is to discriminate a specific speaker from other speakers. Not surprisingly, the human voice changes over long periods of time, due to aging or due to speaker's health. And even in short periods of time the voice may sound different. These variations arise from the person herself, i.e. changes occur if the person has a cold, if she is lucky or sad or if she is just tired. The effects of speaker variabilities can be avoided by collecting speech samples of a speaker over a longer period of time.

As we know from speech recognition, the recognition rate always depends on the quality of speech samples. The recorded signal may contain background noise, like traffic or babble of other voices. Also different microphone types and transmission channels have a negative effect. Especially if these variabilities change between enrollment and test the recognition of the speaker becomes a challenge. In speech recognition these variabilities may be averaged out by a large amount of speech that is recorded in countless separate sessions. But in speaker recognition the amount of speech as well as the number of sessions for each speaker is limited and other methods dealing with these mismatches have to be applied. Some of these methods, like feature mapping are part of the recognition system presented in chapter 7.

2.5 Likelihood Ratio Test

Let us denote a speaker model of a speaker i by λ_i, $\lambda_i \in \{\lambda_1, ..., \lambda_N\}$, where N is the number of speakers in the set. The probability that a set of feature vectors $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_T\}$ is produced by the model λ_i of the speaker i, is computed by the log-likelihood over the whole sequence averaged by the number of feature vectors:

$$\log P(\boldsymbol{X}|\lambda_i) = \frac{1}{T} \sum_{t=1}^{T} \log P(\boldsymbol{x}_t|\lambda_i). \tag{2.1}$$

In the task of closed-set speaker identification, the sequence is simply assigned to the speaker with the highest production probability of the corresponding speaker model:

$$i^\star = \underset{j}{\operatorname{argmax}} \log P(\boldsymbol{X}|\lambda_j). \tag{2.2}$$

In the case of speaker verification we have to decide if the sequence is generated by the hypothesized speaker or not. The decision is realized by a **likelihood ratio test** (LRT). The LRT is a statistical test of the goodness-of-fit between two parametric models. In the case of speaker verification the ratio of two models λ_i and $\overline{\lambda_i}$ is given by

$$\frac{P(\boldsymbol{X}|\lambda_i)}{P(\boldsymbol{X}|\overline{\lambda_i})}, \tag{2.3}$$

where $P(\boldsymbol{X}|\lambda_i)$ represents the probability that the sequence $\boldsymbol{X}$ is from the hypothesized speaker i and $P(\boldsymbol{X}|\overline{\lambda_i})$ that the sequence $\boldsymbol{X}$ is not from the hypothesized speaker, see [Higgins et al., 1991].

For the probability $P(\boldsymbol{X}|\overline{\lambda_i})$ the concept of a **universal background model** (UBM) is used, see [Higgins et al., 1991] and [Reynolds, 1995]. Given a collection of speech samples from a large number of speakers a single model λ_{UBM} is trained to represent the probability of the denominator in formula (2.3). Normally, the same UBM is used as background probability for all target speakers in the speaker verification task. For a more detailed representation of different speakers, gender dependent background models are often used. But it is also possible, to model several specific characteristics of speech and to use one UBM for each of these characteristics.

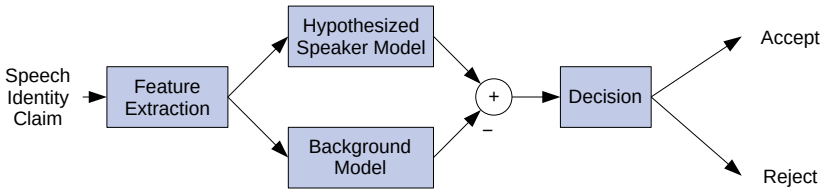

Figure 2.3: Block diagram of a typical speaker verification system based on the likelihood ratio test.

As can be seen in figure 2.3 the resulting score of the recognition process is computed by the logarithm of equation (2.3)

$$S_{\lambda_i}(\boldsymbol{X}) = \log(P(\boldsymbol{X}|\lambda_i)) - \log(P(\boldsymbol{X}|\overline{\lambda_i})) \tag{2.4}$$

Based on this score the system has to decide if the speaker is accepted as the claimed one or not. This is done by verifying whether the score exceeds a given threshold or not. The magnitude of this threshold is crucial for the system performance, see below.

2.6 Generative Classifiers

In this section we discuss how probabilities of class membership are generated by generative classifiers. This approach exclusively concentrates on supervised learning problems, where a training set $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ is given. The samples $\boldsymbol{x_i} \in \mathcal{X}$ are **independently and identically distributed** (IID) from an unknown but fixed probability distribution $P(\boldsymbol{x})$. The labels y_i specify the corresponding class membership of the sample and belong to the label set $\mathcal{Y} = \{1, \ldots, Y\}$.

The role of the classifier is to map from the samples $\boldsymbol{x}_i$ to the corresponding label $y_i \in \mathcal{Y}$. In the generative approach the joint distribution $P(\boldsymbol{x}, y)$ of training examples is modeled by learning the class-conditional density $P(\boldsymbol{x}|y)$ and the class prior probability $P(y)$. The posterior probability $P(y|\boldsymbol{x})$ is then obtained using Bayes' rule

$$P(y|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|y)P(y)}{P(\boldsymbol{x})}. \tag{2.5}$$

The corresponding Bayes classifier attempts to select the label by

$$f(\boldsymbol{x}) = \underset{y}{\operatorname{argmax}}\, \hat{P}(\boldsymbol{x}|y)\hat{P}(y) \tag{2.6}$$

where the hats represent estimated values. The prior $\hat{P}(y)$ can be estimated by counting the observations y in the training set. The probability $\hat{P}(\boldsymbol{x}|y)$ is often assumed as a parametric distribution where the distribution parameters have to be determined. If the estimated model $\hat{P}(\boldsymbol{x}|y)$ is close enough to the true unknown distribution $P(\boldsymbol{x}|y)$ then it could be used to generate samples with similar statistics. Typically, the probabilities $\hat{P}(\boldsymbol{x}|y)$ are estimated by

continuous density functions like Gaussian Mixture Models.

2.6.1 Gaussian Mixture Models

In speech and speaker recognition acoustic events are usually modeled by Gaussian **probability density functions** (PDFs), described by the mean vector $\boldsymbol{\mu}_i$ and the covariance matrix $\boldsymbol{\Sigma}_i$. But unimodal PDFs with only one mean and covariance are unsuitable to model all variations of a single event in speech signals. Therefore, a mixture of single densities is used to model the complex structure of the density probability. Such **Gaussian Mixture Models** (GMM) consist of C weighted Gaussian densities, each of the densities is represented by a mean $\boldsymbol{\mu}_i$, a covariance matrix $\boldsymbol{\Sigma}_i$ and a mixture weight c_i:

$$p(\boldsymbol{x}_t|\lambda) = \sum_{i=1}^{C} c_i \mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{2.7}$$

with $\sum_{i=1}^{C} c_i = 1$. The vector $\lambda = \{c, \mu, \Sigma\}$ contains all model parameters of the specific GMM. The i-th Gaussian density of the GMM in d-dimensional space is defined as

$$\mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_t - \boldsymbol{\mu}_i)^\top \Sigma_i^{-1}(\boldsymbol{x}_t - \boldsymbol{\mu}_i)\right) \tag{2.8}$$

While the parameter of a single Gaussian density distribution can be estimated directly by a standard **Maximum Likelihood** (ML) function, the mixture membership of the training samples is unknown and so the estimation of a mixture of such density parameters is more complicated.

The parameters of a GMM are often estimated by the **Expectation Maximization** (EM) algorithm. The EM algorithm is an iterative algorithm where each iteration consists of two steps, the estimation E and the maximization M. In the E-step the posterior probabilities of the samples are computed using the old parameters of the model:

$$\begin{aligned} P(i|\boldsymbol{x}, \lambda) &= \frac{p(\boldsymbol{x}|i, \lambda)P(i|\lambda)}{p(\boldsymbol{x}|\lambda)} \\ &= \frac{c_i\, \mathcal{N}(\boldsymbol{x}|\lambda_i)}{\sum_{j=1}^{C} c_j\, \mathcal{N}(\boldsymbol{x}|\lambda_j)} \end{aligned} \tag{2.9}$$

Based on the resulting membership probabilities of the samples, new model parameters are estimated in the M-step by maximizing the likelihood of the given data. The new model parameters are computed by:

$$\boldsymbol{\mu}_i^{new} = \frac{\sum_{n=1}^{N} P(i|\boldsymbol{x}_n, \lambda_i)\boldsymbol{x}_n}{\sum_{n=1}^{N} P(i|\boldsymbol{x}_n, \lambda_i)}, \tag{2.10}$$

$$\boldsymbol{\Sigma}_i^{new} = \frac{\sum_{n=1}^{N} P(i|\boldsymbol{x}_n, \lambda_i)(\boldsymbol{x}_n - \boldsymbol{\mu}_i^{new})(\boldsymbol{x}_n - \boldsymbol{\mu}_i^{new})^\top}{\sum_{n=1}^{N} P(i|\boldsymbol{x}_n, \lambda_i)}, \tag{2.11}$$

$$\boldsymbol{c}_i^{new} = \frac{1}{N}\sum_{n=1}^{N} P(i|\boldsymbol{x}_n, \lambda_i). \tag{2.12}$$

The steps of expectation and maximization are repeated until the model parameters converge. Alternatively, one can stop after a predefined number of iterations.

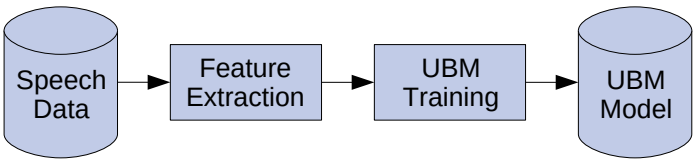

Figure 2.4: Training of the background model on a large amount of background speech, covering different characteristics of speakers as well as different transmission channels.

As already discussed in section 2.5 state-of-the-art speaker recognition systems are based on universal background models (UBMs). Using GMMs for modeling, the UBM is typically represented by a GMM trained on a large amount of speech data to model the speaker independent distribution $P(\boldsymbol{X}|\overline{\lambda_i})$ (see figure 2.4).

2.6.2 Parameter Adaptation of Speaker Models

The parameters of the GMM-UBM are estimated via the EM algorithm on a large amount of data. If one needs a model of a new dataset then one might think to start the whole parameter estimation again from scratch, especially

if the new data is independent from the already used set. However, if the new dataset is of the same structure as the used set, then it is possible to derive the new model by adapting the well estimated parameters of the existing model. Also if the amount of data for a new model is limited, it is not possible to model events that are not observed in the training data. This is usually the case in speaker recognition and as a consequence, hypothesized speaker models are derived successfully by adapting parameters of the UBM.

The adaptation method described in this section is known as **maximum a posteriori** (MAP) adaptation [Gauvain and Lee, 1994]. Like the EM algorithm, it is a two step estimation process where the first step is identical to the E-step of section 2.6.1. In the second step the new statistical estimates of the E-step are combined with the statistics from the previous model. Given a well estimated model and a set of new training data $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ we first determine the posteriori probability $P(i|\boldsymbol{x}_t)$ that the new vector $\boldsymbol{x}_t$ belongs to mixture i of the GMM:

$$P(i|\boldsymbol{x}_t) = \frac{c_i p_i(\boldsymbol{x}_t)}{\sum_{j=1}^{C} c_j p_j(\boldsymbol{x}_t)}. \tag{2.13}$$

The probability $P(i|\boldsymbol{x}_t)$ is then used to compute the statistics for the new weights c_i, the new means and the new variances:

$$E_i(\boldsymbol{x}) = \frac{1}{n_i} \sum_{t=1}^{T} P(i|\boldsymbol{x}_t)\boldsymbol{x}_t \tag{2.14}$$

$$E_i(\boldsymbol{x}^2) = \frac{1}{n_i} \sum_{t=1}^{T} P(i|\boldsymbol{x}_t)\boldsymbol{x}_t^2 \tag{2.15}$$

where n_i is the defined as the summed posteriori probabilities

$$n_i = \sum_{t=1}^{T} P(i|\boldsymbol{x}_t) \tag{2.16}$$

and $\boldsymbol{x}^2 = \mathrm{diag}(\boldsymbol{x}\boldsymbol{x}^\top)$. These statistics can be used to update the parameters

of the existing GMM to create a new model:

$$\hat{c}_i = \left(\alpha_i \frac{n_i}{T} + (1 - \alpha_i) c_i\right) \gamma \tag{2.17}$$

$$\hat{\boldsymbol{\mu}}_i = \alpha_i E_i(\boldsymbol{x}) + (1 - \alpha_i)\boldsymbol{\mu}_i \tag{2.18}$$

$$\hat{\boldsymbol{\sigma}}_i{}^2 = \alpha_i E_i(\boldsymbol{x}^2) + (1 - \alpha_i)(\sigma_i^2 + \boldsymbol{\mu}_i^2) - \boldsymbol{\mu}_i^2 \tag{2.19}$$

where γ is a scale factor computed over all adapted mixture weights $\hat{c}_i$ to ensure that $\sum_i \hat{c}_i = 1$. The adaptation coefficient α_i controls the balance between the old and the new estimates:

$$\alpha_i = \frac{n_i}{n_i + r} \tag{2.20}$$

with the so called **relevance factor** r. This relevance factor is predefined by the user depending on the amount of training data.

If only a small amount of data is observed then the corresponding adaptation coefficient tends to zero and the influence of the new statistics is small. Otherwise for large n_i the adaptation coefficient tends to one and the influence of the original parameter is reduced.

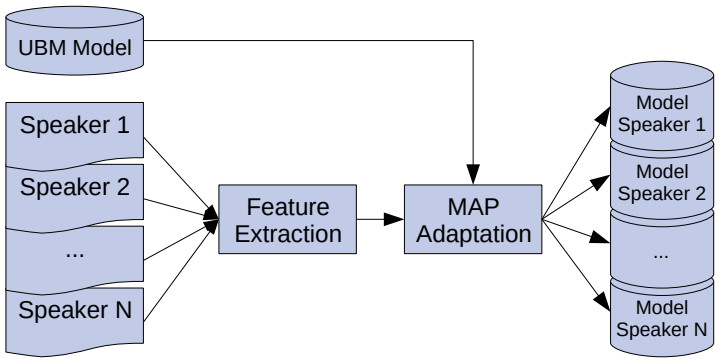

Figure 2.5: Training (or enrollment) of speaker models based on MAP adaptation from the Gaussian mixture background model.

The training or adaptation of speaker models is usually called **enrollment phase**. Figure 2.5 illustrates this process in case of MAP adaptation of speaker models.

2.7 Feature Mapping

The compensation of intersession variabilities is considered to be one of the major problems in speaker recognition. These variabilities occur between training and testing by using different telephone handsets and/or different transmission channels. Feature Mapping [Reynolds, 2003] is a popular normalization method to reduce the influence of different transmission channels and telephone handsets. It operates directly in the feature space by first identifying the used channel (or handset) in the observation and secondly transforming the feature vectors into a channel independent feature space.

The channel dependent models are adapted from the channel independent *root*-UBM by a one pass MAP adaptation. This results in a shift of the mean vectors of the adapted models that is hopefully characteristic for the specific channel. Typical channels are landline and cellular and in more details carbon or electret, standard or cordless handset, gsm or cdma transmission channels. Having adapted the channel dependent GMMs, all observations are scored against the dependent models and each feature vector is mapped to the channel independent space. This mapping is realized by the parameters of the Gaussian mixture achieving the highest mixture probability of the corresponding channel dependent model:

$$\boldsymbol{x}_{id} = (\boldsymbol{x}_{dp} - \boldsymbol{\mu}_{dp})\frac{\boldsymbol{\sigma}_{id}}{\boldsymbol{\sigma}_{dp}} + \boldsymbol{\mu}_{id} \tag{2.21}$$

with the channel dependent (independent) feature vector $\boldsymbol{x}_{dp}$ ($\boldsymbol{x}_{id}$), the channel dependent (independent) Gaussian mean $\boldsymbol{\mu}_{dp}$ ($\boldsymbol{\mu}_{id}$) and the respective variances $\boldsymbol{\sigma}_{dp}$ and $\boldsymbol{\sigma}_{id}$.

2.8 Performance Evaluation

In speaker verification (or detection) usually two error rates occur. The first error rate is the **false reject** error rate (or miss rate), FR: The speaker **is** the claimed client but is classified as an impostor. The second error rate is defined as **false accept** (or false alarm) error rate, FA: The speaker **is not** the claimed one but is classified as the client. Both of these error rates are dependent on the decision threshold and effect each other. The lower the threshold is the more impostor identity claims are accepted as true targets. This also leads to a low false reject error rate but of course to a high false accept error rate. If the threshold is set to a very high value all the identity claims are rejected.

The result is a high false reject error rate and a low false accept error rate. If the system is evaluated on a large range of decision thresholds we receive a number of $\{FA, FR\}$ pairs that describe different **operation points** of the system. So, the best operation point of the system is always a trade-off between the two types of verification (or detection) errors. This trade-off is illustrated in a **receiver operating characteristic** (ROC) curve, where the miss probability is plotted against the probability of false alarm. For a better presentation of the results it is possible to rescale the axes to be normally distributed so that Gaussian distributed scores result in a straight line. This variant of the ROC curve is called **detection error tradeoff** (DET) [Martin et al., 1997].

The two error rates can be combined for a single performance measure of the verification system, the **detection cost function** (DCF). The DCF is defined as a cost function, where the costs C_{FA} and C_{FR} are assigned to the FA and the FR rates:

$$C_{det} = C_{FR} \cdot FR \cdot P_{Target} + C_{FA} \cdot FA \cdot P_{NonTarget} \tag{2.22}$$

with the a priori probabilities P_{Target} and $P_{NonTarget} = 1 - P_{Target}$ of target and impostor speakers, respectively. Often the best operating point of the system is included as circle on the DET curve. If we assume equal costs $C_{FA} = C_{FR} = 1$ and equal a priori probabilities the DCF is reduced to the **half total error rate** (HTER):

$$HTER = \frac{1}{2}(FR + FA) \tag{2.23}$$

Often a more intuitive measure is used to report the performance of the speaker recognition system: the **equal error rate** (EER). The EER occurs if the decision threshold of the verification system is set to a value so that the number of false rejection equals the number of false acceptance. In practice the EER is an approximation representing the point where the distance between FA and FR is minimal.

2.9 Decision Threshold Setting

On the basis of the classification scores obtained from figure 2.3 it is necessary to determine a decision threshold of accepting or rejecting a speaker claim during verification. This threshold can be set for each speaker separately, or

as a global threshold for all speakers using the same system.

Figure 2.6 shows an example of the target and impostor score distributions. The smaller the overlap between the two histograms the smaller the probability of error.

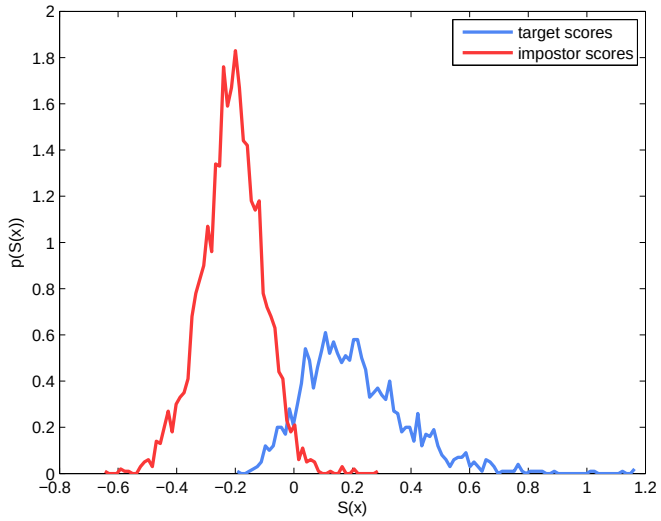

Figure 2.6: Score distributions for target speaker and impostor scores $S(\boldsymbol{x})$.

The value of the decision threshold depends on the task of the recognition system. In authentication applications like telephone banking or access control the false acceptance rate should be very small. The DCF can be used to estimate a good operating point by setting the costs C_{FR} and C_{FA} to adequate values. But the DCF is based on an a posteriori minimization of the cost function during the verification. Of course, this is always to the advantage of the system and for real-world evaluations one should determine a priori thresholds on a development set using (2.22).

Following the NIST evaluation protocols [Przybocki and Martin, 2006], the target speaker probability is set to $P_{Target} = 0.01$ and the impostor speaker probability to$P_{NonTarget} = 0.99$. The relative costs of detection errors in this function are the costs of miss (C_{Miss}) and false alarm errors ($C_{FalseAlarm}$), these parameters are set to $C_{FR} = C_{Miss} = 10$ and $C_{FA} = C_{FalseAlarm} = 1$.

2.10 Score Normalization

It is possible to use the output scores of the Likelihood Ratio Test directly in the verification process. But the scores values are effected by variabilities between enrollment and test conditions, background noise, quality of speaker models and duration of speech segments.

The output scores depend on the target speaker model as well as on the test segment itself, so that the scores are biased to the speaker or the observation. Finally, the decision threshold is fixed for all speakers and depends on the distribution of the output scores, see section 2.5. This means that the scores of different target speakers should be centered around the same value and the scores of impostor speakers also. Therefore, the purpose of normalization is to perform a distribution scaling on the scores.

The score normalization techniques used in this thesis are based on normalizing the score by

$$\hat{S}_\lambda(\boldsymbol{X}) = \frac{S_\lambda(\boldsymbol{X}) - \mu_\lambda}{\sigma_\lambda} \tag{2.24}$$

where mean μ_λ and standard deviation σ_λ are estimated on scores from known impostor speakers, so called **pseudo impostors**. The commonly used score normalization approaches are the Test-Normalization (TNorm) and the Zero-Normalization (ZNorm).

2.10.1 TNorm

In the **Test Normalization** (TNorm) approach [Auckenthaler et al., 2000] a set of TNorm models is trained on pseudo impostor speakers. The models are derived in the same way from the background model as the target speaker models. During the evaluation the test utterance is scored as usual by the speaker model but also by the set of TNorm models. The speaker scores are then normalized by the mean $\boldsymbol{\mu}_\lambda$ and the standard deviation $\boldsymbol{\sigma}_\lambda$ of the TNorm scores as given in equation (2.24). A disadvantage of the TNorm is that the test utterance has to be scored by an additional amount of models. While the TNorm set sometimes consists of several hundred of pseudo impostor models it is possible to select a subset of pseudo impostor speakers depending on different approaches like speaker specific TNorm sets, e.g., selecting the most similar TNorm models compared to the target speaker.

2.10.2 ZNorm

The **Zero Normalization** (ZNorm) approach [Li and Porter, 1988] is also based on the estimate of mean and variance parameters. The target speaker model is tested against a set of pseudo impostor speech utterances. The ZNorm parameters $\boldsymbol{\mu}_\lambda$ and $\boldsymbol{\sigma}_\lambda$ are then estimated on these speaker specific impostor scores. During the evaluation the test utterance is scored by the speaker model and the normalized score is computed by shifting and scaling the speaker score by the precomputed mean and standard deviation as in (2.24). The advantage of the ZNorm is that the normalization parameters are estimated offline as a part of the training process.

2.11 Summary

In Chapter 2 we reviewed basic methods used in state-of-the-art speaker recognition systems. First we summarized several possibilities to apply these methods in different recognition fields and discussed difficulties concerning text dependence and robustness. In the following we concentrated on the main decision process which is based on a traditional universal background model (UBM) concept and Gaussian mixture models (GMMs). Additionally, we described standard methods for channel mismatch compensation and score normalization.

In general, this chapter addressed the fundamentals of speaker recognition to understand the integration of novel classification methods and recognition environments. In the next chapter we review the main methods of speech processing that will be used in this work.

3 Fundamentals of Speech Processing

This chapter reviews the fundamental principles of speech production in human beings and shows how these mechanics leads to several different approaches of speech parameterization to speech and speaker recognition. Then, in section 3.2 we discuss three popular approaches of speech parameterization that will be used in the speaker recognition systems presented in chapter 6 and 7. For a more detailed description of the fundamentals of Speech Processing we refer to [Rabiner and Juang, 1993, Wendemuth, 2004a]

3.1 Human Speech Production

In this section we outline speech production in human beings and focus on important points for the extraction of relevant features for speaker recognition.

Figure 3.1 shows a schematic diagram of the human voice production system. First, air is expelled from the lungs through the **trachea** to the **larynx** and passes the **vocal chords** within the larynx. Secondly the vocal chords vibrate by the airstream depending on the kind of expressed sound. Thirdly sounds are produced by this vibration of the vocal chords and the airstream is chopped into quasi-periodic pulses with a frequency varying from 60 Hz to 200 Hz for males and from 120 Hz to more than 350 Hz for females. This frequency is also called the **fundamental frequency F0**. For unvoiced sounds the vocal chords are open and the airstream can pass directly through the larynx.

Beyond the vocal chords the continuous or chopped air flow reaches the most important and complex part of the speech production system: the **vocal tract**. The vocal tract is basically an acoustical tube and can be divided into the **oral tract** (together with the **pharyngeal**) and the **nasal tract**. The oral tract is terminated by the vocal chords at one and by the lips at the other end. The average length depends on the sex of the speaker, about 17 cm for male and about 15 cm for female speakers. Different parts of the oral tract, the

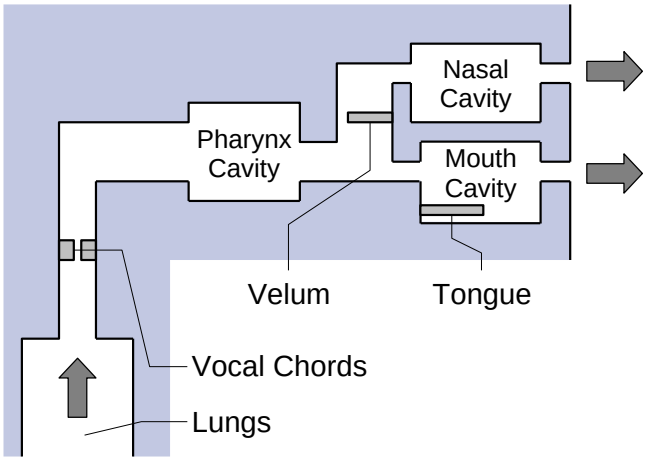

Figure 3.1: Schematic view of the human voice production system.

velum, the tongue and the lips serve as **articulators**. Different proportions of these articulators can cause various cross-sections of the oral tract which are responsible for multiple possible resonance frequencies of the vocal tract. Conversely, the cross-section of the nasal tract is fixed and the amount of air flow to the nasal tract is controlled by the velum.

3.1.1 Source-Filter Model

The speech production system can be simplified to the **source-filter model** presented by [Fant, 1960]. As illustrated in figure 3.2 the whole production process is reduced to two different excitations and an acoustic filter. The two excitations represent the vocal chords and the acoustic filter models the vocal tract. The first excitation models the tensed vocal chords and is realized by a pulse train source characterizing voiced sounds like vowels and nasals. The second one is a simple white noise source which is needed to produce unvoiced sounds like fricatives, where the vocal chords are open and the airstream passes through.

The vocal tract considered as the acoustic filter is characterized by its resonance frequencies, where the energy of the source signal reaches local maxima. These local maxima of the spectrum are called **formants** and typically there are up to four resonance frequencies or formants of significance. Figure 3.3 gives a small example of a short speech frame. The formants can be observed as local maxima of the smoothed signal.

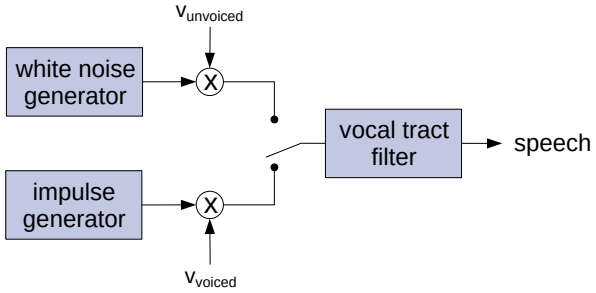

Figure 3.2: Speech production model with two excitations, white noise and pulse train.

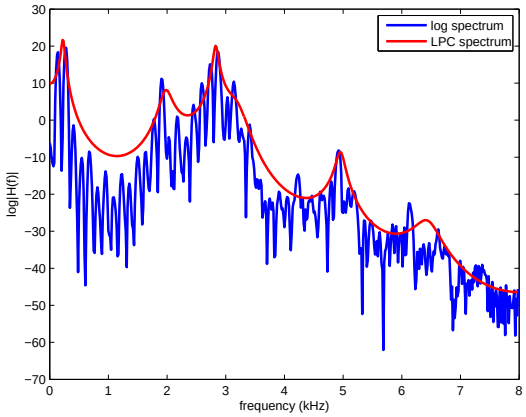

Figure 3.3: Example of a short voiced speech sample in the frequency domain. Representation of original log spectrum and corresponding formant frequencies by linear predictive coding.

3.2 Speech Parameterization

For the purpose of speech and speaker recognition the speech signal itself contains a lot of redundant and irrelevant information. Therefore we want to extract the features that contain relevant information of the speech signal in the speech parameterization process. The most important information for speech processing tasks are given by the characteristics of the vocal tract.

As discussed in section 3.1 these characteristics are evidenced in the frequency domain by the location of the formants. The formants, observable as peaks in figure 3.3, are primarily responsible for the recognition of specific phonemes in speech recognition and of course for the identification of a specific gender or a specific speaker.

Additionally, we reduce the amount of data to get a compact and less redundant representation of the speech data in the parameterization process.

In a first step the speech data has to be discretized by a sampler using an adequate sampling frequency f_a. To prevent spectral alias effect the signal should be low-pass filtered with a corner frequency lower than $f_a/2$ before the sampling.

As known from signal processing, traditional methods of spectral evaluation are only reliable in the case of stationary signals. But unfortunately the voice signal is only stationary within a very short time and exclusively for voiced speech. Consequently we have to select short sections of speech where the speech signal is assumed as stationary. This selection is realized by a small window that is moved over the speech signal with a predefined window size and window shift. Typically widow sizes are between 20 ms and 30 ms and the window shift is chosen to be between 10 ms and 20 ms. The window shift has to be smaller than the window size to guarantee that all information are captured by the window. The simplest window is the rectangular window. The rectangular window has excellent resolution characteristics for signals of comparable strength but it is a poor choice for signals of disparate amplitudes like speech. This is caused by the window's side effects in which the disparate amplitudes causes high frequencies in the signal's spectrum. A good choice for speech representation is the use of the so called **Hamming window**. This window tapers samples on each window so that discontinuities at window edges are attenuated:

$$w_k = \begin{cases} 0.54 - 0.46\ \cos(2\pi \cdot \frac{k}{M-1}), & 0 \leq k \leq M-1, \\ 0 \qquad \text{otherwise}, \end{cases} \tag{3.1}$$

where M is the window length. The resulting windowed speech section is also called speech frame.

Typically, a **pre-emphasis** is applied to the speech signal before any further processing. This pre-emphasis is used to compensate the negative spectral slope of the speech signal and is realized by a first-order high-pass filter $H(z) = 1 - \alpha z^{-1}$, where α controls the slope of the filter. The filter is implemented as a first order difference of the speech frame $s(n)$, where the filter coefficient α falls typically between 0.9 and 1:

$$y(n) = s(n) - \alpha s(n-1). \tag{3.2}$$

For spectral analysis the windowed signal has to be transformed into the frequency domain by the **Fourier Transformation** (FT). After the sampling process the speech signal is a discrete signal and the FT is evaluated only for a discrete number of values. If these values are equally spaced the **Discrete Fourier Transformation** (DFT) of all frames of the speech signal can be obtained:

$$\boldsymbol{S}_m(k) = \sum_{n=0}^{N-1} s(m-n)w(n)e^{-j2\pi kn/N} \tag{3.3}$$

where $s(m-n)w(n)$ represents the windowed speech signal. While the complexity of the DFT computation is $O(N^2)$ the computation can be realized via an efficient implementation called **Fast Fourier Transformation** (FFT). The computational complexity of the FFT is $O(N \log_2 N)$. The radix-M FFT is defined through $N = M^q$ with some $M \in \mathbb{N}$, typically $M = 2$ or 4.

3.2.1 Cepstral Features

The cepstral analysis is a method to deconvolve a convolution of two signals. As mentioned earlier, the speech signal is a convolution of the excitation and the vocal tract filter response, see figure 3.2. After the transformation of the speech signal into the frequency domain by the FFT the convolution becomes a product of excitation and filter response. The logarithm of the multiplication becomes a summation of the two parts. Finally, the expression is transformed into the cepstral domain by the **Inverse Fast Fourier Transformation** (IFFT) where the excitation is represented as a small peak. Denoting the excitation signal $u(n)$ and the filter response $h(n)$ the cepstrum is computed as:

$$\begin{aligned} IFFT\{\log FT\{s(n)\}\} = IFFT\{\log FT\{u(n)\}\} \\ + IFFT\{\log FT\{h(n)\}\} \end{aligned} \tag{3.4}$$

3.2.2 Filter Bank Processing

In speech and speaker recognition filter banks are commonly used to simulate the human perception of the frequency content of sounds. Because the human ear perceives only larger ranges of frequencies it can not discriminate closely adjacent frequencies. Different filter shapes, like rectangular or triangular,

can be used to integrate the spectrum at different frequencies. Additionally, the resolving power of the ear decreases towards higher frequencies. As a result frequencies can be warped according to some non-linear functions, like mel or bark-scale.

3.2.3 Mel-Cepstral Features

Frequencies are resolved non-linearly across the frequency spectrum by the human ear. Interestingly it has been shown empirically that designing the feature extraction in a similar manner improves the recognition performance in speech and speaker recognition.

The filters of the mel-scale filter bank are spaced non uniformly along the frequency axis to obtain the desired non-linear frequency resolution as a consequence of this observation. The filter bank is constructed of a set of triangular filters that are equally spaced along the mel-scale [Davis and Mermelstein, 1980] which is defined as:

$$mel(f) = 2995 \log(1 + \frac{f}{700\,\mathrm{Hz}}) \tag{3.5}$$

Supposing an N-point FFT spectrum $S(j)$ and a M-point filter bank with the i-th filter response $H_i(j)$, the output $Y(i)$ is given by

$$Y(i) = \sum_{j=1}^{N} S(j) H_i(j). \tag{3.6}$$

Since the logarithm of the filter bank output is real and symmetric, the IFFT is reduced to the **Discrete Cosine Transformation** (DCT):

$$c_j = \sqrt{\frac{2}{M}} \sum_{i=1}^{M} Y(i) \cos\left(\frac{\pi j}{M}(i - 0.5)\right) \tag{3.7}$$

to achieve the **mel frequency cepstral coefficients** (MFCC).

3.2.4 Linear Prediction Analysis

In contrast to the deconvolution method described in the last section the characteristics of speech can also be modeled in the time domain by **linear predictive coding** (LPC). LPC is based on the underlying idea, that

a given speech sample can be approximated as a linear combination of the past p speech samples and an excitation term. This is also known as an **autoregressive** (AR) process, or autoregressive model. The AR model assumes that speech is produced by exciting a linear filter, which is the vocal tract in speech production, by either a series of impulses, if the sound is voiced, or white noise, if it is unvoiced (see figure 3.2). Therefore the AR model provides a good approximation of the vocal tract spectral envelope. For more details we refer to [Rabiner and Juang, 1993, Wendemuth, 2004b].

In **linear prediction** (LP) analysis the vocal tract is modeled by an AR model (or all-pole filter) with transfer-function

$$H(z) = \frac{1}{\sum_{i=0}^{p} a_i z^{-i}} \tag{3.8}$$

where p is the order of the model and is equivalent to the number of poles and $a_0 = 1$, [Fant, 1960]. The filter coefficients a_i are estimated by minimizing the mean square error of the filter prediction over the windowed speech signal. Technically, the minimization is done by the autocorrelation method, where the prediction coefficients are recursively estimated by the **Levinson-Durbin** recursion, see [Levinson, 1947, Durbin, 1960].

To use the LPC coefficients in speech or speaker recognition feature vectors should be de-correlated like the cepstral coefficients. The cepstral coefficients c are derived from the LPC coefficients by

$$c_0 = \ln \sigma^2, \tag{3.9}$$

$$c_m = a_m + \sum_{k=1}^{m-1} \frac{k}{m} c_k a_{m-k}, \qquad 1 \leq m \leq p \tag{3.10}$$

$$c_m = \sum_{k=m-p}^{m-1} \frac{k}{m} c_k a_{m-k}, \qquad m \geq p \tag{3.11}$$

where σ^2 is the gain term in the LPC model [Rabiner and Juang, 1993]. The final feature vectors are called **linear predictive cepstral coefficients** (LPCC).

3.2.5 Perceptual Linear Prediction

Alternatively to the mel-frequency cepstral and linear predictive cepstral coefficients the **perceptual linear prediction** (PLP) coefficients [Hermansky, 1990] can be applied.

The PLP feature extraction is based on the standard mel-frequency filter bank where the mel filter bank coefficients are additionally weighted by an equal-loudness curve and then compressed by taking the cubic root. LP coefficients are estimated and converted to cepstral coefficients from the resulting auditory spectrum in the same way as in equations (3.9) to (3.11).

Figure 3.4 provides a comparison of the described feature extraction methods, i.e. the cepstral, the autoregressive and the perceptual method.

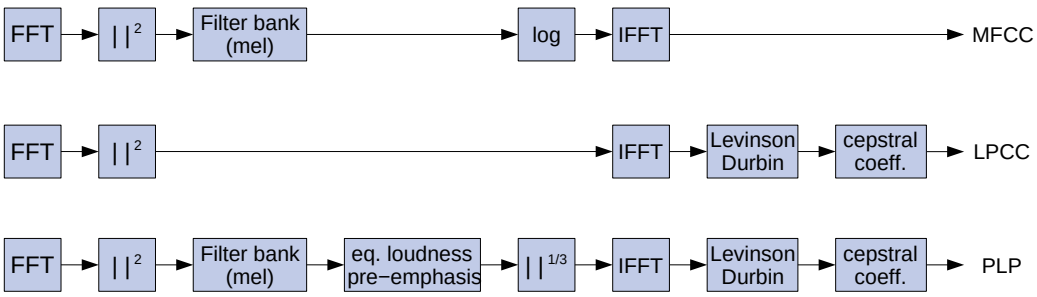

Figure 3.4: Comparison of different feature extraction methods used in this thesis: MFCC, LPCC and PLP

3.2.6 Dynamic Features and Energy Measure

Up to this point we only considered the static feature vectors computed for each speech frame and ignored the dynamic evolution of the speech signal. A further improvement in performance could be obtained by appending dynamic features to the static vectors.

Dynamic features are the time differences of the static feature vectors and usually first and second order differences are used, but also third order differences may be used. Instead using the time differences directly, one usually approximate the dynamic features by a first order polynomial regression of

the static feature vectors given by c_t, [Furui, 1986]:

$$\Delta_t = \frac{\sum_{i=1}^{N} i(c_{t+i} - c_{t-i})}{2\sum_{i=1}^{N} i^2}. \tag{3.12}$$

These features are also called delta (Δ) coefficients. In a similar way formula (3.12) can by applied to obtain the second order time differences or acceleration coefficients that are called delta-delta ($\Delta\Delta$) coefficients.

Frame energy is often used in speech recognition since differences in energy can be observed among difference speech sounds. The frame energy is added as additional component of the feature vector and yields a performance improvement of the recognizer. For a better representation of the whole range of energy values the logarithm is used. The logarithm of the energy can be computed from the speech signal by:

$$E = \log \sum_{i=0}^{d} s(i)^2. \tag{3.13}$$

The energy by itself normally does not contain speaker relevant information, but it has been shown that the first and second time differences of the energy are useful for the speaker recognition process. The time differences are computed in the same way as for the other feature components in equation (3.12).

3.3 Speech Detection

An important step in text-independent speaker recognition is the speech detection. For a better estimation of speaker models the silence or sections that only contain background noise have to be removed from the speech signal. A good indicator of speech is the energy, see equation (3.13) of the windowed signal.

First, the energy of each speech frame of an utterance is computed by equation (3.13). Secondly, the frame energy is smoothed by a particular window function in order not to discard single frames. Then, the smoothed energy features are used to construct a probability model of speech, which gives a probability that a frame contains speech or silence. Only frames that are labeled as speech are used for the further process and all other frames (labeled as silence) are discarded.

3.4 Summary

In this chapter we outlined the general human speech production and revisited three popular speech parameterization techniques, i.e. MFCC, LPCC, and PLP. We will investigate these techniques in our speaker recognition environments that are presented in chapter 6 and 7.

Finally, we described an energy based method for speech detection. Speech detection is an essential step in speaker recognition to discard frames that do not contain any speech and as a consequence no relevance information about the speaker.

4 Statistical Learning Theory

In this chapter we introduce classification methods based on statistical learning theory. We concentrate on supervised learning methods, where a training set $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ is given. The samples $\boldsymbol{x_i} \in \mathcal{X}$ are **independently and identically distributed** (IID) from an unknown but fixed probability distribution $P(\boldsymbol{x})$. The role of the classifier is to map from the samples $\boldsymbol{x}_i$ to the corresponding label $y_i \in \mathcal{Y}$.

In the following chapter we investigate different types of discriminative classifiers. While the generative classifier, like the GMM, models the distribution that generates the sample $\boldsymbol{x}_i$, the discriminative one models the class membership of the sample directly.

4.1 Loss Functions

Given the input data point $\boldsymbol{x} \in \mathcal{X}$, the prediction $f(\boldsymbol{x}) \in \mathcal{Y}$ and the true result $y \in \mathcal{Y}$, then the map

$$l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R} \tag{4.1}$$

is called the loss function $l(f(\boldsymbol{x}), y)$. The loss function is a measure of fit between the model assumption of data and the actual data. Simply speaking, the loss function $l(f(\boldsymbol{x}), y)$ measures how costly it is when the prediction of the input $\boldsymbol{x}$ is $f(\boldsymbol{x})$ and the true result is y. In binary classification the simplest choice of a loss function is the zero/one loss:

$$l(f(\boldsymbol{x}), y) = \begin{cases} 0 & \text{if } f(\boldsymbol{x}) = y \\ 1 & \text{otherwise} \end{cases} . \tag{4.2}$$

If the data point $\boldsymbol{x}$ is classified correctly, there is no penalty, otherwise the function counts the misclassification error. Usually the set of true results is $\mathcal{Y} = \{-1, +1\}$ and classification is realized by the signum function of $f(\boldsymbol{x})$. Other loss functions used in this thesis are the **soft margin** loss [Bennett and

Mangasarian, 1992]

$$l(f(\boldsymbol{x}), y) = \begin{cases} 0 & \text{if } yf(\boldsymbol{x}) \geq 1 \\ 1 - yf(\boldsymbol{x}) & \text{otherwise} \end{cases}, \tag{4.3}$$

and the **logistic loss**, defined by

$$l(f(\boldsymbol{x}), y) = \log(1 + \exp(-yf(\boldsymbol{x}))). \tag{4.4}$$

In the case of the soft margin loss, the quality of the estimate depends on the product $yf(\boldsymbol{x})$. The product will be positive, if the signum function of $f(\boldsymbol{x})$ and y agree. As we will see later, the logistic loss leads to probabilistic outputs of corresponding classifiers. Another popular loss function for **least squares** is the **square loss** function

$$l(f(\boldsymbol{x}), y) = (y - f(\boldsymbol{x}))^2. \tag{4.5}$$

A comparison of the different loss functions is given in figure 4.1. As we will see later, it is important that loss functions satisfy certain properties. They should be cheap to compute, and be convex in order to ensure a global minimum in the optimization process.

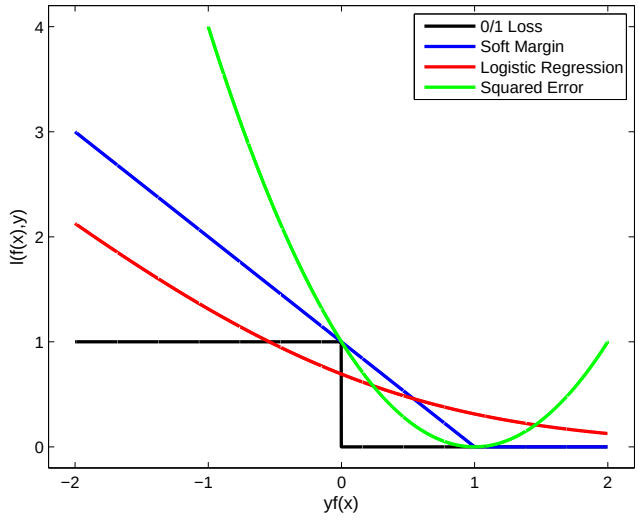

Figure 4.1: Comparison of different loss functions for binary classification with true outputs $y = \pm 1$ and predictions $f(\boldsymbol{x})$.

4.2 Learning from Examples

Suppose we are given a training set of input samples $\boldsymbol{x}$ and corresponding targets y

$$(\boldsymbol{x}, y) \in \mathbb{R}^d \times \{\pm 1\} \tag{4.6}$$

that were generated from the unknown distribution $p(\boldsymbol{x}, y)$. We now want to learn a function $f : \mathbb{R}^d \to \{\pm 1\}$ out of a set of functions $\mathcal{F}$ which minimizes the expected value of any loss function $l(f(\boldsymbol{x}), y)$. This expectation is called the **actual risk** and is given by

$$\mathbf{R}(f) = \int l(f(\boldsymbol{x}), y) p(\boldsymbol{x}, y) d\boldsymbol{x} dy. \tag{4.7}$$

Since the distribution $p(\boldsymbol{x}, y)$ is unknown we cannot minimize the risk $\mathbf{R}(f)$ directly. Therefore, an approximation has to be provided that is close to the optimal function f which minimizes the actual risk $\mathbf{R}(f)$ among all $f \in \mathcal{F}$. Such a method is called an **induction principle**. Instead of minimizing the actual risk (4.7) we minimize the risk on a limited training set by the **empirical risk** $\mathbf{R}_{\text{emp}}$:

$$\mathbf{R}_{\text{emp}} = \frac{1}{N} \sum_{i=1}^{N} l(f(\boldsymbol{x}_i), y_i). \tag{4.8}$$

However, it is not guaranteed that the minimum of the empirical risk is the solution that minimizes the actual risk. Figure 4.2 illustrates an example of possible solutions for a classification task. The training data may be perfectly discriminated by the complex function (dotted) but does not generalize to unseen examples. Conversely, if the function f chosen from the function class $\mathcal{F}$ is too simple (dashed), the samples are under-fit. A good tradeoff between the two function is the solid hypothesis. To avoid these problems we need to restrict the complexity of the function class.

The questions of consistency, over- and under-fitting are closely related and will lead us to the principle of **Structural Risk Minimization**. Let us define more closely what consistency means and how it can be characterized. Let us denote by f the function $f \in \mathcal{F}$ that minimizes the empirical risk (4.8) for a given training set $\mathcal{X}$ of size N. The notion of consistency implies that

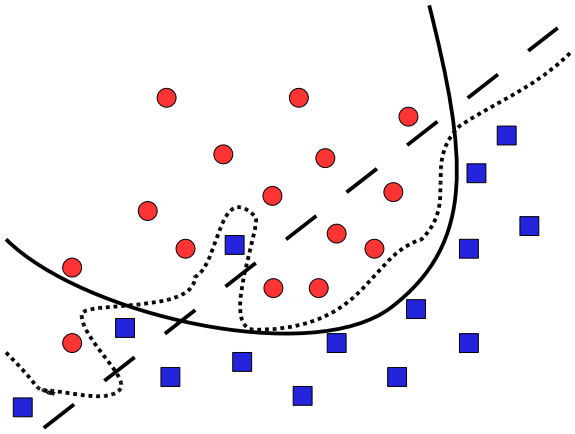

Figure 4.2: Illustration of the under and over-fitting dilemma. The simple linear function (dashed) under-fits the data while the complex solution (dotted) perfectly discriminates the two classes. The solid hypothesis appears to be a good trade-off and is more likely to generalize unseen samples.

for $N \to \infty$, the empirical risk is consistent for all $f \in \mathcal{F}$, if

$$|\mathbf{R}(f) - \mathbf{R}_{\text{emp}}(f)| \to 0. \tag{4.9}$$

As we have already seen in figure 4.2 such convergence may not be the case in general, the reason being that f now depends on the training set $\mathcal{X}$. One can show [Vapnik, 1998] that a necessary and sufficient condition for consistency is uniform convergence, over all functions in $\mathcal{F}$, of the difference between the expected and the empirical risk to zero. This insight is summarized in the following theorem:

Theorem 4.1 (One-sided uniform convergence of risk) *One-sided uniform convergence in probability, i.e.*

$$\lim_{N\to\infty} P\left[\sup_{f\in\mathcal{F}} (\mathbf{R}(f) - \mathbf{R}_{emp}(f)) > \epsilon\right] = 0 \tag{4.10}$$

for all $\epsilon > 0$, is a necessary and sufficient condition for nontrivial consistency of empirical risk minimization.

4.3 Vapnik Chervonenkis Dimension

A specific way of controlling the complexity of a function class $\mathcal{F}$ is given by the **Vapnik Chervonenkis** (VC)-theory and the **Structural Risk Minimization** (SRM) principle [Vapnik, 2000]. Here we use the VC dimension to capture the concept of complexity.

The VC dimension is a measure of model capacity and was introduced by Vladimir Vapnik and Alexey Chervonenkis [Vapnik, 2000]. The VC dimension h gives the maximum number of data points in $\mathbb{R}^d$ that can be shattered by a given model. A common example of the VC dimension is a two-class separation problem given a linear classification function in $\mathbb{R}^d$ with $d = 2$. While it is possible to separate any $N = 3$ data points by an one-dimensional hyperplane in $d^N = 2^3$ different ways, it is not possible to separate all arrangements of $N \geq 4$ data points in all possible ways in 2 dimensions. Extrapolating this example to the d-dimensional space, the VC dimension h of a linear classification function is given by $h = d + 1$. For comparison, the VC dimension of a linear classification function plus threshold operation (e.g. sign) is approximately $h = 2d$. For an analytical treatment, see [Cover and Thomas, 1991].

4.4 Structural Risk Minimization

The minimization of the empirical risk does not guarantee a small actual risk, but it is possible to give an upper bound of the actual risk in (4.7). Choosing some η such that $0 \leq \eta \leq 1$ the bound

$$\mathbf{R} \leq \mathbf{R}_{\text{emp}} + \sqrt{\frac{h(\log(2N/h) + 1) - \log(\eta/4)}{N}} \tag{4.11}$$

holds with a probability $1-\eta$. The right side of equation (4.11) consists of two quantities. The first summand is the empirical risk given by equation (4.8) and the second summand is the so called **VC confidence**. Given a fixed training set and a set of function of a high complexity, the learning algorithm will be adapted to the training data and the empirical error will tend to zero. Of course, the high complexity also requires a high VC dimension and so the VC confidence of equation (4.11) increases.

The principle of **Structural Risk Minimization** (SRM) is to control the two quantities R_{emp} and VC confidence. While the empirical risk depends on

the function chosen by the training procedure, the VC confidence depends on the chosen class of functions. To control the VC dimension, a structure is introduced by dividing the set of functions $\mathcal{F}$ into nested subsets $\mathcal{F}_k$:

Create subsets $\mathcal{F}_k$ of $\mathcal{F}$ with $k \in \mathbb{N}$

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset \cdots \subset \mathcal{F}_k \subset \ldots \tag{4.12}$$

whose VC dimensions result in

$$h(\mathcal{F}_1) \leq h(\mathcal{F}_2) \leq \cdots \leq h(\mathcal{F}_k) \leq \ldots \tag{4.13}$$

and proceed as follows: Find the function f_k in the each subset $\mathcal{F}_k$ which minimizes the empirical risk. Choose then the specific function for which the sum of empirical risk and VC confidence is minimal.

We can summarize that the SRM principle leads to tradeoff between the complexity of the approximated function and the quality of the approximation. In section 5.3 we will present a popular example of this principle, namely the Support Vector Machine. But beforehand we need to review some more fundamentals of statistical learning theory, like the kernel trick and the reproducing kernel Hilbert space.

4.5 Kernel Functions

4.5.1 Feature Space

For linear classification functions it is not possible to discriminate classes where the samples are distributed in a non-linearly separable way, which is normally the case for real world data. Figure 4.3 illustrates a simple example of two classes that can not be separated by a linear function.

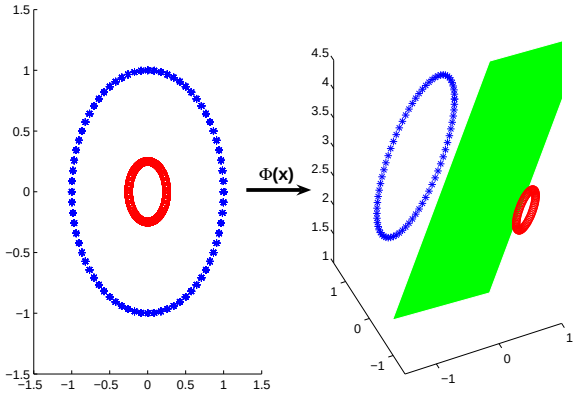

Figure 4.3: Example of a non-linear decision surface in the 2-dimensional space. Via a non-linear mapping Φ into the 3-dimensional feature space the samples can be separated by a linear hyperplane.

In this 2-dimensional space we have to create a decision region in the form of a circle or other complex functions. On the other hand it might be helpful to map the samples into a 3-dimensional space by a function Φ:

$$\Phi : \mathbb{R}^2 \rightarrow F = \mathbb{R}^3. \tag{4.14}$$

In our example we choose the non-linear map in the form of monomials of degree 2:

$$(x_1, x_2) \rightarrow (x_1^2, \sqrt{2}x_1x_2, x_2^2). \tag{4.15}$$

From figure 4.3 it is clear that the two classes can be separated in the 3-dimensional feature space by a linear hyperplane. This explicit transformation works fine for small and low dimensional datasets, but if we want to use M-dimensional features and monomials of degree d the resulting feature space blasts to $\binom{M+d-1}{d}$ dimensions. If the samples occur only in the form of dot-

products then the dot-product of the samples in the feature space is computed by $\Phi(\boldsymbol{x})\Phi(\boldsymbol{z})$.

$$\begin{aligned}\Phi(\boldsymbol{x})\Phi(\boldsymbol{z}) &= (x_1^2, \sqrt{2}x_1x_2, x_2^2)(z_1^2, \sqrt{2}z_1z_2, z_2^2)^\top \\ &= x_1^2z_1^2 + 2x_1x_2z_1z_2 + x_2^2z_2^2 \\ &= \left((x_1, x_2)(z_1, z_2)^\top\right)^2 \\ &= (\boldsymbol{x} \cdot \boldsymbol{z}^\top)^2 \end{aligned} \tag{4.16}$$

The results of equation (4.16) allows us to compute the dot-product of $\Phi(\boldsymbol{x})\Phi(\boldsymbol{z})$ in feature space F without explicit mapping the samples into F. Such a direct computation method of the dot-product is called **kernel** function.

The consequence of this simple example leads to the reverse question, which kernel function k corresponds to a dot-product in feature space F.

4.5.2 Reproducing Kernel Hilbert Space

Suppose we are given a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{X}$. Then the $N \times N$ matrix K with entries $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is defined as the **Gram** matrix of k. The matrix K is called positive definite for all $\boldsymbol{x}_i \in \mathbb{R}$, if for any $\alpha_1, \ldots, \alpha_N \in \mathbb{R}$

$$\sum_{ij}^{N} \alpha_i\alpha_j K_{ij} \geq 0 \tag{4.17}$$

holds. The function k is also called a positive definite kernel, iff for all $N \in \mathbb{N}$ and all $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{X}$ the Gram matrix of k is positive definite. Since the kernel matrix K is also symmetric, the following equation holds:

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(\boldsymbol{x}_j, \boldsymbol{x}_i). \tag{4.18}$$

If k is a real positive definite kernel and $\mathcal{X} \neq \emptyset$, we define the mapping from $\mathcal{X}$ into the function space $\mathcal{H} = \mathbb{R}^{\mathcal{X}} = \{f : \mathcal{X} \to \mathbb{R}\}$ by

$$\Phi : \mathcal{X} \to \mathbb{R}^{\mathcal{X}}, \boldsymbol{x} \to k(\cdot, \boldsymbol{x}). \tag{4.19}$$

One can show that the set of all linear combinations

$$f(\cdot) = \sum_{i}^{N} \alpha_i k(\cdot, \boldsymbol{x}_i) \quad \text{with any } N \in \mathbb{N}, \alpha_i \in \mathbb{R}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{X} \tag{4.20}$$

forms a vector space. For this reason we construct a vector space by the linear combination of the form

$$g(\cdot) = \sum_{j}^{N'} \beta_j k(\cdot, \boldsymbol{x}_j^{'}) \quad \text{with any } N' \in \mathbb{N}, \beta_j \in \mathbb{R}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N'} \in \mathcal{X}, \tag{4.21}$$

and write the dot product of the two functions $f(\cdot)$ and $g(\cdot)$ as

$$\langle f, g \rangle = \sum_{i}^{N} \sum_{j}^{N'} \alpha_i \beta_j k(\boldsymbol{x}_i, \boldsymbol{x}_j^{'}) \tag{4.22}$$

$$= \sum_{i}^{N} \alpha_i g(\boldsymbol{x}_j) = \sum_{j}^{N'} \beta_j f(\boldsymbol{x}_j^{'}), \tag{4.23}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product in some Hilbert space $\mathcal{H}$. The positive definite kernel k implies, that for function f

$$\langle f, f \rangle = \sum_{i,j=1}^{N} \alpha_i \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0 \tag{4.24}$$

holds. Equation (4.24) means, that the dot product is a positive definite kernel by itself. It follows that the value $f(\boldsymbol{x})$ of a function $f \in \mathcal{H}$ can be expressed as a dot product in $\mathcal{H}$ by

$$f(\boldsymbol{x}) = \langle k(\cdot, \boldsymbol{x}), f \rangle \tag{4.25}$$

$$\text{and in particular } k(\boldsymbol{x}, \boldsymbol{x}^{'}) = \langle k(\cdot, \boldsymbol{x}), k(\cdot, \boldsymbol{x}^{'}) \rangle. \tag{4.26}$$

Equation (4.26) is the reason why positive definite kernels are also called reproducing kernels: they reproduce the evaluation of f on $\boldsymbol{x}$. It also shows that the kernel k indeed computes the dot product in F for $\Phi(\boldsymbol{x})$. Hence (4.19) is one possible realization of the mapping associated with a kernel and is called the feature map. The following definition of a Reproducing Kernel Hilbert Space is based on the work of [Aronszajn, 1950].

Definition 4.2 (Reproducing Kernel Hilbert Space) *Suppose we are given a not-empty set $\mathcal{X}$ and a Hilbert space $\mathcal{H}$ $f : \mathcal{X} \to \mathbb{R}$. Then $\mathcal{H}$ is called a* **reproducing kernel Hilbert space** *(RKHS) with dot product $\langle \cdot, \cdot \rangle$ and norm $\|f\| = \sqrt{\langle f, f \rangle}$, iff there exists a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$*

1. *satisfying $f(\boldsymbol{x}) = \langle f, k(x, \cdot) \rangle \quad \forall f \in \mathcal{H}$*
2. *spanning $\mathcal{H}$, i.e., $\mathcal{H} = span\ \{k(\boldsymbol{x}, \cdot) | \boldsymbol{x} \in \mathcal{X}\}$.*

One can show, that the kernel k for such a RKHS is uniquely determined, a complete overview of the concepts of RKHS can be found in [Aronszajn, 1950].

In [Vapnik, 2000] Mercer's Theorem is used as an alternative way to identify a feature space F together with an associated kernel function. Mercer's Theorem states that if the kernel function k gives rise to a positive integral operator, the kernel can be expressed as an absolute and uniformly convergent series. Additionally, if the set $\mathcal{X}$ on which the kernel is defined, is also compact, a kernel is a Mercer kernel iff it is a positive definite kernel.

Theorem 4.3 (Mercer's Theorem [Mercer, 1909]) *Suppose k is a continuous symmetric function such that the integral operator T_k*

$$T_k : L_2(\mathcal{X}) \to L_2(\mathcal{X}), \quad (T_k f)(\boldsymbol{x}) = \int_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{z}) f(\boldsymbol{z}) d\boldsymbol{z} \tag{4.27}$$

is positive, that is

$$\int_{\mathcal{X} \times \mathcal{X}} k(\boldsymbol{x}, \boldsymbol{z}) f(\boldsymbol{x}) f(\boldsymbol{z}) d\boldsymbol{x} d\boldsymbol{z} \geq 0 \tag{4.28}$$

for all $f \in L_2(\mathcal{X})$ and $\mathcal{X}$ being a compact subset of $\mathbb{R}^d$. Then there exists an orthonormal basis consisting of eigenfunctions $\Phi_j \in L_2(\mathcal{X})$ such that the associated sequence of eigenvalues λ_j is non-negative and the kernel function can be written as

$$k(\boldsymbol{x}, \boldsymbol{z}) = \sum_{j=1}^{M} \lambda_j \Phi_j(\boldsymbol{x}) \Phi_j(\boldsymbol{z}), \tag{4.29}$$

where $M < \infty$.

If we construct a mapping function Φ and choose a feature space $F = L_2$ such as

$$\Phi : \boldsymbol{x} \to \left(\sqrt{\lambda_1} \Phi_1(\boldsymbol{x}), \sqrt{\lambda_2} \Phi_2(\boldsymbol{x}), \dots \right), \tag{4.30}$$

we see from equation (4.29) that the Mercer kernel k corresponds to the dot-product in L_2.

Common kernel functions satisfying Mercer's condition are the **linear**, the **polynomial** and the **Radial Basis Function** (RBF) kernel. An overview is given in table 4.1. More sophisticated kernels designed for special applications can be found in [Vapnik, 2000], [Jaakkola and Haussler, 1999b], [Tsuda et al., 2002] and [Shawe-Taylor and Cristianni, 2004].

Table 4.1: Widely used kernel functions that are known to fulfill Mercer's condition: the linear, the polynomial and Gaussian RBF kernel.

$$\textbf{linear kernel} \quad k(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x}^\top \boldsymbol{z} \tag{4.31}$$

$$\textbf{polynomial kernel} \quad k(\boldsymbol{x}, \boldsymbol{z}) = ((a\boldsymbol{x}^\top \boldsymbol{z}) + b)^d \tag{4.32}$$

$$\textbf{RBF kernel} \quad k(\boldsymbol{x}, \boldsymbol{z}) = \exp\left(\frac{-\|\boldsymbol{x} - \boldsymbol{z}\|^2}{2\sigma^2}\right) \tag{4.33}$$

4.6 Regularization

We have shown in 4.2 that the minimization of the empirical risk $\mathbf{R}_{\text{emp}}$ does not lead to an optimal solution. One possible method to overcome this problem is to restrict the space of possible solutions to compact subsets of the original function space $\mathcal{F}$. This is achieved by minimizing the following objective function:

$$\mathbf{R}_{\text{reg}}(f) = \mathbf{R}_{\text{emp}}(f) + \lambda\Omega(f). \tag{4.34}$$

Here $\mathbf{R}_{\text{emp}}(f)$ is the empirical risk (4.8) and $\lambda\Omega(f)$ the regularization term. The influence of the regularizer is controlled by λ. If we set $\lambda = 0$ then equation (4.34) is reduced to the unregularized problem. Otherwise setting $\lambda \to \infty$ only the regularizer $\Omega(f)$ is minimized. Usually $\mathbf{R}_{\text{emp}}(f)$ and $\Omega(f)$ are chosen to be convex, since this ensures a global minimum of $\mathbf{R}_{\text{reg}}(f)$. In the next subsection we show that regularization is not only restricted to the input space of our data, but also holds in a reproducing kernel Hilbert space.

4.6.1 Representer Theorem

The Representer Theorem is based on the work of [Kimeldorf and Wahba, 1971] and is expressed in the following theorem.

Theorem 4.4 (Representer Theorem [Kimeldorf and Wahba, 1971]) *Let $l(f(\boldsymbol{x}_i), y_i)$ be an arbitrary loss function and $\Omega(\|f\|)$ any strictly monotonically increasing function. If $f \in \mathcal{F}$ is a minimizer of the regularized risk function*

$$\sum_{i=1}^{N} l(f(\boldsymbol{x}_i), y_i) + \lambda\Omega(\|f\|) \tag{4.35}$$

with $\lambda > 0$, then there exists a vector $\boldsymbol{\alpha} \in \mathbb{R}^N$ such that

$$f(\cdot) = \sum_{i=1}^{N} \alpha_i k(\cdot, \boldsymbol{x}_i), \tag{4.36}$$

i.e., $f \in span\{k(\boldsymbol{x}_1, \cdot), \ldots, k(\boldsymbol{x}_N, \cdot)\}$.

Any function f in RKHS can be decomposed into two parts,

$$f : f_{\|} + f_{\perp} \tag{4.37}$$

where $f_{\|}$ is contained in the linear span of $k(\boldsymbol{x}_1, \cdot), \ldots, k(\boldsymbol{x}_N, \cdot)$ and $f_{\perp}$ is in the orthogonal complement, i.e., $\langle f_{\|}, f_{\perp} \rangle = 0$. By the reproducing property of the kernel K in the native space, see section 4.5.2 we have

$$\begin{aligned} f^{\star}(\boldsymbol{x}_k) &= \langle f(\cdot), k(\boldsymbol{x}_k, \cdot) \rangle \\ &= \sum_{i=1}^{N} \alpha_i \langle k(\boldsymbol{x}_i, \cdot) \rangle + \langle f_{\perp}, k(\boldsymbol{x}_k, \cdot) \rangle \\ &= \sum_{i=1}^{N} \alpha_i k(\boldsymbol{x}_k, \boldsymbol{x}_i) \end{aligned} \tag{4.38}$$

Consequently, every solution admits a representation of the form (4.35). The Representer Theorem allows for a quite general class of learning algorithms to be applied in the feature space. Examples of these algorithms are the Support Vector Machine, section 5.3 and the kernel logistic regression, section 5.5.

4.7 Summary

In this chapter we discussed the main ideas of statistical learning theory. We used the VC-dimension as capacity measure and showed how the actual risk is bounded by the VC-confidence. This led us to an induction principle called structural risk minimization.

To overcome the problem of only using linear functions we introduced the idea of kernel functions. These functions allow us to compute a dot-product in feature space F without explicit mapping the samples into that space. Finally, we described the method of regularization as another approach of structural risk minimization.

This chapter laid the necessary foundations of statistical learning theory for the classifiers which we will present in the next chapter.

5 Discriminative Classifiers

In supervised learning it is the goal to find a rule which assigns a sample $\boldsymbol{x}$ to a label y. Suppose we are given a set of N labeled input/output pairs $\{\boldsymbol{x}_i, y_i\}$, $i = 1, ..., N$, where $\boldsymbol{x}_i$ is denoted as the input sample in $\mathbb{R}^d$ and y_i as the associated output label with $y_i \in \mathbb{R}$ for regression and $y_i \in \{+1, -1\}$ for classification.

As we have seen in chapter 4 we have to find a function f from the set of functions $\mathcal{F}$. From this set $\mathcal{F}$ we choose as a particular (but not necessarily optimal) option linear functions $f(\boldsymbol{x}_i)$ that are also called **linear learning machines** in the field of machine learning.

5.1 Linear Regression

Linear regression attempts to model the relationship between variables $\boldsymbol{x}_i$ and y_i. For each pair of training examples we have to find a linear function

$$f(\boldsymbol{x}_i) = \sum_{j=1}^{d} \beta_j x_{i,j} + \beta_0 \tag{5.1}$$

that best predicts the label y_i from the sample $\boldsymbol{x}_i$.

Typically, that function $f(\boldsymbol{x}_i)$ is chosen, that minimizes the sum of the squares of the distances between the training points and the regression line. This technique is called **least squares** and is given by

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{N} (y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)^2 \tag{5.2}$$

where each training sample is augmented by a constant entry of one, i.e., $\boldsymbol{x}_i = (1, x_{i,1}, \ldots, x_{i,d})^\top$ and β_0 is added to the parameter vector $\boldsymbol{\beta}$. The function $l(\boldsymbol{\beta})$ is called the square loss function (see section 4.1). For a more compact notation we define $\boldsymbol{X}$ as the $(N \times d)$ matrix of training vectors $\boldsymbol{x}_i$, i.e., $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$.

The minimization of $l(\boldsymbol{\beta})$ is realized by differentiating $l(\boldsymbol{\beta})$ with respect to the parameter vector $\boldsymbol{\beta}$ and setting the result to zero:

$$\frac{\partial L}{\partial \boldsymbol{\beta}} = -2\boldsymbol{X}\boldsymbol{y} + 2\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{\beta} = 0. \tag{5.3}$$

If the inverse of $\boldsymbol{X}^\top \boldsymbol{X}$ exists, the solution of the least squares problem is

$$\boldsymbol{\beta} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}\boldsymbol{y}. \tag{5.4}$$

If the matrix $\boldsymbol{X}^\top \boldsymbol{X}$ is singular one can use a solution that was proposed by Hoerl and Kennard [Hoerl and Kennard, 1970], known as **ridge regression**. Ridge regression chooses a function that minimizes a combination of the square loss and the L_2-norm of the parameter vector $\boldsymbol{\beta}$. The influence of this kind of regularization is controlled by the regularization parameter λ (section 4.6). The ridge regression version of equation (5.2) reads then:

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{N} (y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)^2 + \lambda \|\boldsymbol{\beta}\|. \tag{5.5}$$

The minimization of (5.5) leads to

$$\boldsymbol{\beta} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}\boldsymbol{y}, \tag{5.6}$$

with the identity matrix $\boldsymbol{I}$. The result of the regularization term is that an adequate value λ is added on the main diagonal of the matrix $\boldsymbol{X}^\top \boldsymbol{X}$. If λ is sufficiently large enough the matrix becomes positive definite.

5.2 Linear Classification

Suppose we are given a set of N labeled input/output pairs $\{\boldsymbol{x}_i, y_i\}$, $i = 1, ..., N$, where $\boldsymbol{x}_i$ is an input vector in $\mathbb{R}^d$ and y_i the corresponding class label with $y_i \in \{+1, -1\}$.

Using the linear function of equation (5.1) we have to map the real values of $f(\boldsymbol{x}_i)$ to ± 1. This is realized by applying the signum function to $f(\boldsymbol{x}_i)$. We now assign the input vector to the positive class, if $\text{sign}(f(\boldsymbol{x}_i)) \geq 0$ and otherwise to the negative class.

For the special case where $f(\boldsymbol{x}_i) = 0$ the resulting hyperplane is called **decision boundary**. The parameters of the classifier can be estimated by

several learning methods, popular ones are **linear discriminants** [Fisher, 1936] and **perceptrons** [Rosenblatt, 1958]. Other methods are described in [Duda et al., 2001, Hastie et al., 2001] or [Wendemuth, 1994] for correct estimation of the margin-optimal normal vector and threshold.

In this thesis we focus on two learning machines, namely the **Support Vector Machine** (SVM) and the **Logistic Regression** (LR), that are based on the theoretical concepts given in chapter 4.

5.3 Support Vector Machines

The **Support Vector Machine** (SVM) developed by Vapnik [Vapnik, 1998] and others [Boser et al., 1992, Cortes and Vapnik, 1995, Schölkopf, 1997] is one of the most successful and popular classification techniques over the last decade. SVMs optimize their parameters by minimizing the classification error using the concept of structural risk minimization (see section 4.4). Structural risk minimization strikes a balance between the empirical risk and the complexity of the mapping function.

The decision hyperplane of the binary classification problem can be expressed as

$$\boldsymbol{w}^\top \boldsymbol{x}_i + b = 0 \tag{5.7}$$

$$\boldsymbol{w}^\top \boldsymbol{x}_i + b \geq +1 \qquad y_i = +1 \qquad i = 1, ..., N, \tag{5.8}$$

$$\boldsymbol{w}^\top \boldsymbol{x}_i + b \leq -1 \qquad y_i = -1 \qquad i = 1, ..., N, \tag{5.9}$$

where b is the distance of the hyperplane from the origin. The margin of the SVM is defined as the distance from the separating hyperplane to the closest positive and negative samples. If the equality condition of equation (5.7) holds for the data point $\boldsymbol{x}$ then $\boldsymbol{x}$ lies on the separating hyperplane. Therefore the marginal hyperplane can be characterized by

$$\boldsymbol{w}^\top \boldsymbol{x} + b = \pm 1. \tag{5.10}$$

Additionally, the points $\boldsymbol{x}_i$ that satisfy

$$\boldsymbol{w}^\top \boldsymbol{x}_+ + b = +1 \quad \text{and} \quad \boldsymbol{w}^\top \boldsymbol{x}_- + b = -1 \tag{5.11}$$

will fall on the two hyperplanes parallel to the decision plane and orthogonal

to $\boldsymbol{w}$. Subtracting (5.11, left) from (5.11, right) results in

$$\boldsymbol{w}^\top(\boldsymbol{x}_+ - \boldsymbol{x}_-) = 2, \tag{5.12}$$

$$\left(\frac{\boldsymbol{w}^\top}{\|\boldsymbol{w}\|}\right)(\boldsymbol{x}_+ - \boldsymbol{x}_-) = \frac{2}{\|\boldsymbol{w}\|}, \tag{5.13}$$

where we normalize the margin by the norm of the normal vector to the hyperplane and $\frac{1}{\|\boldsymbol{w}\|}$ is the distance d between the two parallel hyperplanes:

$$2d = \frac{2}{\|\boldsymbol{w}\|}. \tag{5.14}$$

This distance provides an objective measure of how separable the two classes of training data are. The distance d can also be considered as a safety margin of the classifier.

The combination of the constraints (5.8) and (5.9) gives the single constraint

$$y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 \qquad \forall_i = 1, ..., N. \tag{5.15}$$

During the optimization process the objective is to find vectors $\boldsymbol{x}_+$ and $\boldsymbol{x}_-$ that maximize the margin of separation. These training vectors are called **support vectors**. Instead of minimizing equation (5.14) it is possible to minimize the complexity term by minimizing $\frac{1}{2}\|\boldsymbol{w}\|^2$. This leads to the quadratic optimization problem

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2, \tag{5.16}$$

subject to equation (5.15).

5.3.1 Dual Formulation

We now transform the optimization process into its corresponding dual formulation. Introducing the Lagrange multipliers α_i gives the primal Lagrangian

$$L(\boldsymbol{w}, b, \alpha) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i(y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) - 1) \tag{5.17}$$

where $L(\boldsymbol{w}, b, \alpha)$ has to be minimized with respect to $\boldsymbol{w}$ and b and maximized with respect to α. For the minimization the partial derivations lead to the

Karush-Kuhn-Tucker (KKT) conditions

$$\frac{\partial L(\boldsymbol{w}, b, \alpha)}{\partial b} = 0 \;\Rightarrow\; \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{5.18}$$

$$\frac{\partial L(\boldsymbol{w}, b, \alpha)}{\partial \boldsymbol{w}} = 0 \;\Rightarrow\; \boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i. \tag{5.19}$$

Substituting (5.18) in (5.17) results in the dual formulation of the optimization problem:

$$\begin{aligned} L_D(\boldsymbol{w}, b, \alpha) &= \frac{1}{2}(\boldsymbol{w}^\top \boldsymbol{w}) - \sum_{i=1}^{N} \alpha_i y_i (\boldsymbol{w}^\top \boldsymbol{x}_i) - \sum_{i=1}^{N} \alpha_i y_i b + \sum_{i=1}^{N} \alpha_i \\ &= \frac{1}{2} \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \sum_{j=1}^{N} \alpha_j y_j \boldsymbol{x}_j - \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \sum_{j=1}^{N} \alpha_j y_j \boldsymbol{x}_j + \sum_{i=1}^{N} \alpha_i \\ &= \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i^\top \boldsymbol{x}_j) \end{aligned} \tag{5.20}$$

which finally leads to maximization of L_D with respect to α_i:

$$\max_{\alpha} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i^\top \boldsymbol{x}_j) \tag{5.21}$$

$$\text{subject to} \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{5.22}$$

$$\alpha_i \geq 0, \quad i = 1, \ldots, N \tag{5.23}$$

Substituting (5.19) in (5.7) leads to the decision function

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i^\top \boldsymbol{x} + b. \tag{5.24}$$

If a binary decision $(-1, +1)$ is required, this is achieved by $y = sign(f(\boldsymbol{x}))$.

5.3.2 The Non-Separable Case

The SVM algorithm so far will find no feasible solution if it is applied to linear non-separable problems. There are several ways of dealing with this problem, [Wendemuth, 1995a,b]. The given error measures in [Wendemuth, 1995b] lead to non-convex solutions which are believed to be NP-complete. In any case, they have no global optimum solution and hence require iterative solution approaches.

In order to overcome these problems, a new error measure is introduced, using **slack variables**. This error measure is a compromise since it allows a different slack for each $\boldsymbol{x}_i$ and aims at minimizing the sum of all these slacks, not the maximum. The compromise is achieved by controlling the influence of the slack variables with a new, global parameter C. The constraints of the optimization problem then become:

$$y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i \tag{5.25}$$

$$\xi_i \geq 0 \quad i = 1, \ldots, M. \tag{5.26}$$

The optimization problem by itself has to be expanded by minimizing $\sum_{i=1}^{N} \xi_i$, which leads to the modified optimization problem:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \xi_i \tag{5.27}$$

subject to equation (5.25). The parameter $C > 0$ controls the influence of the summed slack variables with respect to the optimization problem. In other words, it is a tradeoff between the capacity and accuracy of the model. The resulting Lagrangian now reads

$$\begin{aligned} L(\boldsymbol{w}, b, \alpha, \xi, \gamma) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \xi_i \\ - \sum_{i=1}^{N} \alpha_i \left(y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) - 1 + \xi_i\right) - \sum_{i=1}^{N} \gamma_i \xi_i, \end{aligned} \tag{5.28}$$

where γ_i is the Lagrangian multiplier of the slack variables ξ_i. The corresponding KKT conditions for the primal problem are found by optimizing

equation (5.28) with respect to $\boldsymbol{w}$, b and ξ_i:

$$\frac{\partial L(\boldsymbol{w}, b, \alpha, \xi, \gamma)}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \qquad = 0 \tag{5.29}$$

$$\frac{\partial L(\boldsymbol{w}, b, \alpha, \xi, \gamma)}{\partial b} = -\sum_{i=1}^{N} \alpha_i y_i \qquad = 0 \tag{5.30}$$

$$\frac{\partial L(\boldsymbol{w}, b, \alpha, \xi, \gamma)}{\partial \xi_i} = C - \alpha_i - \gamma_i \qquad = 0 \tag{5.31}$$

$$\alpha_i \geq 0 \tag{5.32}$$

$$\gamma_i \geq 0 \tag{5.33}$$

$$\alpha_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b)y_i - 1 + \xi_i = 0 \tag{5.34}$$

$$\gamma_i \xi_i = 0 \tag{5.35}$$

If $\alpha_i < C$ it follows from equation (5.31) together with equation (5.35) that $\xi_i = 0$. This is possible in 2 cases: a) the problem is linearly separable or b) the problem is non-linearly separable, but for data $\boldsymbol{x}_i$, no slack variable is needed, i.e. the solution under inspection will already classify $\boldsymbol{x}_i$ correctly. Obviously, this cannot hold for all data $\boldsymbol{x}_i$. Finally, substituting equations (5.29), (5.31) and (5.34) into the Lagrangian, the dual optimization problem becomes:

$$\max_{\alpha} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i^\top \boldsymbol{x}_j) \tag{5.36}$$

$$\text{subject to} \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{5.37}$$

$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots, N. \tag{5.38}$$

Setting $C \to \infty$ leads to the same optimization problem as in the linear separable case.

5.3.3 Non-Linear Support Vector Machines using Kernels

We are now able to transform the linear SVM into the feature space either by an explicit mapping Φ or by kernel functions. Using a kernel function k the

linear dot product $(\boldsymbol{x}_i \boldsymbol{x}_j)$ in the dual optimization problem of equation (5.36) is replaced by a kernel $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$:

$$\max \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{5.39}$$

$$\text{subject to} \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{5.40}$$

$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots, N. \tag{5.41}$$

5.3.4 Probabilistic Outputs for Support Vector Machines by a Sigmoid Function

As already mentioned, the output $f(\boldsymbol{x})$ of the SVM is a distance measure to the decision boundary, which is also called unmoderated output of the SVM.

To interprete this distance as a posterior probability, the unmoderated SVM output has to be transformed into the $[0, 1]$ interval by an adequate method.

One method of generating probabilistic outputs from this kernel machine was proposed by Hastie [Hastie and Tibshirani, 1998]. The idea is to fit two Gaussians to the class-conditional densities $p(f(\boldsymbol{x})|y = 1)$ and $p(f(\boldsymbol{x})|y = -1)$. The posterior probabilities can then be computed by Bayes' rule:

$$P(y = 1|f(\boldsymbol{x})) = \frac{p(f(\boldsymbol{x})|y = 1)P(y = 1)}{p(f(\boldsymbol{x})|y = -1)P(y = -1) + p(f(\boldsymbol{x})|y = 1)P(y = 1)} \tag{5.42}$$

The most popular method of transforming unmoderated SVM outputs to posterior probabilities was proposed by Platt [Platt, 2000]. Instead of estimating the class-conditional densities Platt suggested to fit the posterior probability $P(y = 1|f(\boldsymbol{x}))$ directly. He considered two exponential functions and used the Bayes' rule of equation (5.42) to transform the SVM outputs to posterior probabilities by a sigmoid function:

$$P(y = +1|\boldsymbol{x}) = \frac{1}{1 + \exp(Af(x) + B)}. \tag{5.43}$$

where the parameters A and B have to be estimated on an adequate training set.

5.3.5 Probabilistic Outputs for Support Vector Machines by Isotonic Regression

A more general calibration of the SVM outputs is given by the isotonic regression. This method is only restricted to the assumption that the calibration function is an isotonic function dependent on the SVM output $f(x)$. An isotonic function implies a strict increase of function values. The isotonic regression is computed by the pair-adjacent violators (PAV) algorithm [Ayer et al., 1955] that finds a stepwise constant solution according to the mean-squared error criterion, see also [Zadrozny and Elkan, 2002]. Assuming training examples $\boldsymbol{x}_i$ and corresponding function values $g(\boldsymbol{x}_i)$ that give the values to be learned the PAV algorithm estimates the isotonic regression $\hat{g}$. If g is not already isotonic, it follows that $g(\boldsymbol{x}_{i-1}) \geq g(\boldsymbol{x}_i)$ where the examples $\boldsymbol{x}_{i-1}$ and $\boldsymbol{x}_i$ are called pair-adjacent violators. All these pair-adjacent violators are now replaced by their average to fulfill the isotonic assumption.

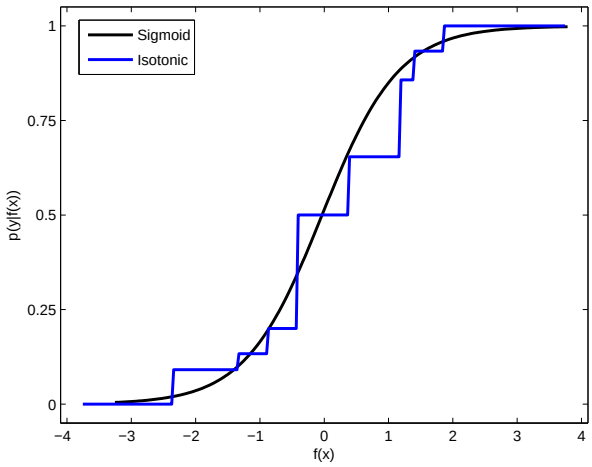

Figure 5.1: Comparison of the sigmoid and the isotonic score calibration on a artifical dataset.

In figure 5.1 we show the behavior of the two different calibration methods on an artificial dataset. While the parametric assumption of the sigmoid fitting leads to the sigmoid shape of the curves, the isotonic fit shows that the function just transforms the data to increasing values.

5.4 Logistic Regression

The idea of **Logistic Regression** (LR) is to model the posterior probability of a classification problem via a linear regression function. Therefore we discussed the simple regression model in section 5.1, where the regression function $f(\boldsymbol{x})$ has a real valued output. The transition from linear to Logistic Regression is a transition from the approximation of a continuous function to a binary classification problem.

We define the response of the model to be $y = 1$ denoting the occurrence of the event of interest and $y = 0$ otherwise. Furthermore, we are not interested in a hard yes/no decision of the classifier, but we want to estimate the probability of one of these two states. Using the ordinary linear regression model of equation (5.1) for the probability estimation with $p(y = 1) = f(x)$ leads to values that clearly can exceed 1 or are negative. The range for regression lies between $[-\infty; +\infty]$ and so we have to normalize the success probability $p(y_i = 1)$ to the range of $[0, 1]$. Therefore we first introduce the so called odds:

$$odds_i = \frac{p(y_i = 1)}{(1 - p(y_i = 1))} \tag{5.44}$$

with the upper limit for the range $p \to 1$:

$$\lim_{p \to 1}(\frac{p}{1-p}) = +\infty. \tag{5.45}$$

Using the logarithm of the odds, the log odds, the lower limit of the range $p \to 0$ is given by

$$\lim_{p \to 0} \ln(\frac{p}{1-p}) = -\infty. \tag{5.46}$$

This logarithm of the odds is called the logit. As discussed earlier we restrict the model complexity to linear functions and model the logit by:

$$\begin{aligned} logit(p(y_i = 1)) = \ln\left[\frac{p(y_i = 1)}{(1 - p(y_i = 1))}\right] &= \beta_0 + \beta_1 \cdot x_{1,i} + \cdots + \beta_n \cdot x_{n,i} \\ &= \beta_0 + \tilde{\boldsymbol{\beta}}^\top \tilde{\boldsymbol{x}}_i \end{aligned} \tag{5.47}$$

To get rid of the constant β_0 it is useful to augment each example vector by a constant entry of one, i.e. $\boldsymbol{x}_i = (1, x_1, \ldots, x_n)^\top$ and $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_n)^\top$.

Then we can rewrite (5.47) as

$$logit(p(y_i = 1)) = \boldsymbol{\beta}^\top \boldsymbol{x}_i. \tag{5.48}$$

From 5.47 follows

$$p(\boldsymbol{x}_i, \boldsymbol{\beta}) = \exp(\boldsymbol{\beta}^\top \boldsymbol{x}_i) - p(\boldsymbol{x}_i, \boldsymbol{\beta}) \exp(\boldsymbol{\beta}^\top \boldsymbol{x}_i) \tag{5.49}$$

$$= \frac{\exp(\boldsymbol{\beta}^\top \boldsymbol{x}_i)}{1 + \exp(\boldsymbol{\beta}^\top \boldsymbol{x}_i)} \tag{5.50}$$

$$= \frac{1}{1 + \exp(-\boldsymbol{\beta}^\top \boldsymbol{x}_i)}. \tag{5.51}$$

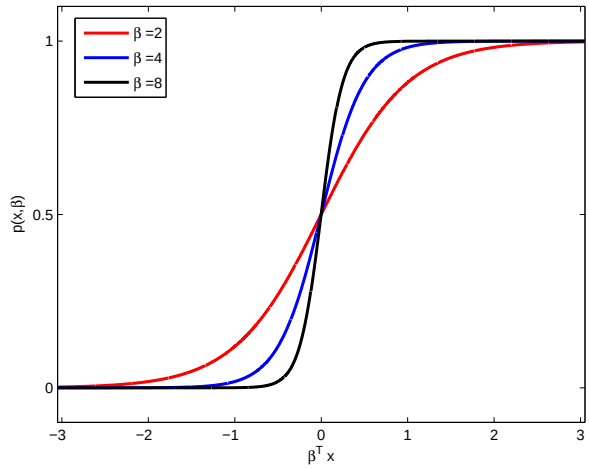

Figure 5.2: Output of the logistic function. Comparison of three different weights β.

The function in equation (5.51) is called the **logistic function** and is plotted in figure 5.2. As can be seen in figure 5.2 the logistic function (or logistic curve) is a monotonic, continuous function, bounded between 0 and 1. The shape of this curve depends on the coefficient $\boldsymbol{\beta}$, for small coefficients the logistic curve converge to a straight line.

5.4.1 Fitting Logistic Regression Models

For binary outcomes, the distribution of the output y_i is a binomial distribution. The binomial distribution is the discrete probability distribution of

the number of successes in a sequence of N independent yes/no experiments, each of which yields success with probability p. Such a success/failure experiment is also called a Bernoulli trial. If we assume that the training data are drawn from such a binomial distribution conditioned on the samples $\boldsymbol{x}_i$, the conditioned probability of $P(y_i|\boldsymbol{\beta}, \boldsymbol{x}_i)$ is

$$P(y_i|\boldsymbol{x}_i, \boldsymbol{\beta}) = \begin{cases} P(\boldsymbol{x}_i, \boldsymbol{\beta}) & y_i = 1 \\ 1 - P(\boldsymbol{x}_i, \boldsymbol{\beta}) & y_i = 0 \end{cases} \tag{5.52}$$

$$= P(\boldsymbol{x}_i, \boldsymbol{\beta})^{y_i}(1 - P(\boldsymbol{x}_i, \boldsymbol{\beta}))^{1-y_i}. \tag{5.53}$$

The estimation of the unknown parameter vector $\boldsymbol{\beta}$ can be realized by **maximum likelihood estimation** (MLE), where the **likelihood function** $L(\boldsymbol{\beta}) = \prod_{i=1}^{N} P(y_i|\boldsymbol{x}_i, \boldsymbol{\beta})$ is maximized. Usually the **log-likelihood function** is chosen to transform the product into a sum, because in any case, the same values will maximize both and it is easier to deal with sums than with products.

However, instead of a maximization process we choose the minimization of the **negative log-likelihood** (NLL) of equation (5.53)

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{N} y_i \log[p(\boldsymbol{x}_i, \boldsymbol{\beta})](1 - y_i) \log[1 - p(\boldsymbol{x}_i, \boldsymbol{\beta})] \tag{5.54}$$

$$= \sum_{i=1}^{N} y_i \left(\log[p(\boldsymbol{x}_i, \boldsymbol{\beta})] - \log[1 - p(\boldsymbol{x}_i, \boldsymbol{\beta})]\right) + \log[1 - p(\boldsymbol{x}_i, \boldsymbol{\beta})] \tag{5.55}$$

$$= \sum_{i=1}^{N} y_i \left(\log[\frac{p(\boldsymbol{x}_i, \boldsymbol{\beta})}{1 - p(\boldsymbol{x}_i, \boldsymbol{\beta})}]\right) + \log[\frac{1}{1 + \exp(\boldsymbol{\beta}^\top \boldsymbol{x}_i)}]. \tag{5.56}$$

And this finally results in

$$L(\boldsymbol{\beta}) = -\sum_{i=1}^{N} y_i \boldsymbol{\beta}^\top \boldsymbol{x}_i + \log\left(1 + \exp(\boldsymbol{\beta}^\top \boldsymbol{x}_i)\right). \tag{5.57}$$

To avoid over-fitting to the training data it is necessary to impose a penalty on large fluctuations of the estimated parameters $\boldsymbol{\beta}$. As described in section 5.1 the most popular method is the ridge penalty $\frac{\lambda}{2}\|\boldsymbol{\beta}\|^2$ that was introduced by [Hoerl and Kennard, 1970]. This quadratic regularization is added to the

NLL function

$$L(\boldsymbol{\beta})_{ridge} = L(\boldsymbol{\beta}) + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 \tag{5.58}$$

where λ is the *regularization* parameter.

To estimate the parameters $\boldsymbol{\beta}$ of the logistic regression model we have to minimize the regularized NLL by computing the derivatives $\frac{\partial L(\boldsymbol{\beta})_{ridge}}{\partial \boldsymbol{\beta}}$ and setting the result to zero:

$$\frac{\partial L(\boldsymbol{\beta})_{ridge}}{\partial \boldsymbol{\beta}} = \sum_{i=1}^{N} \boldsymbol{x}_i(y_i - p(\boldsymbol{x}_i, \boldsymbol{\beta})) - \lambda\boldsymbol{\beta} \tag{5.59}$$

$$= \sum_{i=1}^{N} \boldsymbol{x}_i(y_i - \frac{1}{1 + exp(-\boldsymbol{\beta}^\top \boldsymbol{x})}) - \lambda\boldsymbol{\beta} = 0. \tag{5.60}$$

Since equation (5.60) is non-linear in $\boldsymbol{\beta}$ no closed-form solution exists to minimize $L(\boldsymbol{\beta})_{ridge}$. Therefore the Newton-Raphson algorithm is applied, which requires the second derivative of equation (5.57):

$$\frac{\partial^2 L(\boldsymbol{\beta})_{ridge}}{\partial \boldsymbol{\beta}\boldsymbol{\beta}^\top} = -\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^\top p(\boldsymbol{x}_i, \boldsymbol{\beta})(1 - p(\boldsymbol{x}_i, \boldsymbol{\beta})) - \lambda\boldsymbol{\beta} \tag{5.61}$$

$$= -\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X} - \lambda \boldsymbol{I} \tag{5.62}$$

where $\boldsymbol{W}$ is a $N \times N$ diagonal matrix with the entries $p(\boldsymbol{x}_i, \boldsymbol{\beta})(1 - p(\boldsymbol{x}_i, \boldsymbol{\beta}))$ on the diagonal, and $\boldsymbol{I}$ is the identity matrix.

Starting with $\boldsymbol{\beta}^{old}$, a single step of the iterative Newton-Raphson algorithm is given by

$$\boldsymbol{\beta}^{new} = \boldsymbol{\beta}^{old} - \left(\frac{\partial^2 L(\boldsymbol{\beta})_{ridge}}{\partial \boldsymbol{\beta}\boldsymbol{\beta}^\top}\right)^{-1} \frac{\partial L(\boldsymbol{\beta})_{ridge}}{\partial \boldsymbol{\beta}}. \tag{5.63}$$

This algorithm is referred to as **iteratively re-weighted least squares** (IRLS) in this case, see e.g. [Nabney, 1999]:

$$\boldsymbol{\beta}^{new} = \left(\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{z} \tag{5.64}$$

with the adjusted response:

$$\boldsymbol{z} = \boldsymbol{X}\boldsymbol{\beta}^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p}) \tag{5.65}$$

where $\boldsymbol{p}$ is now the vector of fitted probabilities with the i'th element $P(\boldsymbol{\beta}^{old}, \boldsymbol{x}_i)$. As can be seen in equation (5.64) each of the iterations solves a weighted least squares problem

$$\boldsymbol{\beta}^{new} \leftarrow \underset{\beta}{\operatorname{argmin}}(\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta})^\top \boldsymbol{W}(\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta}). \tag{5.66}$$

In each iteration of the IRLS algorithm the inversion of $\left(\boldsymbol{X}^\top \boldsymbol{W}\boldsymbol{X} + \lambda \boldsymbol{I}\right)$ is required. Therefore, the computational cost of this algorithm depends on the dimension d of the input samples $\boldsymbol{x}_i$. However, the convergence rate also depends on the regularization parameter λ, i.e., for larger λ convergence is reached with fewer iterations of the IRLS algorithm. Usually the regularization parameter is optimized on a development set. A good scheme would be to start with a large λ and gradually anneal it to zero.

Since the NLL is convex, global convergence is guaranteed, but usually the iterative procedure stops when a certain convergence criterion is reached. Popular convergence criteria are based on the averaged change in the coefficients $\boldsymbol{\beta}$ or in the probability $p(\boldsymbol{x}, \boldsymbol{\beta})$. Alternatively, the algorithm converges when the change in the NLL becomes very small. After each update of $\boldsymbol{\beta}^{new}$ the ratio

$$\frac{|L\{\boldsymbol{\beta}\}_{new} - L\{\boldsymbol{\beta}\}_{old}|}{|L\{\boldsymbol{\beta}\}_{old}|} \tag{5.67}$$

is computed and the procedure stops if the ratio in (5.67) is less than a predefined value ϵ.

In [Minka, 2003] several algorithms of estimating the Logistic Regression coefficients are compared. Minka specifically investigated the computational cost of the estimation versus the performance (measured according to the NLL value).

5.5 Kernel Logistic Regression

The complexity of the function class $\mathcal{F}$ used for our discriminative classifiers is limited to linear functions. If we want to apply the logistic regression to non-linear classification problems we can extend LR by the use of kernel methods as stated in section 4.5.

The extension from the linear model to the non-linear one is realized by the non-linear mapping $\Phi : \mathbb{R}^d \rightarrow F$ into the feature space F and yields to a new

parameter vector $\boldsymbol{\beta}$ that lies in the span of all $\Phi(\boldsymbol{x}_i)$, $(i = 1, \ldots, N)$:

$$\boldsymbol{\beta} = \sum_{i=1}^{N} \alpha_i \Phi(\boldsymbol{x}_i). \tag{5.68}$$

The Representer theorem (see theorem 4.4) guarantees that it is possible to rewrite the weight vector $\boldsymbol{\beta}$ in terms of weights on the mapped data. If we extend equation (5.64) and (5.65) with equation (5.68) the whole optimization process is transferred into the feature space F:

$$\left(\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi} + \lambda \boldsymbol{I}\right) \boldsymbol{\Phi}^\top \boldsymbol{\alpha}^{new} = \boldsymbol{\Phi}^\top \boldsymbol{W} \left(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{\alpha}^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p})\right), \tag{5.69}$$

where $\boldsymbol{\Phi}$ denotes the matrix of all mappings $\Phi(\boldsymbol{x}_i)$. If we want to express the optimization in the form of kernels functions, we have to sort equation (5.69) in a way that all $\Phi(\boldsymbol{x})$ occur as dot-products only:

$$\boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{\alpha}^{new} + \lambda \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{\alpha}^{new} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{\alpha}^{old} \tag{5.70}$$

$$+ \boldsymbol{\Phi}\boldsymbol{\Phi}^\top (\boldsymbol{y} - \boldsymbol{p}). \tag{5.71}$$

Introducing now the kernel matrix $\boldsymbol{K}$ with entries $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j)$ and $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ we can write

$$\left(\boldsymbol{K}\boldsymbol{W}\boldsymbol{K} + \lambda \boldsymbol{K}\right) \boldsymbol{\alpha}^{new} = \boldsymbol{K}\boldsymbol{W} \left(\boldsymbol{K}\boldsymbol{\alpha}^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p})\right) \tag{5.72}$$

$$\left(\boldsymbol{K} + \lambda \boldsymbol{W}^{-1}\right) \boldsymbol{\alpha}^{new} = \boldsymbol{K}\boldsymbol{\alpha}^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p}). \tag{5.73}$$

Note, that by the use of the kernel trick there is no need to perform the mapping into the feature space explicitly. The optimal values of the parameter vector $\boldsymbol{\alpha}$ are again estimated via an iteratively re-weighted least squares procedure. Since IRLS requires the inversion of $\left(\boldsymbol{K} + \lambda \boldsymbol{W}^{-1}\right)$ at each iteration, this method becomes very expensive when the number of training examples is large. For that reason, [Keerthi et al., 2005] developed a dual formulation similar to the dual arising in SVMs that allows an efficient optimization of $\boldsymbol{\alpha}$. But in contrast to SVM, the elements of $\boldsymbol{\alpha}$ are usually non-zero, i.e., the optimization of the Kernel Logistic Regression does not lead to a sparse model by itself. Sparseness can be achieved by approximation techniques as we will discuss in the next section. Another promising but computational more expensive techniques called the **least absolute shrinkage and selection operator** (LASSO) was proposed by [Tibshirani, 1996]. Instead of optimizing regression coefficients subject to a bound on the L_2-norm like in ridge

regression, the LASSO imposes a L_1-penalty on the coefficients that leads to an automatic variable selection. Optimization techniques for the LASSO can be found in [Roth, 2004, Zou and Hastie, 2005].

5.6 Sparse Kernel Logistic Regression

The main drawback of the kernel logistic regression is that all training vectors are involved in the final solution which is not acceptable for large datasets like speech or speaker recognition tasks. A sparse solution of the kernel expansion, the **Sparse Kernel Logistic Regression** (SKLR), could be achieved if we involve only basis functions corresponding to a subset $\mathcal{S}$ of the training set $\mathcal{X}$. This is realized by approximating the Representer theorem by

$$\boldsymbol{\beta} = \sum_{i=1}^{M} \alpha_i \Phi(\boldsymbol{x}_i) \quad M \ll N. \tag{5.74}$$

with M training samples.

If we apply equation (5.74) instead of (5.68) in the IRLS algorithm of equation (5.64) we get the following sparse formulation of the kernel logistic regression:

$$\left(\boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi} + \lambda \boldsymbol{I}\right) \boldsymbol{\Phi}_M^\top \boldsymbol{\alpha}^{new} = \boldsymbol{\Phi}^\top \boldsymbol{W} \left(\boldsymbol{\Phi} \boldsymbol{\Phi}_M^\top \boldsymbol{\alpha}^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p})\right), \tag{5.75}$$

where $\boldsymbol{\Phi}_M$ denotes the matrix of all selected mappings $\Phi(\boldsymbol{x}_i)$. If we now expand both sides with the matrix $\boldsymbol{\Phi}_M$ on the right we can write

$$\boldsymbol{\Phi}_M \boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi} \boldsymbol{\Phi}_M^\top \boldsymbol{\alpha}^{new} + \lambda \boldsymbol{\Phi}_M \boldsymbol{\Phi}_M^\top \boldsymbol{\alpha}^{new} = \boldsymbol{\Phi}_M \boldsymbol{\Phi}^\top \boldsymbol{W} \boldsymbol{\Phi} \boldsymbol{\Phi}_M^\top \boldsymbol{\alpha}^{old} \tag{5.76}$$

$$+ \boldsymbol{\Phi}_M \boldsymbol{\Phi}^\top (\boldsymbol{y} - \boldsymbol{p}). \tag{5.77}$$

Finally, we can express $\boldsymbol{\alpha}^{new}$ in terms of kernel functions with the $N \times M$ matrix $\boldsymbol{K}_{NM} = k(\boldsymbol{x}_i, \boldsymbol{x}_j); \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{x}_j \in \mathcal{S}$ and the $M \times M$ regularization matrix $\boldsymbol{K}_{MM} = k(\boldsymbol{x}_i, \boldsymbol{x}_j); \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}$.

$$\boldsymbol{\alpha}^{new} = \left(\boldsymbol{K}_{NM}^T \boldsymbol{W} \boldsymbol{K}_{NM} + \lambda \boldsymbol{K}_{MM}\right)^{-1} \boldsymbol{K}_{NM}^T \boldsymbol{W} \tilde{\boldsymbol{z}} \tag{5.78}$$

where $\tilde{\boldsymbol{z}}$ is the new adjusted response of the sparse kernel logistic regression:

$$\tilde{\boldsymbol{z}} = \boldsymbol{K}_{NM} \boldsymbol{\alpha}^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p}) \tag{5.79}$$

Equation (5.78) provides a sparse approximation of kernel logistic regression, but one important question still remains: How to choose a new basis function for the optimization process?

5.6.1 Subset Selection

The selection approach provided in this thesis is a greedy method which imposes sparsity on the final solution. In subset selection we retain only a subset of the training data, and eliminate the rest from the model. Rather than search through all possible subsets, we can apply a search strategy to find a path through them. Greedy subset algorithms are **forward selection** and **backward selection**, for more details we refer to [Hocking, 1976, Draper and Smith, 1998, Miller, 2002]. Forward selection starts with an empty subsets, and then sequentially adds into the model the predictor that most improves the fit. However, the improvement in fit can be measured in several ways, e.g., decrease of the residual error.

In contrast, backward selection starts with the full model and sequentially removes predictors that are not relevant. This is computationally more expensive since in the first step the whole kernel matrix has to be computed.

In our approach we choose the NLL as selection criterion and include into the model the predictor that provides the minimum value of the NLL. This procedure of vector selection is summarized in algorithm 1. In each step $\boldsymbol{\alpha}$ is approximated by the fitted result from the current subset $\mathcal{S}$ which is estimated in the minimization process (see also [Zhu and Hastie, 2005]).

The selection routine continues until convergence is reached or until a maximum number of samples is selected.

As convergence criterion the relative change in the NLL is used (see equation (5.67)). After each update of $\boldsymbol{\alpha}^{new}$ the change in the NLL of the current and the previous subset is computed and the selection routine stops if the ratio in (5.67) is less than a predefined value ϵ.

5.7 Sparse Kernel Logistic Regression based on Feature Vector Selection

An alternative method of incorporating sparseness into kernel logistic regression is based on the **feature vector selection** (FVS) algorithm presented by [Baudat and Anouar, 2001]. The main idea of this algorithm is to construct

Algorithm 1 Sparse Kernel Logistic Regression

Let $\mathcal{S} = \emptyset$, $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $k = 0$, $L_k = \infty$
repeat
  for all $\boldsymbol{x}_i \in \mathcal{X}$ **do**
    minimize $L(\boldsymbol{x}_i)$
    if $L(\boldsymbol{x}_i) < L_k$ **then**
      $L_k = L(\boldsymbol{x}_i)$
      $\boldsymbol{x}_i^\star = \boldsymbol{x}_i$
    end if
  end for
  Set:
  $\mathcal{S} = \mathcal{S} \cup \{\boldsymbol{x}_i^\star\}$
  $\mathcal{X} = \mathcal{X} \setminus \{\boldsymbol{x}_i^\star\}$
  $L_k = L(\{\boldsymbol{x}_i^\star\})$
  $k = k + 1$
until $\frac{|L_k - L_{k-1}|}{|L_k|} < \epsilon$

a new basis in the subspace of F and to project the training data on this subspace. Therefore the normalized Euclidean distance between the training data in the feature space $\Phi(\boldsymbol{x}_i)$ and a set S of selected training vectors $\Phi_S(\boldsymbol{x}_i)$ is calculated by:

$$\delta_i = \frac{\|\Phi(\boldsymbol{x}_i) - \Phi_S(\boldsymbol{x}_i)\|^2}{\|\Phi(\boldsymbol{x}_i)\|^2}. \tag{5.80}$$

Rewriting equation (5.80) in matrix form, e.g., $\boldsymbol{\Phi}_S = (\Phi(\boldsymbol{x}_1), ..., \Phi(\boldsymbol{x}_M))$ we can express this distance in terms of inner products and finally in terms of kernel functions:

$$\delta_i = 1 - \frac{\boldsymbol{K}_{Mi}^\top \boldsymbol{K}_{MM}^{-1} \boldsymbol{K}_{Mi}}{k_{ii}} \tag{5.81}$$

where $\boldsymbol{K}_{MM}$ is the kernel matrix of selected training vectors and $\boldsymbol{K}_{Mi}$ represents a column vector of inner products of the training vector $\boldsymbol{x}_i$ and selected vectors $\{\boldsymbol{x}_j\}_{j \in S}$. To form a new basis we have to minimize the mean reconstruction error δ_i over all samples which results in the maximization of the fitness function:

$$J_S = \frac{1}{N} \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} \frac{K_{Mi}^T K_{MM}^{-1} K_{Mi}}{k_{ii}} \tag{5.82}$$

where K_{MM} is again the kernel matrix of the selected vectors with the entries $k(\boldsymbol{x}_i, \boldsymbol{x}_j)_{i,j \in S}$ and K_{Mi} is is a column vector of dot-product between $\boldsymbol{x}_i$ and the selected vector set S. Starting with $S = \varnothing$ we add to S the sample combined with vectors already in S that maximizes J_S in each iteration. The algorithm stops, if the matrix is no longer invertible or when the error is below a given value, or a predefined maximum number of selected vectors is reached.

5.8 Efficient Algorithms

We now focus on two efficient implementation aspects of the optimization process for kernel logistic regression. The first aspect in 5.8.1 deals with a probabilistic speed-up of subset selection for the parameter optimization. The computationally most expensive part of the optimization process is the matrix inversion. Therefore we investigate efficient methods of inversion concerning the speed of convergence in subsection 5.8.2.

5.8.1 Probabilistic Speed-up

Using the sparse kernel logistic regression methods presented in the last subsections we have to estimate the regularized NLL for all $N - M$ training samples in every iteration.
A possible speed-up is presented in [Smola and Schölkopf, 2000] and successfully applied to the computation of **Kernel Fisher Discriminant** (KFD) in [Mika et al., 2001].

For uniformly distributed objectives it can be sufficient to consider a learning set of size $\kappa = \log 0.05 / \log 0.95 = 59$ random samples. This means with a probability of 0.95 we obtain an estimate that is among the best 5% of all estimates. So in each iteration we choose just a small fraction of the training samples to estimate the NLL.

5.8.2 Matrix Inversion

The IRLS algorithm solves in each iteration a weighted least squares linear regression problem. The direct computation of the regression coefficients requires to invert the regularized matrix which is of the size $d \times d$ in logistic and of size $N \times N$ in kernel logistic regression. Applying standard techniques for the matrix inversion induce computational costs of the order $O(dN^2)$ for the standard logistic and $O(N^3)$ for the kernel logistic, which is extremely slow.

In [Minka, 2003] several alternative algorithms of computing the logistic regression coefficients are compared. To approximate the solution of the sub-problems one can apply a linear conjugate gradient method to accelerate the estimation in each iteration. The computational costs are then reduced to $O(N^2)$.

Instead of the conjugate gradient algorithm the so called **inversion lemma** or **Sherman-Morrison-Woodbury** formula [Hager, 1989, Golub and van Loan, 1996] can be used to update the inverse matrix. The inverse of the k-order matrix, where k is the number of basis functions in the model, is known from the previous iteration. Thus the inverse of the $k+1$ -order matrix of the next iteration can be stated as

$$\begin{pmatrix} A & U \\ V & -C^{-1} \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} I \\ 0 \end{pmatrix} \tag{5.83}$$

and transformed into an inversion of the $k-1$-order matrix. The inverse of a rank-k correction of some matrix can be computed by doing a rank-k correction to the inverse of the original matrix.

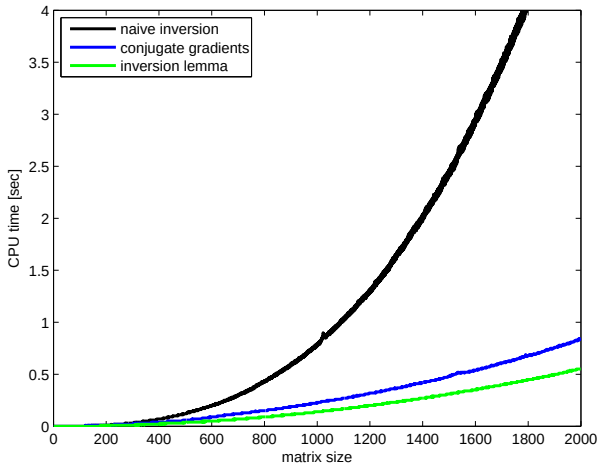

Figure 5.3: CPU time for solving the system of linear equations $\boldsymbol{Ax} = \boldsymbol{b}$ depending on data dimension.

We compare the three matrix inversion methods in an artificial experiment and present the CPU time for the computation of $\boldsymbol{Ax} = \boldsymbol{b}$ in figure 5.3. The CPU time is averaged over 100 repetitions of computing the problem by a naive matrix inversion, by the conjugate gradients methods and by the

inversion lemma. As a result of this experiment we choose the inversion lemma for the SKLR parameter estimation because it is more efficient than the other two methods.

5.9 Binary Classifiers for Multi-class Problems

In this section we address the problem of applying binary classifiers, like SVM and SKLR, for multi-class problems. Since for binary problems only one decision boundary must be inferred, we construct a set of binary classifiers where the data of the observed class is classified against the data of all others. This standard approach is called **one-versus-rest** and is one of the easiest combination strategies for binary classifiers. This one-versus-rest method obtains a membership probability for each class and assigns the sample to the class with the highest probability.

An alternative approach for multi-class problems was proposed in [Price et al., 1995] and is referred to as **one-versus-one**. If the binary classifiers provide class probability outputs, we can construct a binary classifier for all possible pairs in the dataset and learn $C(C-1)/2$ binary classifiers. The binary class probability estimates are then coupled into a joint probability estimate for all C classes.

Let $P_{ij} = P(y = i | y \in i, j, f(\boldsymbol{x})$ be the probabilities for all classification pairs (i, j) with $i \neq j$ and that each example belongs only to one class:

$$P(\bigcup_{j=1}^{C} y_j | f(\boldsymbol{x})) = 1. \tag{5.84}$$

It follows for any given i:

$$P(\bigcup_{j=1}^{C} y_j | f(\boldsymbol{x})) = P(\bigcup_{j=1, j\neq i}^{C} y_{ij} | f(\boldsymbol{x})) = 1 \tag{5.85}$$

Using the closed form expression for the probability of the union of N events

E_i

$$P(\bigcup_{j=1}^{N} E_i) = \sum_{i=1}^{N} P(E_i) + \cdots + (-1)^{k-1} \sum_{i_1<\cdots<i_k}^{N} P(E_{i_1} \wedge \cdots \wedge E_{i_k}) + \ldots + (-1)^{N-1} P(E_1 \wedge \cdots \wedge E_N) \tag{5.86}$$

it follows from (5.85):

$$\sum_{j=1, j\neq i}^{C} P(y_{ij}|f(\boldsymbol{x})) - (C-2)P(y_i|f(\boldsymbol{x})) = 1. \tag{5.87}$$

With

$$P_{ij} = P(y_i|y_{ij} \wedge f(\boldsymbol{x})) = \frac{P(y_i \wedge y_{ij} \wedge f(\boldsymbol{x}))}{P(w_{ij} \wedge f(\boldsymbol{x}))} = \frac{P(y_i|f(\boldsymbol{x}))}{P(y_{ij}|f(\boldsymbol{x}))} \tag{5.88}$$

we finally obtain for the C posterior probabilities $P(y_i|f(\boldsymbol{x}))$ given the class probabilities P_{ij} of the $C(C-1)/2$ classification pairs (i, j):

$$P(y_i|f(\boldsymbol{x})) = \frac{1}{\sum_{j=1, j\neq i}^{C} 1/P_{ij} - (C-2)}. \tag{5.89}$$

Hastie and Tibshirani presented an alternative approach of pairwise coupling that is based on the minimization of the **Kullback Leibler** (KL) distance [Hastie and Tibshirani, 1998], which is summarized in algorithm 2.

Algorithm 2 Pairwise coupling for multi-class problems

1: Start with a guess for the $\hat{P}_i$ and corresponding ν_{ij}
2: **repeat**
3: $\quad \hat{P}_i \leftarrow \hat{P}_i \frac{\sum_{i\neq j} n_{ij}\nu_{ij}}{\sum_{i\neq j} n_{ij}\hat{\nu}_{ij}}$
4: $\quad$ renormalize $\hat{P}_i$ and recompute $\hat{\nu}_{ij}$
5: $\quad \hat{P} \leftarrow \hat{P} / \sum_{i=1}^{C} \hat{P}_i$
6: **until** convergence

5.10 Classification Experiments

For a comparison of the discussed classifiers we report in this section classification experiments on three small speech datasets. The **Deterding** [Robinson, 1989], the **Peterson & Barney** [Peterson and Barney, 1952] and the **isolet** [Fanty and Cole, 1991] task. These three datasets are briefly described in the following.

The first task, the Deterding dataset, consists of 11 spoken vowels that are characterized by 10 LPC coefficients. The training set contains 528 samples composed of six samples of each vowel spoken by eight speakers. For testing seven speakers produced six samples of each vowel. This results in 462 samples for testing.

The features of the second task, the Peterson & Barney dataset, are represented by the four formants of 10 vowels spoken by 76 speakers. The training and the test set consist of 760 samples each.

The third task, the isolet dataset, contains all 26 spoken letters of the alphabet and is characterized by 617 spectral coefficients. The training set contains two realizations of each letter spoken by 120 speakers. The test set is constructed of two realizations spoken by 30 speakers. This results in 6240 samples for training and 1560 samples for testing. Additionally to the whole isolet task we performed experiments on a subset of this task, which is called **E-set**. The E-set consists of 240 examples of each letter {B C D E G P T V Z} for training and 60 realizations per letter for testing. Interestingly the letters of this subset are harder to classify [Smith and Gales, 2002].

To start the experiments we estimated the SVM parameters by using the **SVMTorch** library [Collobert and Bengio, 2001]. Since the SVM has no probabilistic output the a-posteriori probabilities are estimated by Platt's algorithm (5.43).

The algorithms of the KLR, the SKLR and the feature vector selection were implemented in **Matlab**, which is a numerical computing environment and programming language [The MathWorks, 1997].

Since the discussed classifiers are binary classifiers (section 5.9) we applied the one-versus-one classification scheme. Therefore, the data was divided into $C(C-1)/2$ classification pairs, where C represents the number of classes in the dataset. In all experiments we used the RBF kernel (4.33) since this kernel outperforms the polynomial kernel (4.32) in several previous publications [Clarkson and Moreno, 1999, Ganapathiraju, 2001, Krüger et al., 2005, Schafföner et al., 2006]. The kernel parameters were optimized by a cross-validation procedure on the training set. The KLR regularization parameter

λ and the SVM trade-off parameter C were optimized in the same way.

For a better comparison we also report classification results achieved by Gaussian mixture models. The GMM baseline results yield an error rate of 37.9% for the *Deterding* dataset and 18.3% for *Peterson & Barney*. These GMM error rates were taken from [Clarkson and Moreno, 1999].

Table 5.1: Classification experiments on three small speech datasets. Test errors rates (%) of several KLR versions are compared to GMM and SVM models. Error rates for *Deterding* and *Peterson & Barney* are taken from [Clarkson and Moreno, 1999].

Dataset	GMM	SVM	KLR	SKLR	KLR-FVS
vowel Det.	37.9	28.36	30.02	29.44	32.19
vowel P.B.	18.3	11.45	11.17	11.17	13.77
isolet E-set	7.20	4.45	4.63	4.45	4.63
isolet full	5.06	2.89	2.89	3.02	3.14

Table 5.1 shows the classification error rates of the different classifiers on the four datasets (see also [Katz et al., 2005]). It can be clearly seen that all discriminative classifiers outperform the Gaussian mixture models. The classification results of the standard KLR and the SKLR method are comparable to the SVM on all datasets. It becomes apparent that the KLR based on feature vector selection method is outperformed by all other kernel classifiers.

Additionally, we present the sparsity of the discriminative classifiers by comparing the number of kernel functions used in the final solution of each classifier. The results of this comparison are presented in table 5.2. For a better representation the number of kernel functions was averaged over the $C(C-1)/2$ classification pairs. At first it is obvious that for the standard KLR as a non-sparse method, the number of kernel functions is equal to the number of training vectors.

Table 5.2: Sparseness of different KLR variants compared to SVM. The number of kernel functions is averaged by the number of classification pairs.

Dataset	SVM	KLR	SKLR	KLR-FVS
vowel Det.	35.9	96.0	31.1	29.0
vowel P.B.	12.6	152.0	10.1	19.4
isolet E-set	140.0	480.0	76.9	120.0
isolet full	100.9	480.0	59.5	120.0

As can be observed in table 5.2 the SKLR provides classifiers that require a fewer number of kernel functions than the Support Vector Machine in all cases. Especially on the isolet tasks the SKLR is superior. The Support Vector Machines utilizes about 140 kernel functions for the E-set and 100.9 for the full set compared to 76.9 and 59.5 kernel functions for the Sparse Kernel Logistic Regression.

For further investigations of sparsity we performed additional experiments on several partitions of the isolet dataset. We divided the training data into subsets of 50 to 90 percent of the whole isolet task. Using the same parameters as for the full dataset the results in table 5.3 show that the SKLR classifiers achieved a higher sparsity than SVMs on all subsets.

Table 5.3: Sparsity of SKLR compared to SVM on different training sizes of the isolet dataset. The number of kernel functions is averaged by the number of classification pairs.

Dataset	SVM	SKLR
isolet-50	75.7	55.0
isolet-60	82.8	58.3
isolet-70	86.5	56.3
isolet-80	92.1	60.0
isolet-90	96.4	58.8
isolet-100	100.9	59.5

Whereas the averaged number of kernel functions for the SVM increases from about 75.7 to 100.9, the SKLR solution varies between 55 and 60 kernel functions. The results in table 5.3 clarify the advantage of SKLR compared to SVM. Whereas the SVM requires more than 20% of the training data on the full isolet task, the solution of the SKLR only depends on 12%. This effect is also observed in other kernel classifiers that are based on forward selection, e.g. [Andelic et al., 2006, Schafföner et al., 2006].

Summarizing we can say that the SKLR requires dramatically fewer kernel functions for the same amount of training data by a comparable classification performance. Furthermore, in contrast to SVMs the SKLR provides a probabilistic output that gives us the possibility to quantify the uncertainty about the predicted class memberships.

5.11 Summary

In this chapter we described two different classifiers which are rooted in the theories presented in chapter 4. First, we gave a brief overview of Support Vector Machines (SVMs) and introduced in detail the Kernel Logistic Regression (KLR) and its sparse version SKLR. Secondly, we compared the classification performance of these discriminative classifiers with Gaussian Mixtures Models (GMMs) on four speech datasets. Thirdly, we discussed the issue of sparsity when kernel classifiers are applied to speech classification problems. Whereas the number of kernel functions required by the SVM is strongly influenced by the size and complexity of the training set, the SKLR provides dramatically sparse solutions. This sparsity is achieved by approximating the optimal solution in an iterative way by construction until convergence is reached.

As we will see in the following chapter, the sparsity of the SKLR is an important advantage with regard to several applications in speaker recognition.

6 Speaker Recognition on Limited Training Data

This chapter presents a novel speaker recognition system for speaker identification and verification on limited training data. The classifiers which we presented in chapter 5 were directly applied to feature vectors of corresponding speech data in this system.

The recognition system is compared to a traditional GMM system in section 6.6 for speaker identification and in section 6.7 for speaker verification. All experiments were performed on the POLYCOST dataset [Melin and Lindberg, 1996].

6.1 The POLYCOST corpus

The following section describes the POLYCOST corpus that is used to evaluate the proposed methods on small speaker identification and verification tasks. This dataset [Melin and Lindberg, 1996] is a telephone based speaker recognition corpus. It was created during 1996-1999 within the framework of the European project COST250 as a reference database for baseline experiments. The dataset contains 110 speakers (61 females and 49 males) from different European countries. Their utterances are prompted in English as well as in the speaker's native language. The whole dataset is divided into four baseline experiments (BE1-BE4), from which we used two, the text-independent BE4 set for speaker identification and the text-dependent set BE1 for speaker verification experiments. Each of these two sets provides four training sentences and up to five test sentences for each speaker. In the standard setup only the first two sentences were used for enrollment, the other two serve as development data, see [Melin and Lindberg, 1996].

6.2 The Front-End

To extract feature vectors containing relevant speaker information, the front-end of the speaker identification system applied speech parameterization methods, which were previously described in chapter 3.2.

First, feature vectors were extracted from speech using a 20 ms Hamming window and a window shift of 10 ms. Furthermore non-speech frames were removed using energy-based speech activity detection. Since it is useful to reject lower and higher frequency regions for processing telephone speech, the frequency range was limited from 300 Hz to 3400 Hz.

Afterwards the number of Mel cepstrum coefficients was optimized on the development set. For speaker identification we computed 13 component MFCC feature vectors and added the first and second order time differences. Then the first and second order time differences of the frame energy were computed and appended to each feature vector, which results in a 41 dimensional feature vector. Finally, the MFCC feature vectors were normalized to zero mean and unit standard deviation on the remaining speech data.

6.3 The GMM System

The baseline GMM system for speaker identification on limited training data was designed for the POLYCOST dataset. For each of the 110 speakers in the set an individual GMM was trained using the Expectation-Maximization algorithm. These models were composed of 16 mixture components with diagonal covariance matrices.

6.4 The Frame-based Kernel Classifier Systems

In the frame-based approach the extracted feature vectors from the speech data were directly used to train the SVM and (S)KLR parameters [Katz et al., 2006c,a]. More specifically, the feature vectors of the target speaker were trained against feature vectors of a set of background speakers. These models of the target speaker were then estimated against all speakers of the background set using the **one-versus-one** classification scheme as discussed in section 5.9. The developed algorithms for the (S)KLR parameter estima-

tion were implemented in the C++ programming language [Stroustrup, 2000]. SVM models for the frame-based system were trained with SVMTorch [Collobert and Bengio, 2001].

6.5 Design and Implementation

The GMM speaker recognition system used in this thesis is based on the **ALIZE** toolkit, a well known open source software package developed by the Laboratoire d'Informatique d'Avignon, France [Bonastre et al., 2005]. This toolkit provides statistical functionalities especially for the estimation of GMM parameters and likelihood computation. The general architecture of the toolkit is based on a split of functionalities between several software servers. The main servers are the mixture server which deals with Gaussian mixture models and the feature server which manages the extracted feature vectors.

For the speaker recognition evaluation with discriminative classifiers we integrated these classification machines into the existing framework. To increase the computational power we used a **Beowulf** computer cluster [Sterling et al., 1999]. Beowulf is a design for high-performance parallel computing clusters by combining cheap personal computer hardware. The software was parallelized by splitting independent tasks, like the verification of different speakers into separate cooperating processes located on different computers of the cluster. A commonly used library called **Message Passing Interface** (MPI) [Snir et al., 1998] was used for this purpose.

6.6 Speaker Identification

Our novel speaker identification system was especially designed for the POLYCOST dataset [Melin and Lindberg, 1996]. As described in section 6.1 the POLYCOST dataset contains a text-independent speaker identification task where all speakers are registered as client speakers. Since it is a closed-set speaker identification task no explicit impostor trials were performed.

6.6.1 Experiments

As in the baseline GMM system the (S)KLR and the SVM models were applied at the frame-level. For the identification experiments we used two sentences of

each speaker for training and two sentences as development set. Altogether a total amount of only 10 to 20 seconds of free speech is available for parameter estimation of the speaker models. The evaluation set contains up to five sentences per speaker and each utterance has a duration of about five seconds of free speech. This results in 664 true identity tests. Numerous publications present recognition performance on the BE4 speaker identification dataset, e.g., the results of [Magrin-Chagnolleau and Durou, 1999, 2000], [Chakroborty et al., 2007] are given in table 6.1.

Table 6.1: Summary of speaker identification experiments on the BE4 set of the POLYCOST database.

Publication	IER (%)
Chakroborty et al. [2007]	18.43
Magrin-Chagnolleau and Durou [1999, 2000]	9.11

For the SVM and the (S)KLR classifiers we used the RBF kernel function (4.33) and validated the kernel and regularization parameters of the different classifiers on the development set. Due to the fact that all speakers are known to the system, the error rate is simply computed as **Identification Error Rate** (IER):

$$\text{IER} = \frac{\text{number of incorrect identifications}}{\text{total number of identification tests}} \tag{6.1}$$

The log-likelihood was computed over the whole test utterance and the sentence was assigned to the speaker with the highest probability over the whole speech sequence as formulated in equation (2.2). The IER is presented in the left column of table 6.2. Additionally, we generated an N-best list of each test utterance and give the results of finding the best speaker within the 5-best alternatives. These error rates are given in the right column of the table.

Table 6.2: Speaker identification experiments on the BE4 set of the POLYCOST database.

Classifier	IER (%)	5-best (%)
GMM	10.84	6.48
SVM	8.89	4.67
KLR	8.58	4.97
SKLR	8.58	5.87

As can be seen in table 6.2, both classifiers, the SVM and the KLR clearly outperform the GMM baseline system. The KLR as well as the SKLR classifier decreases the IER of the baseline system by more than 20% relatively. Furthermore, evaluating the classification on the 5-best alternatives, the SVM reaches the lowest error rate followed by the non-sparse KLR.

Additionally, we compared the sparseness of the different discriminative classifiers. Therefore we divided the training set into subsets of 50 to 90% of the full BE4 task. Apart from this we used the same SVM and (S)KLR parameters as for the full dataset.

Table 6.3: Sparseness (%) of SKLR compared to SVM on different training sizes using the BE4 dataset.

Dataset	SVM	SKLR	KLR
BE4-50	66.8	82.8	0.0
BE4-60	61.3	81.4	0.0
BE4-70	56.2	80.3	0.0
BE4-80	51.3	79.3	0.0
BE4-90	46.6	78.8	0.0
BE4-100	41.7	78.1	0.0

As result of this experiment we report the sparseness of each subset in table 6.3. The sparseness is defined as the portion of data not used in the final solution. The lower the sparseness the more kernel functions are part of the classifier.

As can be seen in table 6.3 the models of the SKLR approach are sparser than the SVM models on all subsets. On the full set the SVM solution requires about 58.3% of all training data, whereas the SKLR requires only 21.9%. Naturally the non-sparse KLR needs all training vectors in the experiments and its sparsity level is 0%. The number of support vectors increases nearly linear with the amount of training data which results in a decreasing sparseness. Conversely the number of relevant feature vectors for the SKLR increases only slightly for larger datasets. Consequently this is an obvious computational advantage over the SVM as a much smaller portion of kernel products have to be calculated in the test.

With the aim to compare the decision quality of the explored identification systems, we normalized the log-likelihood scores in the N-best list by the T-

Norm (see section 2.10.1). In this case the models of the T-Norm speakers for the (S)KLR and the SVM systems were trained in the same way (one-versus-one) as the models for speakers in the closed set. Figure 6.1 displays the histograms of the true speakers (right) and the average of the 10-best alternatives in the N-best list (left). It is important to note that for the GMM and the SKLR systems the overlap of the distributions is larger than for the other ones. This overlap leads to a higher probability of errors as discussed in section 2.9. In contrast, the smallest overlap could be observed for the SVM models.

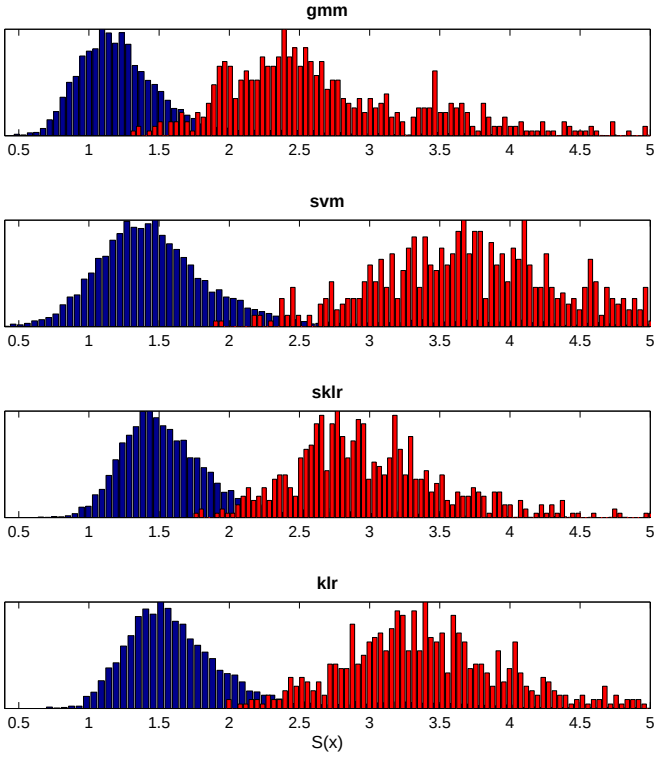

Figure 6.1: Scores of the 10-best alternatives (left) compared to the scores of the true speakers (right) for the different classification methods.

Based on the true and the alternative N-best speakers we can now introduce a threshold for accepting or rejecting a speaker of the identification set as described in section 2.9. The resulting **Equal Error Rate** (EER) gives the

lowest error rate by an equal rate of false-accepts and false-reject errors. The results are given in table 6.4.

Table 6.4: Equal Error Rates of the speaker identification experiments using a decision threshold.

Classifier	EER (%)
GMM	6.69
SVM	5.15
KLR	5.39
SKLR	6.15

With regard to the score distributions of figure 6.1 it becomes clear that the SVM outperforms all other systems. Due to the small overlap of the two distributions the decision threshold leads to a lower EER for the SVM than for the GMM and the Kernel Logistic Regression models. Interestingly, the non sparse KLR achieved a significantly lower error rate than the SKLR.

6.7 Speaker Verification

It was aimed to compare the novel speaker recognition system presented in section 6.4 with a standard GMM verification system. Therefore, the extracted feature vectors from the speech data are directly used to train the parameters of the discriminative kernel classifiers. That means, that the feature vectors of the target speaker are marked as members of the positive class and the feature vectors of a background speaker set are assigned to the negative class (for details see [Katz et al., 2006a] and section 6.4). The following section presents our speaker verification experiments on limited training data.

6.7.1 Experiments

In the verification experiments the sentence "Joe took father's green shoe bench out" is given as a fixed password sentence shared by all clients. Following the corpus guidelines version 2.0 [Melin and Lindberg, 1996] the corpus provides two sentences for enrollment, one from the first recorded session and one from the second session. The evaluation set contains up to five sentences per speaker and each utterance has a duration of about five seconds of free speech. This results in 664 true client and 824 impostor (non client) trials. The genders were evaluated separately and no cross gender trials were performed.

Results from experiments on the POLYCOST verification set BE1 have been reported in several publications, e.g. [Nordström et al., 1998, Katz et al., 2006a]. Our baseline GMM environment consists of gender-dependent background models that were trained by 22 non-client speakers from the POLYCOST database. Speaker models were adapted from the background GMM by MAP adaptation as described in section 6.3.

With the aim to compare the influence of the amount of speech data for enrollment we trained the baseline GMM system as well as the novel discriminative classifier system on one (*1sent*) or two sentences (*2sent*) of each speaker. The results of the three classifiers are given in figure 6.2 as **detection error tradeoff** (DET) plots. The DET plots show the tradeoff between **false-rejects** (FR) and **false-accepts** (FA) as a decision threshold [Martin et al., 1997]. The actual decision costs are denoted as circles.

Additionally we report the equal error rates and the DCF values as performance measure in table 6.5. The parameters of the cost function used in the experiments are $C_{FR} = 10$, $C_{FA} = 1$ and $P_{Target} = 0.01$. The right column of

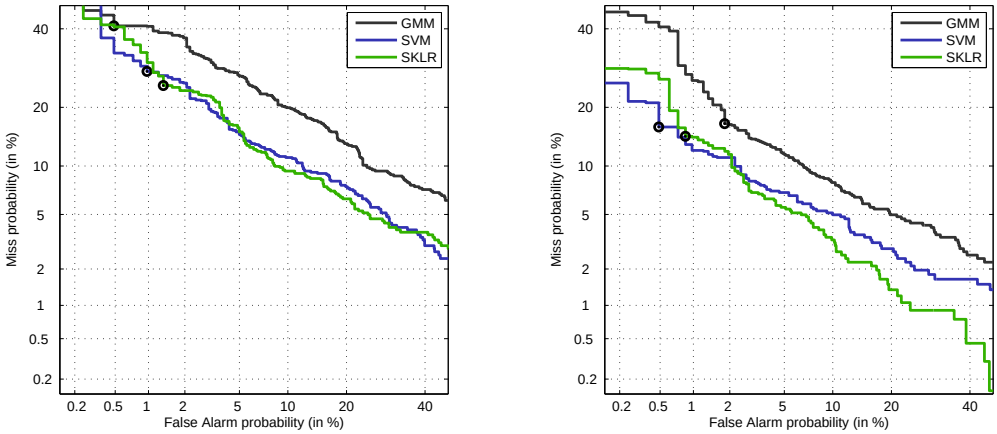

Figure 6.2: Influence of amount of speech data for enrollment using GMM, SVM and SKLR classifiers. The DET curves show systems trained on one sentence (left) and two sentences (right).

the table gives the averaged evaluation time (CPU time) of an identification claim.

Figure 6.2 shows that the discriminative classifiers clearly outperform the GMM baseline in both experiments. Regarding the DET curves of the *1sent* experiment we can observe that SVM and SKLR achieved comparable DCF values. However the SKLR system leads to an EER that is about 14% less than the SVM result. It is important to notice that the behaviour concerning optimal operating points is different for the *2sent* experiment. This is illustrated in the right DET curves of figure 6.2. When the emphasis is placed on security the SVM provides a better operating point than SKLR. This is, for example, the case in security applications where the false accept (or false alarm) probability is more important and should be minimized. In contrast, when emphasis is placed on speaker detection, where the miss (or false reject) probability is more important, the SKLR results in lower miss probabilities than the SVM. Concerning the equal error rate the SKLR again outperforms the SVM system. It leads to an EER that is about 10% less than for the SVM (see table 6.5).

Table 6.5 also shows the computational costs of the verification process. Whereas the costs of computing the GMM probabilities depend on the number of mixture components, the costs for the discriminative approaches result on the number of kernel functions. Independently of the number of training sentences the GMM system requires about 3.75 ms per identification claim.

For the *2sent* experiment the GMM system is about 40 times faster than SVM system and about 20 times faster than the SKLR system. This computational drawback of the discriminative classifiers is due to the calculation of numerous kernel functions for each feature vector. But the sparse solutions of the SKLR is again an important advantage compared to the SVM.

Table 6.5: Comparison of the EER, DCF value and evaluation time (per identification claim) for three systems trained on one ore two sentences of the POLYCOST-BE1 speaker verification task.

Training Sentences	Classifier	EER (%)	DCF	Evaluation Time (ms)
1 sent	GMM	16.01	0.0456	3.75
	SVM	11.03	0.0380	112.77
	SKLR	9.49	0.0381	69.39
2 sent	GMM	8.61	0.0348	3.75
	SVM	6.05	0.0210	148.86
	SKLR	5.45	0.0215	75.22

It can be summarized that in both experiments the SVM yields a better tradeoff between a low false accept (or false alarm) probability and a high miss (false reject) probability than the two other classifiers. Whereas the SKLR system achieves results with respect to low miss (false reject) and high false accept (or false alarm) probabilities.

6.8 Summary

In this chapter we introduced a novel speaker recognition system. The key idea of this system is to directly apply discriminative kernel classifiers on parameterized speech frames. Experimental evaluations on two speaker identification and verification tasks of the POLYCOST dataset confirmed the high competitiveness of the discriminative classifier approach.

In general it can be concluded that the SVM as well as the SKLR system outperformed the GMM classifiers concerning recognition accuracy in all experiments when directly applied on parameterized speech frames. Furthermore the Sparse Kernel Logistic Regression leads to very sparse solutions compared to the Support Vector Machine.

The drawback of discriminative kernel classifiers is the computational cost in parameter estimation as well as in evaluation. Especially for larger training

sets this classification methods would be computationally intractable. For this reason we investigate a different approach of using kernel classifiers on extended training and test data in the following chapter.

7 Speaker Recognition on Extended Training Data

In the following chapter we present the speaker recognition system which was especially designed for the NIST speaker recognition evaluation campaign[1]. It focuses on speaker detection using a large amount of speech data.

In section 7.1 we first give a short overview of the NIST speaker recognition evaluation campaign and outline datasets which we used for enrollment, development and evaluation of the system. Furthermore we describe the front end in section 7.2 and present the baseline GMM system together with baseline experiments using different GMM model sizes as well as different feature parameterization methods in section 7.3.

The integration of discriminative kernel classifiers and related methods is given in section 7.4 ff. Moreover, we discuss the compensation of intersession variabilities and apply this method to our recognition framework (see section 7.6).

A comparative evaluation of SVM and SKLR using different parameterization techniques is performed. Finally, we perform experiments on different fusion techniques of the three subsystems in section 7.7 and conclude the chapter with an experiment on score calibration of unmoderated SVM outputs in section 7.7.2.

7.1 NIST Speaker Recognition Evaluation

Since 1996 the **National Institute of Standards and Technology** (NIST) organizes annual **speaker recognition evaluation** (SRE) campaigns where participating parties can present and openly discuss their speaker recognition systems [Przybocki and Martin, 2004]. The NIST evaluations primarily focus

[1] http://www.nist.gov/speech/tests/sre/

on speaker detection in the context of conversational telephone speech. The GMM system which we will introduce in section 7.3, is a further development of the system we presented at the NIST 2006 SRE campaign [Katz et al., 2006b].

Each SRE campaign provides an explicit evaluation plan with rules and definitions of training and test combinations. The NIST SRE-04, SRE-05 and SRE-06 corpora were drawn from the Mixer corpus [Martin et al., 2004], a multilingual and multi-modal (cross-channel) data collection project undertaken by the **Linguistic Data Consortium** (LDC)[2]. The main language of the collection is English. However a special effort has been made to additionally recruit bilingual subjects who are also able to speak Arabic, Mandarin, Russian or Spanish.

The NIST 2006 corpus serves as state-of-the-art speaker recognition corpus to evaluate the performance of the speaker verification system developed in this thesis.

Table 7.1: Used Datasets in the NIST 2006 evaluation. The number of sentences is given for males and females

Dataset	UBM	T-Norm	Development	Evaluation
SWITCHBOARD Cell.	50/50	25/25	-	-
NIST SRE 2000	100/100	-	-	-
NIST SRE 2004	75/75	75/100	-	-
NIST SRE 2005	25/25	-	274/372	-
NIST SRE 2006	-	-	-	354/462

Therefore we mixed data from four different corpora for background model training, score normalization and parameter optimization on development sets. The background models were trained on parts of the SWITCHBOARD Cellular dataset, the NIST 2000 evaluation set and a part of the NIST 2004 evaluation set. Furthermore we also used data from the SWITCHBOARD and the NIST 2004 sets for T-Norm models. Model parameters were optimized on the NIST 2005 corpus.

The data of the NIST SRE 2006 task is divided into several training and test partitions, for further details see [Przybocki and Martin, 2006]. The main task of these partitions is the so called **core test** with same conditions for training and test; these are a two-channel conversation of approximately five minutes total duration with the target speaker channel designated. The data

[2] http://www.ldc.upenn.edu/

of this condition contains different languages as well as different transmission channels and telephone types. The training set consists of 351 male and 462 female speaker sentences for model training. Each target model is trained and tested by one conversation (1conv) using a four wire telephone line (4w). All together there are 51448 (29317 female and 22131 male) trials in this 1con4w-1con4w evaluation test condition.

Restricting to **English-only** conditions the number of trials is reduced to 24013 (14293 female and 9720 male). We also present results on this subset to compare our system to other recently published results. In the framework of this evaluation it is possible to use alternative conditions like three or eight conversations for training and one conversation for testing, 3con4w-1con4w and 8con4w-1con4w respectively.

An excerpt of different DET plot definitions used in the evaluation are summarized in table 7.2. In this thesis we focus on the core test, defined as DET1 in the first row of the table, and on the so called English-only condition (DET3), which shows the performance of the system when all data are limited to English. Additionally, we report results by gender (DET2, DET4) for the core test and the English-only condition.

Table 7.2: Description of different test conditions used in the NIST 2006 Speaker Recognition Evaluation.

Data	Description
DET1	Shows the overall performance (core test)
DET2	Shows the overall performance by gender
DET3	Shows the performance when all data (test segments and model training) are limited to English (English-only)
DET4	Shows performance when all data (test segments and model training) are limited to English by gender

7.2 The Front-End

The front-end of the speaker verification system for large datasets is based on speech parameterization methods described in chapter 3.2. In all experiments we used three different feature extraction methods, namely Mel Frequency Cepstral Coefficients (MFCC), Perceptual Linear Prediction (PLP) and Linear Predictive Cepstral Coefficients (LPCC).

Feature vectors were extracted from speech by using a 20 ms Hamming window and a window shift of 10 ms. Previous to the extraction of features the audio was band filtered between 300 Hz and 3400 Hz. Non-speech frames were removed using an energy-based speech activity detection as described in chapter 3.3. After feature extraction all parameter vectors were normalized to fit a zero mean and a unit variance distribution. Corresponding means and variances were estimated on each speech file independently.

The first system consisted of 19 LPCC and their first order time differences, for a total of 39 features per vector.

For the MFCC front-end we used a 24-channel filter bank and 13 MFCC as well as 13Δ and 13$\Delta\Delta$ coefficients. The first and second order time differences of the energy were appended.

The PLP system used 50 components including 19 PLP coefficients, 19 first order and 11 second order time differences. Finally, the Δ-energy was appended to complete the PLP feature vector.

In general, the feature extraction was carried out using the SPRO toolkit [Gravier, 2004] and some additional implementations to perform PLP. The optimal number of features for each system was found on the development set using the GMM system presented in the next section.

7.3 The GMM System

The GMM system of our proposed system is based on the Universal Background Model (UBM) approach (see section 2.5). For all systems gender dependent UBMs were trained on background data which was taken from several datasets (see table 7.1). The background data contain different speaker characteristics as well as different microphone and transmission channels to cover a large spectrum of different speech sources. Each gender dependent UBM consists of Gaussian mixture components with diagonal covariances and is trained via the Expectation Maximization (EM) algorithm (section 2.6.1).

In general it is possible to model the speaker specific models from scratch in the same way as the UBM. However, due to limited training data this may result in underestimated parameters of the speaker GMM. It is then possible that unseen feature vectors are classified according to randomly generated mixture distributions. To overcome this problem, the speaker models were adapted from the well trained background model. Therefore the speaker specific models were derived from the UBMs using a one step Maximum A-Posteriori adaptation [Reynolds et al., 2000]. Only the means of the mixtures

were adapted with a predefined relevance factor τ (see section 2.6.2).

During the recognition or detection phase, probabilities of UBM and speaker models have to be computed. According to the GMM definition, one has to sum up all weighted mixture probabilities of both models. Using M mixture components this results in the computation of $2 \cdot M$ Gaussian mixture probabilities. It has been found in [Reynolds et al., 2000], that it is sufficient to approximate the final probabilities by the computation of the N highest mixture probabilities of the speaker model with respect to the background GMM. This reduces the computational cost to $M + N$ probability calculations. An important step in speaker recognition via telephone is the compensation of channel mismatches. This compensation was realized by feature mapping (section 2.7) which was performed on the acoustic feature vectors during enrollment and evaluation.

Detection probabilities were computed through equation (section 2.1) over the whole sequence and normalized by the TNorm, see 2.10.1. The GMM system presented in this section serves as baseline detection system for the following system extentions.

7.3.1 Baseline Speaker Detection Experiments

In a first experiment we investigated the influence of model size on the recognition performance. The DET curves in figure 7.1 show the results using the LPCC, the MFCC and the PLP subsystems. Models with 128, 256, 512, 768 and 1024 mixture components were compared for each subsystem. For all of these subsystems we observe that the EER as well as the DCF value decreases when more than 256 mixture components were used. It becomes apparent in the figure that the performance of systems using 512 and more components is absolutely comparable. Since more components did not improve the performance anymore, we chose GMMs with 512 mixture components for all following experiments.

The DET curves of the three baseline GMM subsystems for the core test and the English-only condition are shown in figure 7.2. The actual DCF values are denoted by circles on the DET curve. Additionally, table 7.3 summarizes the gender dependent Equal Error Rates as well as the DCF values of the three baseline speaker detection systems. The comparison of all presented subsystems points out that in general the detection performance for the female speaker is inferior to the male speaker system. The best detection performance for both gender on the core test was achieved by the PLP subsystem with a

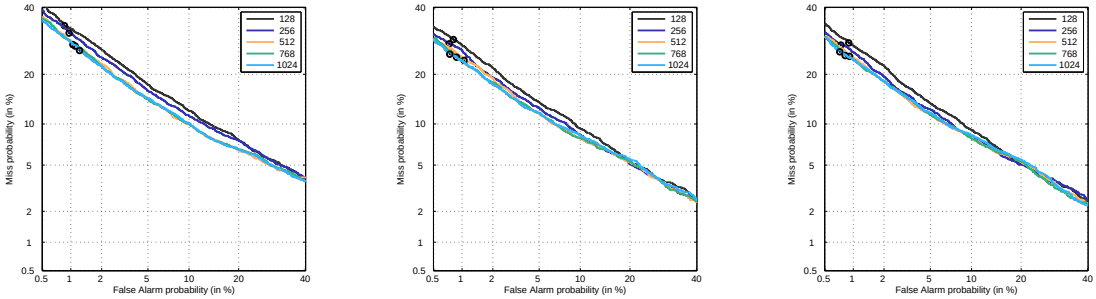

Figure 7.1: Influence of model size on different GMM systems. The DET curves show the LPCC (left) the MFCC (center) and the PLP (right) subsystems with T-Norm on the NIST 2005 development corpus.

DCF value of 0.0345 compared to 0.0365 of the MFCC subsystem and 0.0426 of the LPCC subsystem. Besides, the PLP subsystem achieved the lowest Equal Error Rates compared to the the MFCC and LPCC subsystems. As can be seen in figure 7.2, the performance of the subsystems is consistent over the whole error ranges.

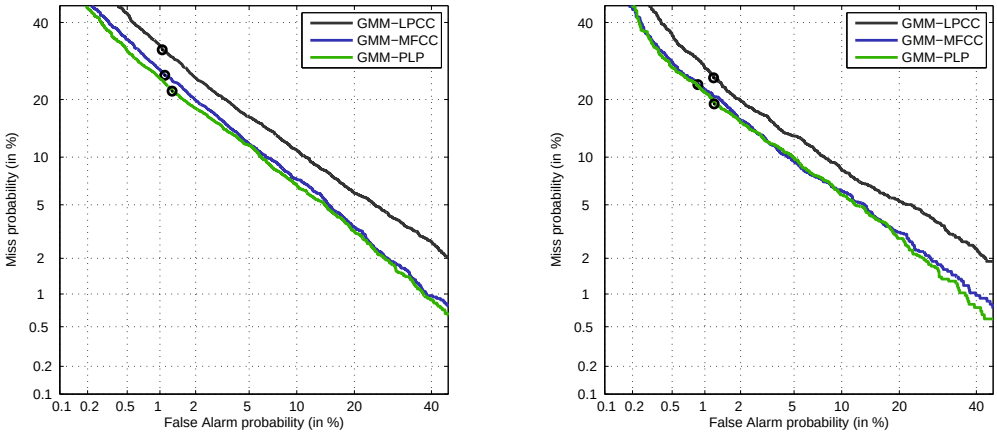

Figure 7.2: Performance comparison of the three baseline GMM systems with TNorm on the NIST 2006 Evaluation corpus. Left: core test results (DET1), right: English-only condition (DET3)

As expected, the evaluation on the English-only condition set yields lower EERs as well as lower DCF values. In contrast to the core test, where the PLP subsystems outperforms the other two subsystems in both measurements, the MFCC subsystem achieves the lowest EER on this subset.

Table 7.3: Detailed comparison of the EER and the DCF for gender dependent GMM subsystems on the NIST 2006 SRE task. Results are reported for core test and the English-only condition.

	GMM	core test		English-only	
	System	EER (%)	DCF	EER (%)	DCF
male	LPCC	9.3	0.0385	7.15	0.0302
	MFCC	7.57	0.0325	5.27	0.0261
	PLP	7.26	**0.0315**	5.80	**0.0251**
female	LPCC	11.36	0.0455	10.42	0.0406
	MFCC	9.02	0.0394	8.79	0.0355
	PLP	8.28	**0.0357**	8.26	**0.0338**
both gender	LPCC	10.49	0.0426	9.18	0.0338
	MFCC	8.39	0.0365	7.39	0.0319
	PLP	8.03	**0.0345**	7.50	**0.0310**

The comparison of the results on these two set points out that on the core test the LPCC and MFCC Equal Error Rates are about 14% (rel.) greater than on the English-only condition. Interestingly, the EER the PLP subsystem only drops about 7% (rel.). It can be inferred that the PLP subsystem is more robust against language variabilities. However, it has to be considered that this behaviour can also be caused by more general effects like channel or session variabilities.

7.4 Classifying Patterns of Variable Length

The frame-based classification method which has been described in in detail in section 6.4 directly classifies parameterized speech frames. Due to the fact that these frames consist of a fixed dimension they are applicable with nonparametric classification methods like SVM or KLR.

However, the direct classification of such feature vectors arises numerous disadvantages of which two are outlined in the following. One major problem is the enormous amount of speech data during enrollment. Through using a frame shift of 10 ms in a text-independent speaker verification system and 1 minute of speech per speaker we are faced with 6000 frames per speaker; this is not feasible for discriminative kernel classifiers by one of the described classification schemes. The second clear disadvantage of the direct classification of frames is the fact that this method discriminates frames whereas our main aim is to discriminate speakers. Consequently, we either have to find a classifier which can deal with patterns of variable length or a projection of variable length patterns to fixed length vectors. Through such a method of mapping variable length patterns to fixed length vectors it is possible to apply standard classification methods like SVMs.

Up to date there exist some approaches in speech recognition where a sequence of observations is first split into segments by a standard speech recognizer where each segment represent a speech unit such as a single monophone. In [Ganapathiraju, 2001] these segments were then divided into three subsegments according to a $3-4-3$ ratio. The frames within each subsegment were averaged and finally concatenated to build a fixed size vector representing a single speech segment. Then the SVM was used to post classify the monophones using N-best lists of the speech recognizer or to rescore the phoneme lattice [Stuhlsatz et al., 2006].

Alternatively, one can also apply a model based mapping instead of mapping variable length patterns to fixed length vectors. The length of the resulting vector is then independent of the data but it depends on the number of used parameters in the statistical model. Such a popular mapping is the **Fisher kernel** introduced by [Jaakkola and Haussler, 1999a]. The Fisher kernel is a kind of similarity measure between two sequences. Given a pre-trained generative model and the corresponding parameterization vector $\boldsymbol{\lambda}$, then the probability that this model generates the observation sequence $\boldsymbol{X}$ is denoted

by $P(\boldsymbol{X}, \boldsymbol{\lambda})$. A fixed length vector can be constructed by computing the derivatives of the log likelihood of the model $P(\boldsymbol{X}, \boldsymbol{\lambda})$, with respect to each of the parameters of the generative model:

$$U(\boldsymbol{X}) = \nabla_{\boldsymbol{\lambda}} \log P(\boldsymbol{X}, \boldsymbol{\lambda}). \tag{7.1}$$

The vector $U(\boldsymbol{X})$ of equation (7.1) is called the **Fisher score**. For two observation sequences $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ with the corresponding **Fisher information matrix** $\boldsymbol{F} = E(U(\boldsymbol{X}_1)U(\boldsymbol{X}_2))$ the Fisher kernel returns a scalar $k(\boldsymbol{X}_1, \boldsymbol{X}_2)$ where

$$k(\boldsymbol{X}_1, \boldsymbol{X}_2) = (\nabla_{\boldsymbol{\lambda}} \log P(\boldsymbol{X}_1, \boldsymbol{\lambda}))^\top \boldsymbol{F}^{-1} \nabla_{\boldsymbol{\lambda}} \log P(\boldsymbol{X}_2, \boldsymbol{\lambda}). \tag{7.2}$$

However, for large datasets the Fisher information matrix $\boldsymbol{F}$ becomes very large, which makes an inversion difficult to compute. In [Jaakkola and Haussler, 1999a] it is shown that in this case $\boldsymbol{F}$ can be replaced by the Identity matrix $\boldsymbol{I}$.

Interestingly, a generalization of the Fisher kernel was proposed by [Smith and Gales, 2002] and is referred to as a **score space** approach. This variant uses the ratio between the likelihood $P(\boldsymbol{X}, \boldsymbol{\lambda}_1)$ that the observation sequence is generated by model $\boldsymbol{\lambda}_1$ and $P(\boldsymbol{X}, \boldsymbol{\lambda}_2)$ that the observation sequence is generated by model $\boldsymbol{\lambda}_2$:

$$\phi(\boldsymbol{X}) = \log \frac{P(\boldsymbol{X}, \boldsymbol{\lambda}_1)}{P(\boldsymbol{X}, \boldsymbol{\lambda}_2)} \tag{7.3}$$

Another popular scheme for performing kernel classifiers on variable length patterns is the **Generalized Linear Discriminant Sequence** (GLDS) kernel [Campbell, 2002]. The latest and most powerful solution of this problem is the GMM supervector modeling concept, that we will describe in the following section.

7.5 GMM Supervector Modeling

The GMM supervector modeling concept was introduced by [Campbell et al., 2006]. In this approach the connection between discriminative classifiers and GMMs is based on the similarity measurement of GMMs. A natural choice of computing the similarity of two probability distributions is the **Kullback-Leibler** (KL) divergence. The KL divergence between the distributions $p(\boldsymbol{x}|\boldsymbol{\lambda}_a)$ and $p(\boldsymbol{x}|\boldsymbol{\lambda}_b)$ is defined as:

$$\mathcal{D}(p(\boldsymbol{x}|\boldsymbol{\lambda}_a)||p(\boldsymbol{x}|\boldsymbol{\lambda}_b)) = \int p(\boldsymbol{x}|\boldsymbol{\lambda}_a) \log \left(\frac{p(\boldsymbol{x}|\boldsymbol{\lambda}_a)}{p(\boldsymbol{x}|\boldsymbol{\lambda}_b)} \right) d\boldsymbol{x}. \tag{7.4}$$

Using Gaussian mixture models with parameter vector $\boldsymbol{\lambda} = \{\boldsymbol{c}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ to model $p(\boldsymbol{x}|\boldsymbol{\lambda}_a)$ and $p(\boldsymbol{x}|\boldsymbol{\lambda}_b)$ no closed form expression exists and the divergence $\mathcal{D}(p(\boldsymbol{x}|\boldsymbol{\lambda}_a)||p(\boldsymbol{x}|\boldsymbol{\lambda}_b))$ has to be estimated. In [Hershey and Olsen, 2007] several estimation methods like the Monte-Carlo method are investigated.

Instead of the divergence in equation (7.4) we use the log-sum inequality presented by [Cover and Thomas, 1991] to approximate the KL divergence of the two GMMs. This divergence is upper bounded as follows [Do, 2003]:

$$\begin{aligned} \mathcal{D}(p(\boldsymbol{x}|\boldsymbol{\lambda}_a)||p(\boldsymbol{x}|\boldsymbol{\lambda}_b)) \leq\ & \mathcal{D}(\boldsymbol{c}^a||\boldsymbol{c}^b) \\ & + \sum_{i=1}^{C} c_i \mathcal{D}(\mathcal{N}(\cdot; \mu_i^a, \boldsymbol{\Sigma}_i^a)||\mathcal{N}(\cdot; \mu_i^b, \boldsymbol{\Sigma}_i^b)). \end{aligned} \tag{7.5}$$

In case of MAP adaptation, and when only the GMM means were adapted from the background model, i.e. $\boldsymbol{c}^a = \boldsymbol{c}^b$ and $\boldsymbol{\Sigma}_i^a = \boldsymbol{\Sigma}_i^b$, the approximation in equation (7.5) leads to the estimate (upper bound) $\mathcal{D}_e$ as follows:

$$\mathcal{D}_e(\boldsymbol{\mu}_a||\boldsymbol{\mu}_b) = \sum_{i=1}^{C} c_i (\boldsymbol{\mu}_i^a - \boldsymbol{\mu}_i^b)^\top \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{\mu}_i^a - \boldsymbol{\mu}_i^b). \tag{7.6}$$

If $\boldsymbol{\Sigma}_i$ is a diagonal covariance matrix, equation 7.6 results in a similarity measure between the two mixture models that is homogenous with the square of the Euclidean distance between the weighted mean vectors $\boldsymbol{\mu}^a$ and $\boldsymbol{\mu}^b$.

Therefore we consider the Euclidean distance between two vectors $\Phi(\boldsymbol{\mu}^a)$ and $\Phi(\boldsymbol{\mu}^b)$ in feature space F. This distance can be computed in form of

kernel functions as

$$\begin{aligned}\mathcal{D}(\boldsymbol{\mu}^a||\boldsymbol{\mu}^b) &= \|\Phi(\boldsymbol{\mu}^a) - \Phi(\boldsymbol{\mu}^b)\|_2^2 \\ &= K(\boldsymbol{\mu}^a, \boldsymbol{\mu}^a) - 2K(\boldsymbol{\mu}^a, \boldsymbol{\mu}^b) + K(\boldsymbol{\mu}^b, \boldsymbol{\mu}^b). \end{aligned} \tag{7.7}$$

Equating now the distance $\mathcal{D}_e(\boldsymbol{\mu}_a||\boldsymbol{\mu}_b)$ of equation (7.6) and $\mathcal{D}(\boldsymbol{\mu}^a||\boldsymbol{\mu}^b)$ of equation (7.7) we can find the corresponding inner product which is the kernel function

$$\begin{aligned}K(\boldsymbol{\mu}^a, \boldsymbol{\mu}^b) &= \Phi(\boldsymbol{\mu}^a)\Phi(\boldsymbol{\mu}^b) \\ &= \sum_{i=1}^{C} (\sqrt{c_i}\boldsymbol{\Sigma}_i^{-\frac{1}{2}}\boldsymbol{\mu}_i^a)^\top (\sqrt{c_i}\boldsymbol{\Sigma}_i^{-\frac{1}{2}}\boldsymbol{\mu}_i^b). \end{aligned} \tag{7.8}$$

This kernel only contains the means $\boldsymbol{\mu}^a$ and $\boldsymbol{\mu}^b$ of Gaussian mixture models that are weighted by the corresponding mixture weights and covariances.

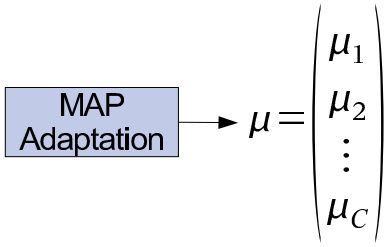

Figure 7.3: Illustration of the GMM supervector construction. All means of a single GMM are concatenated into a so called GMM supervector.

We can now apply this kernel to speaker recognition as follows: Speaker specific GMMs are adapted from the UBM for all background and target speakers. From these adapted models all resulting means of each GMM are concatenated into single supervectors as shown in figure 7.3 and a discriminative kernel classifier can be trained on this data.

7.5.1 GMM Supervector Experiments

In our first GMM supervector experiment we trained SVM and SKLR models with supervectors generated from background and target speakers using the linear kernel given in equation (7.8). We applied a **one-vs-UBM** training procedure where the target vector of a speaker is trained against the whole UBM set. During testing we first created a GMM on the acoustic data of the

test utterance. This model was adapted from the UBM in the same way as the client models, and finally the GMM supervector of the test segment was classified by the classifier of the claimed target speaker.

Based on the structure of the dot product kernel given in equation (7.8) we can precompute the results of the discriminative classifiers up to a single dot product. For the case of SVM this results in:

$$f(\boldsymbol{\mu}) = \left(\sum_{i=1}^{M} \alpha_i y_i \Phi(\boldsymbol{\mu}_i) \right)^{\top} \Phi(\boldsymbol{\mu}) + b_0 \tag{7.9}$$

with M support vectors, corresponding class labels y_i and the bias b_0. During evaluation we only have to calculate a single dot product between the precomputed bracket of equation (7.9) and the GMM supervector of the test segment.

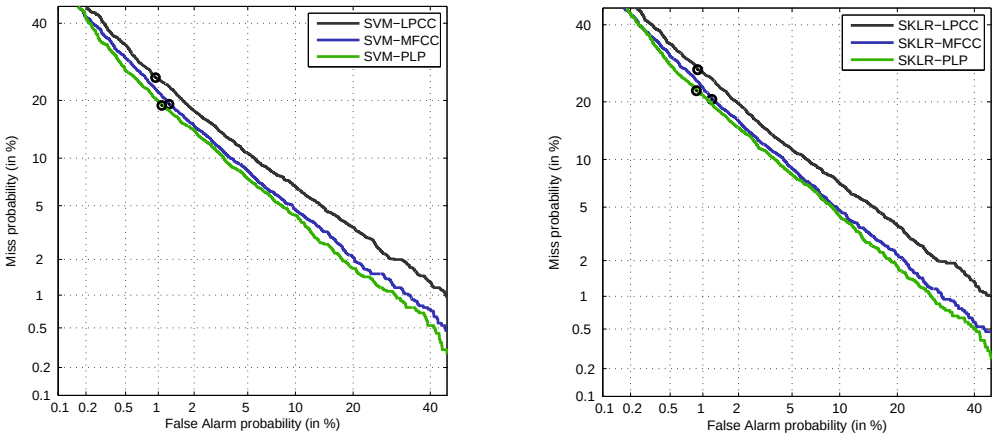

Figure 7.4: Performance comparison of SVM systems (left) and SKLR systems (right) using GMM supervectors on the NIST 2006 Evaluation corpus (core test).

The DET curves of this GMM supervector system are presented in figure 7.4. We applied the SVM as well as the SKLR classifier to GMM supervectors of the LPCC, MFCC and PLP subsystems.

Additionally, we summarize the results of the SVM and SKLR classifiers in table 7.4 and table 7.5, respectively. In comparison to the standard GMM system both discriminative classifier increased the recognition performance. The best GMM subsystem (the PLP system) achieved a DCF value of 0.0345 compared to 0.0295 and 0.0312 of the corresponding SVM and SKLR sub-

systems respectively. On the average of all three subsystems the DCF values of the SVM approach decrease more than 13% (rel.). However, applying the SKLR classifier the DCF values are about 9% (rel.) lower than the values of the GMM systems.

Table 7.4: Detailed comparison of EER and DCF for gender dependent SVM subsystems using GMM supervectors on the NIST 2006 SRE task.

	System	EER (%)	DCF
male	SVM_LPCC	7.13	0.0316
	SVM_MFCC	6.11	0.0286
	SVM_PLP	5.67	**0.0274**
female	SVM_LPCC	8.57	0.0366
	SVM_MFCC	6.90	0.0332
	SVM_PLP	6.80	**0.0309**
both gender	SVM_LPCC	7.86	0.0345
	SVM_MFCC	6.58	0.0315
	SVM_PLP	6.29	**0.0295**

Table 7.5: Detailed comparison of EER and DCF for gender dependent SKLR subsystems using GMM supervectors on the NIST 2006 SRE task.

	System	EER (%)	DCF
male	SKLR_LPCC	7.45	0.0331
	SKLR_MFCC	6.37	0.0288
	SKLR_PLP	6.11	**0.0283**
female	SKLR_LPCC	9.02	0.0390
	SKLR_MFCC	7.30	0.0350
	SKLR_PLP	7.15	**0.0324**
both gender	SKLR_LPCC	8.25	0.0365
	SKLR_MFCC	6.92	0.0324
	SKLR_PLP	6.70	**0.0312**

7.6 Intersession Variability Compensation

The compensation of intersession variabilities is considered to be one of the major problems in speaker recognition. These variabilities occur between training and testing by using different telephone handsets and/or different transmission channels. One of the first methods of compensating these variabilities in the acoustic space was the **Feature Mapping** approach, proposed by [Reynolds, 2003]. In the GMM framework [Kenny et al., 2005] proposed the **Factor Analysis** method. To address the problem in the SVM framework, the **Nuisance Attribute Projection** (NAP) was proposed by [Solomonoff et al., 2005].

7.6.1 Nuisance Attribute Projection

In addition to feature mapping in the acoustic space we need to compensate the session variabilities of speakers (see section 2.4) in the GMM-space. This compensation of channel and session variabilities in the SVM expansion space can be realized by the **Nuisance Attribute Projection** (NAP), introduced by [Solomonoff et al., 2005], which is based on the well known **Principal Component Analysis** (PCA). This projection tries to remove the subspaces containing nuisance variabilities by a particular projection of the GMM supervectors. Therefore, the NAP method constructs a new kernel in the feature space by two speakers $\boldsymbol{u}_a$ and $\boldsymbol{u}_b$:

$$K(\boldsymbol{u}_a, \boldsymbol{u}_b) = \langle \boldsymbol{P}\Phi(\boldsymbol{u}_a), \boldsymbol{P}\Phi(\boldsymbol{u}_b) \rangle \tag{7.10}$$

$$= \Phi(\boldsymbol{u}_a)^\top \boldsymbol{P}\Phi(\boldsymbol{u}_b) \tag{7.11}$$

$$= \Phi(\boldsymbol{u}_a)^\top (\boldsymbol{I} - \boldsymbol{S}\boldsymbol{S}^\top)\Phi(\boldsymbol{u}_b). \tag{7.12}$$

The projection matrix $\boldsymbol{P}$ is represented by $(\boldsymbol{I} - \boldsymbol{S}\boldsymbol{S}^\top)$ with the low rank matrix $\boldsymbol{S}$ whose columns are orthonormal. It can be shown that the solution of an optimal projection matrix $\boldsymbol{P}$ is equal to the principle components of the supervectors [Campbell et al., 2006]. Therefore, the covariance matrix

$$C = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\mu}_i \boldsymbol{\mu}_i^\top \tag{7.13}$$

is set up by all M GMM supervectors $\boldsymbol{\mu}_i$. The eigenvectors of the covariance matrix C have to be computed to determine the principle components. The projection tries to remove the subspaces containing nuisance variabilities by the projection:

$$\boldsymbol{\mu}^* = (\boldsymbol{I} - \boldsymbol{S}\boldsymbol{S}^\top)\boldsymbol{\mu} \tag{7.14}$$

$$= \boldsymbol{\mu} - \boldsymbol{S}(\boldsymbol{S}^\top\boldsymbol{\mu}) \tag{7.15}$$

where $(\boldsymbol{I} - \boldsymbol{S}\boldsymbol{S}^\top)$ is the complementary PCA projection with matrix $\boldsymbol{S}$ containing the eigenvectors corresponding to the largest eigenvalues of the PCA eigenvalue problem on features $\boldsymbol{\mu}$. In a further step we reduce the influence of speakers by subtracting the mean of supervectors corresponding to the same speaker:

$$C = \frac{1}{S}\sum_{s=1}^{S}\boldsymbol{W}_s \tag{7.16}$$

$$\boldsymbol{W}_s = \frac{1}{M_S}\sum_{i=1}^{M_S}(\boldsymbol{\mu}_{s,i} - \tilde{\boldsymbol{\mu}}_s)(\boldsymbol{\mu}_{s,i} - \tilde{\boldsymbol{\mu}}_s)^\top \tag{7.17}$$

where M_S represents the number of supervector and $\tilde{\boldsymbol{\mu}}_s$ the supervector mean of the s^{th} speaker in the set.

Resulting from the structure of the NAP kernel in equation (7.12) we do not need to project the supervectors of the test data into the subspace.

7.6.2 Experiments

For each of the three gender dependent subsystems we created a projection matrix $\tilde{\boldsymbol{P}}$ on data from the SRE 2004 corpus using the GMM models of section 2.6.1. The number of utilized eigenvectors for the projection is optimized on the development set. The influence of the number of utilized eigenvectors for the data projection is presented in figure 7.5.

The results obtained in the figure suggest that more than 48 but less than 80 eigenvalues lead to promising results. Especially the performance of the MFCC system decreases when more eigenvalues are selected. In the average we obtain the best improvement for all subsystems by selecting the 64 largest eigenvalues.

Therefore, we used in the evaluation experiments the corresponding eigenvectors of the 64 largest eigenvalues for the data projection. The results of

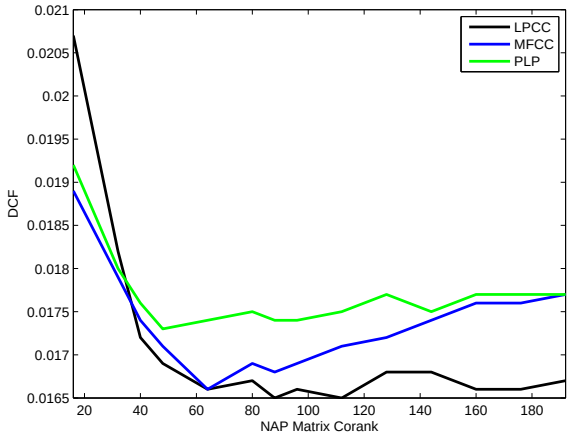

Figure 7.5: Influence of the number of utilized eigenvectors on the SVM supervector system. Comparison of the resulting DCF on the NIST SRE05 1conv-1conv development test, both gender.

the SVM and SKLR subsystems on the core test using the Nuisance Attribute Projection are presented in figure 7.6.

Table 7.6: Comparison of EER and DCF for gender dependent SVM systems using NAP on the NIST 2006 SRE task.

	System	core test		English-only	
		EER (%)	DCF	EER (%)	DCF
male	SVM_LPCC	3.88	**0.0186**	2.15	**0.0109**
	SVM_MFCC	3.82	0.0192	2.29	0.0117
	SVM_PLP	3.44	0.0193	2.03	0.0115
female	SVM_LPCC	4.70	**0.0205**	3.50	**0.0169**
	SVM_MFCC	4.95	0.0226	4.22	0.0182
	SVM_PLP	4.89	0.0237	3.86	0.0204
both gender	SVM_LPCC	4.26	**0.0200**	3.02	**0.0147**
	SVM_MFCC	4.57	0.0212	3.50	0.0157
	SVM_PLP	4.24	0.0220	3.18	0.0170

Whereas the LPCC achieved the highest error rates and DCF values in the GMM experiments, the DCF value of the SVM_LPCC is lower than the DCF values of the MFCC and PLP systems. The best equal error rates were achieved by the supervector SKLR_PLP approach with an EER of 4.10%

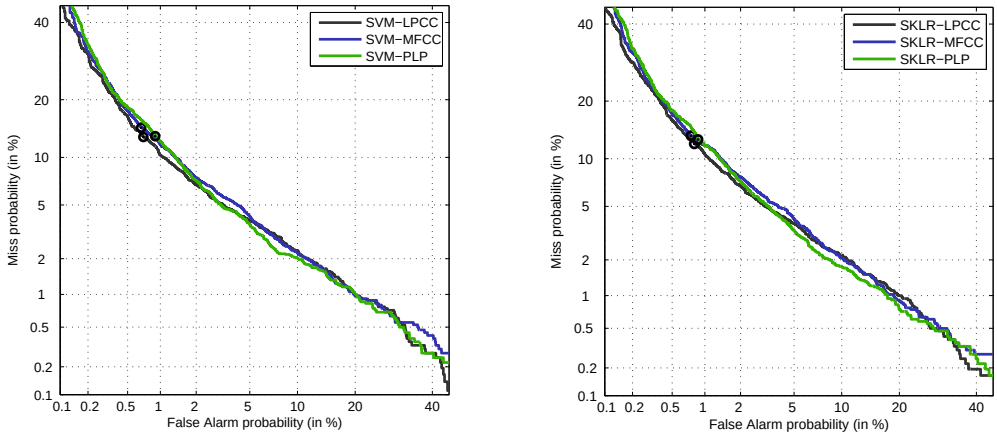

Figure 7.6: Performance comparison of gender dependent SVM (left) and SKLR systems (right) using NAP on the NIST 2006 Evaluation corpus.

Table 7.7: Comparison of EER and DCF for gender dependent SKLR systems using NAP on the NIST 2006 SRE task.

	System	core test		English-only	
		EER (%)	DCF	EER (%)	DCF
male	SKLR_LPCC	3.71	0.0189	1.89	**0.0109**
	SKLR_MFCC	3.77	0.0189	2.16	0.0122
	SKLR_PLP	3.25	**0.0184**	2.05	0.0112
female	SKLR_LPCC	4.55	**0.0208**	3.60	**0.0168**
	SKLR_MFCC	4.99	0.0222	4.13	0.0178
	SKLR_PLP	4.75	0.0232	3.86	0.0197
both gender	SKLR_LPCC	4.21	**0.0201**	3.02	**0.0149**
	SKLR_MFCC	4.51	0.0209	3.55	0.0157
	SKLR_PLP	4.10	0.0214	3.13	0.0164

compared to 4.21% of the SKLR_LPCC and 4.51% of the SKLR_MFCC system. In comparison to the SVM approach, which achieved Equal Error Rates of 4.24%, 4.26% and 4.57% on the equivalent subsystems, the SKLR has the lowest error rates. In contrast to the EER, the lowest DCF values are achieved by the LPCC subsystems. Interestingly, the SVM_LPCC system achieved a slightly lower DCF value compared to the SKLR_LPCC system.

For a better comparison to other published systems we also present our results of the English-only condition. Together with the outcomes of our best

subsystems we give a summary of competitive state-of-the-art single systems in table 7.8.

Table 7.8: Summary of recently published results based on single systems on the NIST 2006 SRE task, both gender.

System	core test		English-only	
	EER (%)	DCF	EER (%)	DCF
[Shriberg and Ferrer, 2007]	-	-	4.00	0.0197
[Matejka et al., 2007]	5.40	0.0283	3.61	0.0171
proposed SVM_LPCC system	4.26	0.0200	3.02	0.0147
proposed SKLR_LPCC system	4.21	0.0201	3.02	0.0149

All systems in table 7.8 are based on Support Vector Machines. The approach presented in [Matejka et al., 2007] also applies the supervector concept as well as nuisance attribute projection to remove channel variabilities. Corresponding eigenvector of the first 40 eigenvalues are used for the projection. The system used in [Shriberg and Ferrer, 2007] is a **maximum likelihood linear regression** (MLLR) system that additionally makes use of information from a speech recognizer to obtain speaker specific model transforms (for more details see [Stolcke et al., 2007]).

As we can see in table 7.8 our proposed speaker recognition subsystems based on LPCC clearly outperform recently published systems. In contrast to our best performing subsystem, both published approaches are based on cepstral features. Another difference to [Matejka et al., 2007] is the number of eigenvectors used for NAP. Relating to our findings presented in figure 7.5 we used 64 eigenvectors for the projection.

In the next section we will apply different score combination methods on the results of our three subsystems and compare these fusions with the single subsystems.

7.7 Fusion Schemes

As we have seen in section 6.4 and 7.5 there exist several different methods like SVMs or SKLR for classifying and detecting speakers. We also can apply different feature extraction methods like LPCC, MFCC or PLP. In numerous publications, the successful combination of independently developed systems has been shown. Most of these systems combine so called **low** and **high-**

level systems. In general low-level systems are based on acoustic features like MFCC and high-level systems represent **pitch-based**, **phone-based** and **word-based** systems. In this work we explore fusion of different low-level subsystems, namely LPCC, MFCC and PLP subsystems. In the following we discuss three different fusion approaches.

In **feature-level** fusion, different feature vectors from several modalities or different feature extraction algorithms are combined directly. In this approach one has to guarantee that the feature vectors are synchronous, i.e. the same frame-shift and the same time index is used in the feature extraction. The combination can then be realized by concatenating the feature vectors in each time step. Examples of feature-level fusion of features derived from different modalities can be found in [Chibelushi et al., 1997].

The second approach, the **GMM-Level** fusion, is based on the GMM supervector method of section 7.5. Different GMM supervectors can be concatenated to form combined feature vectors for the following discriminative classifier. Notwithstanding, the resulting dimension of these concatenated supervectors is very high and additional subspace transformations like **Linear Discriminant Analysis** (LDA) should be used to extract relevant information for the classifier.

The **score-level** fusion is the third approach and combines the scores from different subsystems in a post classification step. In this approach it is possible to fuse several independent systems. Fusion performed in this work is based on the score-level approach.

However, there are different ways of fusing scores, e.g., weighted sum

$$S_{\text{sum}}(\boldsymbol{X}) = \sum_{i=1}^{M} \omega_i S(\boldsymbol{X}_i) \tag{7.18}$$

where the score $S(\boldsymbol{X}_i)$ of each subsystem is weighted by its own weight ω_i. The weights can be estimated on a development set by minimizing a certain error criterion. Other approaches are fusion by maximum rule

$$S_{\max}(\boldsymbol{X}) = \max S(\boldsymbol{X}_i), \tag{7.19}$$

where the final score is obtained by finding that subsystem with the highest score.

Another approach of score fusion is concatenate all scores of the subsystem in an M-dimensional score vector $\boldsymbol{S}$, where the dimension M is equal to the number of used subsystems. These score vectors serves as input of a new

binary classifier with one impostor and one client class. Again, a number of classification approaches like GMMs, SVM or **Neural Networks** (NNs) are candidates for this approach.

7.7.1 Score Fusion Experiments

In this thesis we restrict to three simple score fusion techniques: Logistic Regression (5.51), weighted sum (7.18) and average. The average is identical to the weighted sum with equal weights for each score.

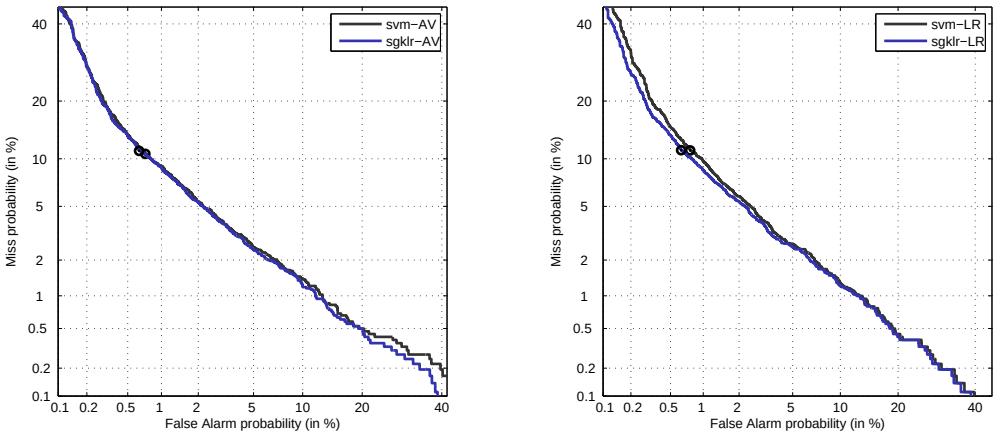

Figure 7.7: Performance comparison of fused SVM and SKLR systems using simple average (left) and logistic regression (right) on the NIST 2006 Evaluation corpus.

Fusion results are presented in figure 7.7 and table 7.9. As we can see from figure 7.7 both classifiers achieve comparable DET curves. In all cases the SKLR systems performs slightly superior to the SVM system but not significantly. Comparing equal error rates for the Logistic Regression fusion in table 7.9 we can see that the fused SKLR system achieves equal error rates that are about 4.0% (core test) and 4.5% (English-only) lower than the SVM results.

Again, we compare our results with findings of competitive state-of-the-art systems in table 7.10. The fused system presented in [Matejka et al., 2007] is based on several SVM low-level subsystems that were independently developed by different groups. Their best subsystem also applies the supervector concept as well as nuisance attribute projection. Score fusion is realized by Logistic Regression.

Table 7.9: Comparison of the EER and the DCF for different fused SVM subsystem on the NIST 2006 SRE task, both gender.

	GMM	core test		English-only	
	System	EER (%)	DCF	EER (%)	DCF
SVM	Average	3.47	0.0178	2.48	0.0129
	Weighted Sum	3.52	**0.0176**	2.43	0.0129
	Logistic Regression	3.46	**0.0176**	2.43	**0.0127**
SKLR	Average	3.43	0.0174	2.43	0.0128
	Weighted Sum	3.44	0.0174	2.42	0.0130
	Logistic Regression	3.32	**0.0173**	2.32	**0.0128**

Campbell et al. presented the performance of their fused system on the NIST 2006 task (post-evaluation) in [Campbell et al., 2007]. This system combines several low-level subsystems based on GMMS, SVM-NAP and SVM with GLDS kernel [Campbell, 2002]. Additionally they used high-level (word-based) subsystems and fused their scores with a Neural Network.

The system given in [Shriberg and Ferrer, 2007] also uses different SVM subsystems that are based on GMM supervectors as well as on MLLR (see 7.6.2). They combine these low-level subsystems with high-level systems based on prosody, phones and words.

Table 7.10: Summary of recently published results based on fused systems on the NIST 2006 SRE task.

System	Fusion	core test		English-only	
		EER	DCF	EER	DCF
[Campbell et al., 2007]	Neural Net	4.40	0.0220	2.70	0.0140
[Shriberg and Ferrer, 2007]	feature-level	-	-	2.59	0.0144
[Matejka et al., 2007]	LR	3.83	0.0214	2.32	0.0126
proposed SVM	LR	3.46	0.0176	2.43	0.0127
proposed SKLR	LR	3.32	0.0173	2.32	0.0128

Table 7.10 shows that the performance of our fused SKLR system is comparable with the results presented in [Matejka et al., 2007] on the English-only condition. However, the recognition environment introduced in this thesis together with the novel SKLR classifier achieved the lowest DCF value as well as the lowest EER on the core test. As can be seen in table 7.10 the fusion of subsystems based on different speech parameterization techniques leads to

better results than the fusion of different classification approaches.

In general we can conclude that the fusion of our subsystem scores complement each other and that the fused SVM as well as the SKLR systems outperform all single subsystems presented in section 7.6.2.

Since the SVM output scores are not restricted to a defined range, we will apply a score calibration on the unmoderated SVM scores in the next section.

7.7.2 Score Calibration of SVM output scores

In this final experiment we investigate a calibration of unmoderated SVM score for a better fusion of the subsystem scores. Therefore we transformed the scores of the SVM subsystems by Platt's algorithm to probabilities and fused these moderated scores by different fusion techniques. The parameters for Platt's algorithm (see section 5.3.4) were estimated on the development set. It is important to note that since no speaker dependent development data could be used for parameter optimization, the final calibration parameters are speaker independent. The results of the calibrated and fused subsystems are presented in table 7.11.

Table 7.11: Comparison of the EER and the DCF for different fused SVM subsystems on the NIST 2006 SRE task. SVM scores are calibrated by a sigmoid function.

Fusion	EER (%)	DCF
Average	3.46	0.0177
Weighted Sum	3.50	0.0177
Logistic Regression	3.42	0.0176

Table 7.11 shows that a speaker independent score calibration has no influence on the DCF value. Only the EER slightly decreased in all cases. For example the EER of the additionally calibrated LR fusion decreases about 1.2% compared to the un-calibrated one.

7.8 Summary

In this chapter we presented a speaker verification/detection system that successfully combines discriminative and generative classifiers. The system is based on three different acoustic parameterization techniques and fuses the scores of these subsystems.

First, we reviewed related methods for classifying patterns of variable length in section 7.4 and gave a brief overview of the GMM supervector concept in section 7.5. Secondly, we described the Nuisance Attribute Projection (NAP) as compensation of intersession variabilities in section 7.6.1.

In the following we presented our experiments which were carried out with two discriminative kernel classifiers, namely SVM and SKLR, on the current NIST SRE 2006 data. Compared to traditional GMM systems our new SKLR supervector system improves the EER on the core test about 50% from 8.03% to 4.10% and the DCF value from 0.0345 to 0.0214 using the PLP-subsystem. We achieved the best result by fusing the SKLR subsystems using Logistic Regression. While the SVM system achieved an EER of 3.46% and a DCF value of 0.0176, our novel SKLR approach outperforms the SVM system with an EER of 3.32% and a DCF value of 0.0173.

8 Conclusion

This thesis aimed to address the integration and combination of different discriminative classifiers, like Support Vector Machines (SVM) and Kernel Logistic Regression (KLR) into the field of speaker recognition.

First, we gave an overview of how speaker recognition is linked to the field of biometrics and discussed advantages and disadvantages of identifying or verifying claimants by their voice. Furthermore, the most common techniques for speaker recognition were surveyed in chapter 2, including speaker verification based on the likelihood ratio test with Gaussian Mixture Models (GMMs). Moreover, chapter 3 reviewed the speech production in human beings and described different parameterization methods of speech data that are applicable for speaker recognition.

In chapter 4 we revisited the theoretical background of statistical learning theory and addressed the principles of structural risk minimization with the aim to lay the foundation for the discriminative classifiers presented in this thesis.

After describing the popular Support Vector Machine and its abilities in chapter 5 we introduced the Kernel Logistic Regression (KLR) as a non-linear expansion of Logistic Regression, a well known method in the world of statistics. Since memory requirements for parameter optimization of the standard KLR depend on the size of the training data this approach is not feasible for large training data.

Due to this fact we developed the so called Sparse Kernel Logistic Regression (SKLR) that provides a sparse solution where the memory requirements scale with the size of the solution. The solution is achieved by a forward selection of adequate training samples. In comparison to the SVM the SKLR also provides a probability output of class membership instead of a distance measure.

Rooted in the framework of realizing speaker recognition by discriminative classifiers we presented our novel speaker recognition environment as one major result of this work. The discriminative classifiers were integrated into the recognizer in two different ways: The first recognition approach is designed especially for limited training data as it is often the case in real world security

applications. The second realization is developed for speaker detection with an extended amount of speech data.

With reference to the first approach we demonstrated the power of SVM and SKLR classifiers in chapter 6 on small real world identification and verification problems, where the classifiers were applied directly on feature vectors in the same way as in the GMM approach. While the SKLR inherently provides a probability output, the unmoderated SVM outputs had to be transformed to probability measures. In case of speaker identification we additionally applied techniques for combining all probabilities of the one-vs-one pairs into multi-class probabilities.

Importantly, both discriminative classifiers achieved significantly better identification and verification rates than the traditional GMM method on the POLYCOST corpus. Although the recognition rates of SVM and SKLR are comparable to each other it is important to notice that the SKLR also provides a very sparse solution that contains less selected vectors than the SVM. Thus, it clearly outperforms the SVM concerning classification speed. Due to the computational cost of SVM and SKLR this frame-based classification method is not feasible for large speech utterances.

Therefore we set up our second state-of-the-art speaker recognition system designed for speaker detection on large datasets which has been reported in chapter 7. This system is based on the Universal Background Model (UBM) approach and uses gender-dependent GMMs for background models as well as MAP adapted GMMs for specific speaker models. We compared several approaches of projecting variable length speech utterances to fixed sized vectors that could be classified by our discriminative classifiers. As a result of this comparison the GMM supervector concept was chosen and successfully integrated into the recognition system. Further, the discriminative classifiers were then applied to the GMM supervectors.

We demonstrated the outstanding recognition ability of the SKLR in several experiments on the NIST 2006 Speaker Recognition Evaluation task and compared our results with comparable published systems.

Simultaneously, we also compared different feature extraction methods in this work to identify the most suitable one for speaker recognition. After a comparison of different parameterization techniques this goal was accomplished by a fully text-independent speaker recognition system that combines classification results of different feature extraction methods. We could show that the fusion of subsystem scores complement each other and that the final

SVM and SKLR systems outperform all single subsystems.

Significantly, the combination of different subsystems also benefits from the probability output of the SKLR because all scores are well calibrated and lie in a defined range between zero and one.

In general the performance of speaker recognition systems is deeply influenced by the amount of speech data used during enrollment and test. This was also shown in chapter 6. In future works one needs to investigate this influence to SVM and SKLR. In case of using discriminative classifiers directly on feature vectors, more efficient optimization algorithms for SVM and SKLR would solve the problem of more speech data in the enrollment. However, at the same time alternative classification schemes that are more efficient than the used one-vs-one approach should be investigated. Future work also includes exploring alternative supervector constructions for smaller datasets. As a consequence of other construction methods that are not based on a MAP adaptation of speaker models, a development of different kernel functions is needed.

In conclusion we could prove that discriminative classifiers can be successfully applied to the field of speaker recognition in different ways. More generally, future research is needed to further improve and refine these new ideas in the field of speaker recognition.

Schriftliche Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Insbesondere habe ich nicht die Hilfe einer kommerziellen Promotionsberatung in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Magdeburg, den 19. Dezember 2007

Bibliography

List of authored publications

Marcel Katz, Hans-Günter Meier, Hans Dolfing, and Dietrich Klakow. Robustness of linear discriminant analysis in automatic speech recognition. In *16th International Conference on Pattern Recognition (ICPR)*, volume 3, pages 371–374, 2002. ISBN 0-7695-1695-X.

Marcel Katz, Sven E. Krüger, Martin Schafföner, Edin Andelic, and Andreas Wendemuth. Kernel methods for discriminant analysis in speech recognition. In *Proceedings of the Ninth International Conference "Speech and Computer" (SPECOM)*, pages 95–98, 2004. ISBN 5-7452-0110-x.

Marcel Katz, Martin Schafföner, Edin Andelic, Sven E. Krüger, and Andreas Wendemuth. Sparse kernel logistic regression for phoneme classification. In *Proceedings of the 10th International Conference "Speech and Computer" (SPECOM)*, pages 523–526, 2005. ISBN 5-7452-0110-x.

Marcel Katz, Sven E. Krüger, Martin Schafföner, Edin Andelic, and Andreas Wendemuth. Speaker identification and verification using support vector machines and sparse kernel logistic regression. In *Advances in Machine Vision, Image Processing, and Pattern Analysis, IWICPAS 2006, Proceedings*, volume 4153 of *Lecture Notes in Computer Science*, pages 176–184, Berlin/Heidelberg, 2006a. Springer. ISBN 3-540-37597-X.

Marcel Katz, Martin Schafföner, Edin Andelic, Sven E. Krüger, and Andreas Wendemuth. The iesk-magdeburg speaker detection system for the nist speaker recognition evaluation. In *Proceedings of NIST Speaker Recognition Evaluation*, 2006b.

Marcel Katz, Martin Schafföner, Edin Andelic, Sven E. Krüger, and Andreas Wendemuth. Sparse kernel logistic regression using incremental feature selection for text-independent speaker identification. In *IEEE Odyssey 2006 — The Speaker and Language Recognition Workshop, Proceedings*, pages 1–6, 2006c. ISBN 1-4244-0472-X.

Marcel Katz, Edin Andelic, Sven E. Krüger, Martin Schafföner, and Andreas Wendemuth. Discriminative kernel classifiers in speaker recognition. In *33rd German Annual Conference on Acoustics (DAGA)*, 2007a.

Marcel Katz, Martin Schafföner, Sven E. Krüger, and Andreas Wendemuth. Score calibrating for speaker recognition based on support vector machines and gaussian mixture models. In *9th IASTED International Conference on Signal and Image Processing (SIP)*, 2007b.

List of co-authored publications

Edin Andelic, Martin Schafföner, Sven E. Krüger, Marcel Katz, and Andreas Wendemuth. Iterative implementation of the kernel fisher discriminant for speech recognition. In *Proceedings of the Ninth International Conference "Speech and Computer" (SPECOM)*, pages 99–103, 2004. ISBN 5-7452-0110-x.

Edin Andelic, Martin Schafföner, Sven E. Krüger, Marcel Katz, and Andreas Wendemuth. Acoustic modelling using kernel-based discriminants. In *Proceedings of the 10th International Conference "Speech and Computer" (SPECOM)*, pages 139–142, 2005. ISBN 5-7452-0110-x.

Edin Andelic, Martin Schafföner, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. Kernel least squares models using updates of the pseudoinverse. *Neural Computation*, 18(12):2928–2935, 2006a.

Edin Andelic, Martin Schafföner, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. A hybrid HMM-based speech recognizer using kernel-based discriminants as acoustic models. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR)*, volume 2, pages 1158–1161, 2006b. ISBN 0-7695-2521-0.

Edin Andelic, Martin Schafföner, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. Updates for nonlinear discriminants. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 660–665, 2007.

Manfred Deutscher, Marcel Katz, and Sven Krüger. Learning about the environment by analyzing acoustic information. how to achieve predictability in unknown environments? (part ii). In Armada Manuel and Pablo

González de Santos, editors, *7th International Conference on Climbing and Walking Robots (CLAWAR 2004)*, pages 399–410. Springer, 2004. ISBN 978-3-540-22992.

Manfred Deutscher, Marcel Katz, and Sven Krüger. Sensing and rating different environmental states: A basic approach exemplified on moving in unknown terrain. *Informatik LIVE! Beiträge der 35. Jahrestagung der Gesellschaft für Informatik e.V*, 2:302–306, 2005.

Sven E. Krüger, Sebastian Barth, Marcel Katz, Martin Schafföner, Edin Andelic, and Andreas Wendemuth. Free energy classification at various temperatures for speech recognition. In *Proceedings of the Ninth International Conference "Speech and Computer" (SPECOM)*, pages 104–107, 2004. ISBN 5-7452-0110-x.

Sven E. Krüger, Martin Schafföner, Marcel Katz, Edin Andelic, and Andreas Wendemuth. Speech recognition with support vector machines in a hybrid system. In *Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, pages 993–996. ISCA, 2005a.

Sven E. Krüger, Martin Schafföner, Marcel Katz, Edin Andelic, and Andreas Wendemuth. Using support vector machines in a HMM-based speech recognition system. In *Proceedings of the 10th International Conference "Speech and Computer" (SPECOM)*, pages 329–332, 2005b. ISBN 5-7452-0110-x.

Sven E. Krüger, Martin Schafföner, Marcel Katz, Edin Andelic, and Andreas Wendemuth. Mixture of support vector machines for HMM based speech recognition. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR)*, volume 4, pages 326–329, 2006. ISBN 0-7695-2521-0.

Sven E. Krüger, Martin Schafföner, Marcel Katz, Edin Andelic, and Andreas Wendemuth. Support vector machines as acoustic models in speech recognition. In *33rd German Annual Conference on Acoustics (DAGA)*, pages 1–4, 2007.

Kinfe T. Mengistu, Marcel Katz, and Andreas Wendemuth. Acoustic modeling for a speaker-independent telephone-based spoken dialog system. In

Proceedings of the 12th International Conference "Speech and Computer" (SPECOM), 2007.

Martin Schafföner, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. Improved robustness of automatic speech recognition using a new class definition in linear discriminant analysis. In *Proceedings of the 8th European Conference on Speech Communication and Technology (EUROSPEECH)*, pages 2841–2844, 2003.

Martin Schafföner, Edin Andelic, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. Kernel fisher discriminants as acoustic models in HMM-based speech recognition. In George Kokkinakis, editor, *Proceedings of the 10th International Conference "Speech and Computer" (SPECOM)*, pages 349–352, 2005. ISBN 5-7452-0110-x.

Martin Schafföner, Edin Andelic, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. Limited training data robust speech recognition using kernel-based acoustic models. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 1137–1140, 2006. ISBN 1-4244-0469-X.

Martin Schafföner, Edin Andelic, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. Memory-efficient orthogonal least squares kernel density estimation using enhanced empirical cumulative distribution functions. In Merina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 417–424, 2007. ISBN 0-9727358-2-8.

Andre Stuhlsatz, Hans-Günter Meier, Marcel Katz, Sven E. Krüger, and Andreas Wendemuth. Support vector machines for postprocessing of speech recognition hypotheses. In *Proceedings of International Conference on Telecommunications and Multimedia (TEMU)*, 2006.

References

[Andelic et al. 2006] E. Andelic, M. Schafföner, M. Katz, S. E. Krüger, and A. Wendemuth. Kernel Least Squares Models using Updates of the Pseudoinverse. *Neural Computation*, 18(12):2928–2935, 2006.

[Aronszajn 1950] N. Aronszajn. Theory of Reproducing Kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[Auckenthaler et al. 2000] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas. Score Normalization For Text-Independent Speaker Verification Systems. *Digital Signal Processing*, 10:42–54, 2000.

[Ayer et al. 1955] M. Ayer, H. Brunk, G. Ewing, W. Reid, and E. Silverman. An Empirical Distribution Function for Sampling With Incomplete Information. *Annals of Mathematical Statistics*, 5:641–647, 1955.

[Baudat and Anouar 2001] G. Baudat and F. Anouar. Kernel-Based Methods and Function Approximation. In *International Joint Conference on Neural Networks*, pages 1244–1249, 2001.

[Bennett and Mangasarian 1992] K. P. Bennett and O. L. Mangasarian. Robust Linear Programming Discrimination of Two Linearly Inseparable Sets. *Optimization Methods and Software*, 1:23–34, 1992.

[Bimbot et al. 1994] F. Bimbot, G. Chollet, and A. Paoloni. Assessment Methodology for Speaker Identification and Verification Systems - an Overview of SAM-A Esprit Project 6819 - Task 2500. In *Proceedings of the ESCA Workshop on Automatic Speaker Recognition, Identification, and Verification*, pages 75–82, 1994.

[Bimbot et al. 2004] F. Bimbot, J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-García, D. Petrovska-Delacrétaz, and D. A. Reynolds. A Tutorial on Text-Independent Speaker Verification. *EURASIP Journal on Applied Signal Processing*, 4:430–451, 2004.

[Bonastre et al. 2005] J.-F. Bonastre, F. Wils, and S. Meignier. ALIZE, A Free Toolkit For Speaker Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 737–740, 2005.

[Boser et al. 1992] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, New York, NY, USA, 1992. ACM Press. ISBN 0-89791-497-X.

[Campbell 1997] J. P. Campbell. Speaker Recognition: A Tutorial. *Proceedings of the IEEE*, 85(9):1437–1462, 1997.

[Campbell 2002] W. Campbell. Generalized Linear Discriminant Sequence Kernels for Speaker Recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 161–164, Orlando, Florida, 2002.

[Campbell et al. 2006] W. Campbell, D. Sturim, D. Reynolds, and A. Solomonoff. SVM Based Speaker Verification using a GMM Super-Vector Kernel and NAP Variability Compensation. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 97–100, 2006.

[Campbell et al. 2007] W. Campbell, D. Sturim, W. Shen, D. Reynolds, and J. Navratil. The MIT-LL/IBM 2006 Speaker Recognition System: High-Performance Reduced-Complexity Recognition. In *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007.

[Chakroborty et al. 2007] S. Chakroborty, A. Roy, and G. Saha. Improved Closed Set Text-Independent Speaker Identification by combining MFCC with Evidence from Flipped Filter Banks. *International Journal of Signal Processing*, 4(2), 2007.

[Chibelushi et al. 1997] C. C. Chibelushi, J. S. D. Mason, and F. Deravi. Feature Level Data Fusion for Bimodal Person Recognition. In *6th International Conference on Image Processing and its Applications*, volume 1, pages 399–403, 1997.

[Clarkson and Moreno 1999] P. R. Clarkson and P. J. Moreno. On the use of Support Vector Machines for Phonetic Classification. In *Proc. ICASSP*. IEEE, 1999.

[Collobert and Bengio 2001] R. Collobert and S. Bengio. SVMTorch: Support Vector Machines for Large-Scale Regression Problems. *The Journal of Machine Learning Research*, 1:143–160, 2001.

[Cortes and Vapnik 1995] C. Cortes and V. N. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. ISSN 0885-6125.

[Cover and Thomas 1991] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley-Interscience, New York, USA, 1991. ISBN 0-471-06259-6.

[Davis and Mermelstein 1980] S. Davis and P. Mermelstein. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28:357–366, 1980.

[Do 2003] M. Do. Fast Approximation of Kullback-Leibler Distance for Dependence Trees and Hidden Markov Models. *IEEE Signal Processing Letters*, 10(4):115–118, 2003.

[Draper and Smith 1998] N. R. Draper and H. Smith. *Applied Regression Analysis.* Wiley & Sons, 1998. ISBN 0-471-17082-8.

[Duda et al. 2001] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* John Wiley & Sons, 2001.

[Durbin 1960] J. Durbin. The Fitting of Time Series Models. *Review of the International Institute of Statistics*, 28:233–243, 1960.

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

[Fanty and Cole 1991] M. Fanty and R. Cole. Spoken Letter Recognition. In R. Lippmann, J. Moody, and D. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 3, pages 220–226. Morgan Kaufmann Publishers Inc., 1991.

[Fisher 1936] R. A. Fisher. The Use of Multiple Measurements in Taxonomic Problems. In *Annals of Eugenics*, volume 7, pages 179–188. Cambridge University Press, 1936.

[Furui 1986] S. Furui. Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech Spectrum. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(1):52–59, 1986. ISSN 0096-3518.

[Galton 1888a] F. Galton. Personal Identification and Description. *Proceedings of the Royal Institution*, 12:346–360, 1888a.

[Galton 1888b] F. Galton. Personal Identification and Description. *Nature*, 38:173–177, 201–202, 1888b.

[Ganapathiraju 2001] A. Ganapathiraju. *Support Vector Machines For Speech Recognition*. PhD thesis, Mississippi State University, 2001.

[Gauvain and Lee 1994] J. L. Gauvain and C.-H. Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions on Speech and Audio Processing*, 2(2):291–298, 1994.

[Golub and van Loan 1996] G. H. Golub and C. F. van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, USA, 3rd edition, 1996. ISBN 0-8018-5414-8.

[Gravier 2004] G. Gravier. *Speech Signal Processing Toolkit (SPro)*, 2004. URL `http://www.irisa.fr/metiss/guig/spro.html`.

[Hager 1989] W. W. Hager. Updating the inverse of a matrix. *Society for Industrial and Applied Mathematics (SIAM)*, 31(2):221–239, 1989.

[Hastie and Tibshirani 1998] T. Hastie and R. Tibshirani. Classification by Pairwise Coupling. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems 10*. MIT Press, Cambridge, MA, USA, 1998. ISBN 0-262-10076-2.

[Hastie et al. 2001] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[Hermansky 1990] H. Hermansky. Perceptual Linear Predictive (PLP) Analysis Of Speech. *Journal of the Acoustic Society of America*, 87:1738–1752, 1990.

[Hershey and Olsen 2007] J. R. Hershey and P. A. Olsen. Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages 317–320, 2007.

[Higgins et al. 1991] A. Higgins, L. Bahler, and J. Porter. Speaker Verification Using Randomized Phrase Prompting. *Digital Signal Processing*, 1(2):89–106, 1991.

[Hocking 1976] R. R. Hocking. The Analysis and Selection of Variables in Linear Regression. *Biometrics*, 32:1–49, 1976.

[Hoerl and Kennard 1970] A. Hoerl and R. Kennard. Ridge Regression: Biased Estimation for NonOrthogonal Problems. *Technometrics*, 12:55–67, 1970.

[Jaakkola and Haussler 1999a] T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 487–493, Cambridge, MA, USA, 1999a. MIT Press.

[Jaakkola and Haussler 1999b] T. S. Jaakkola and D. Haussler. Probabilistic Kernel Regression Models. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics (AISTASTS)*, 1999b.

[Jain et al. 2004] A. K. Jain, A. Ross, and S. Prabhakar. An Introduction to Biometric Recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):4–20, 2004.

[Katz et al. 2005] M. Katz, M. Schafföner, E. Andelic, S. E. Krüger, and A. Wendemuth. Sparse Kernel Logistic Regression for Phoneme Classification. In *Proceedings of the 10th International Conference "Speech and Computer" (SPECOM)*, pages 523–526, 2005. ISBN 5-7452-0110-x.

[Katz et al. 2006a] M. Katz, S. E. Krüger, M. Schafföner, E. Andelic, and A. Wendemuth. Speaker Identification and Verification Using Support Vector Machines and Sparse Kernel Logistic Regression. In *Advances in Machine Vision, Image Processing, and Pattern Analysis, IWICPAS 2006, Proceedings*, volume 4153 of *Lecture Notes in Computer Science*, pages 176–184, Berlin/Heidelberg, 2006a. Springer. ISBN 3-540-37597-X.

[Katz et al. 2006b] M. Katz, M. Schafföner, E. Andelic, S. E. Krüger, and A. Wendemuth. The IESK-Magdeburg Speaker Detection System for the NIST Speaker Recognition Evaluation. In *Proceedings of NIST Speaker Recognition Evaluation*, 2006b.

[Katz et al. 2006c] M. Katz, M. Schafföner, E. Andelic, S. E. Krüger, and A. Wendemuth. Sparse Kernel Logistic Regression using Incremental Feature Selection for Text-Independent Speaker Identification. In *IEEE Odyssey 2006 — The Speaker and Language Recognition Workshop, Proceedings*, pages 1–6, 2006c. ISBN 1-4244-0472-X.

[Keerthi et al. 2005] S. S. Keerthi, K. Duan, S. K. Shevade, and A. N. Poo. A Fast Dual Algorithm for Kernel Logistic Regression. *Machine Learning*, 61:151–165, 2005.

[Kenny et al. 2005] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel. Factor Analysis Simplified. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 637–640, 2005.

[Kimeldorf and Wahba 1971] G. S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematics Analysis and Applications*, 33:82–95, 1971.

[Krüger et al. 2005] S. E. Krüger, M. Schafföner, M. Katz, E. Andelic, and A. Wendemuth. Speech Recognition with Support Vector Machines in a Hybrid System. In *Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, pages 993–996. ISCA, 2005.

[Kung et al. 2004] S.-Y. Kung, M.-W. Mak, and S.-H. Lin. *Biometric Authentication. A Machine Learning and Neural Network Approach.* Prentice Hall Information and System Sciences Series. Prentice Hall, 2004. ISBN 0-131-47824-9.

[Levinson 1947] N. Levinson. The Wiener RMS Error Criterion in Filter Design and Prediction. *Jour. Math. Phys.*, 25:261–278, 1947.

[Li and Porter 1988] K.-P. Li and J. Porter. Normalizations and Selection of Speech Segments for Speaker Recognition Scoring. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 595–598, 1988.

[Magrin-Chagnolleau and Durou 1999] I. Magrin-Chagnolleau and G. Durou. Time-Frequency Principal Components Of Speech: Application To Speaker Identification. In *6th European Conference on Speech Communication and Technology (EUROSPEECH)*, pages 759–762, 1999.

[Magrin-Chagnolleau and Durou 2000] I. Magrin-Chagnolleau and G. Durou. Application of Vector Filtering to Pattern Recognition. In *15th International Conference on Pattern Recognition (ICPR)*, volume 3, pages 433–436, 2000.

[Martin et al. 1997] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET Curve in Assessment of Detection Task Performance. *Proceedings of the 5th European Conference on Speech Communication and Technology (EUROSPEECH)*, 4:1895–1898, 1997.

[Martin et al. 2004] A. Martin, D. Miller, M. Przybocki, J. Campbell, and H. Nakasone. Conversational Telephone Speech Corpus Collection for the NIST Speaker Recognition Evaluation 2004. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 587–590, 2004.

[Matejka et al. 2007] P. Matejka, L. Burget, P. Schwarz, O. Glembek, M. Karafiat, J. Cernocky, D. van Leeuwen, N. Brümmer, A. Strasheim, and F. Grezl. STBU System for the NIST 2006 Speaker Recognition Evaluation. In *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007.

[Melin and Lindberg 1996] H. Melin and J. Lindberg. Guidelines for Experiments on the Polycost Database. In *Proceedings of a COST 250 workshop on Application of Speaker Recognition Techniques in Telephony*, pages 59–69, Vigo, Spain, 1996.

[Mercer 1909] J. Mercer. Functions of Positive and Negative Type and their Connection with the Theory of Integral Equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 209:415–446, 1909.

[Mika et al. 2001] S. Mika, A. Smola, and B. Schölkopf. An Improved Training Algorithm for Kernel Fisher Discriminants. In T. Jaakkola and T. Richardson, editors, *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics (AISTATS)*, pages 98–104, San Francisco, CA, USA, 2001. Society for Artificial Intelligence and Statistics, Morgan Kaufmann.

[Miller 2002] A. Miller. *Subset Selection in Regression*. CRC Monographs on Statistics & Applied Probability, 2002. ISBN 1-584-88171-2.

[Minka 2003] T. P. Minka. A Comparison Of Numerical Optimizers for Logistic Regression. Technical report, Microsoft Research, 2003. URL `http://research.microsoft.com/~minka/papers/logreg/`.

[Nabney 1999] I. Nabney. Efficient training of RBF networks for classification. In *Artificial Neural Networks – ICANN 1999*, volume 1, pages 210–215, 1999.

[Nordström et al. 1998] T. Nordström, H. Melin, and J. Lindberg. A Comparative Study of Speaker Verification Systems using the Polycost Database. In *5th International Conference on Spoken Language Processing (ICSLP)*, pages 1359–1362, 1998.

[Peterson and Barney 1952] G. Peterson and H. Barney. Control Methods Used in a Study of the Vowels. *The Journal of the Acoustical Society of America*, 24:175–184, 1952.

[Platt 2000] J. C. Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In P. Bartlett, B. Schölkopf, D. Schuurmans, and A. Smola, editors, *Advances in Large-Margin Classifiers*, pages 61–74. MIT Press, Cambridge, MA, USA, oct 2000. ISBN 0-262-19448-1. URL `http://research.microsoft.com/~jplatt/abstracts/SVprob.html`.

[Price et al. 1995] D. Price, S. Knerr, L. Personnaz, and G. Dreyfus. Pairwise Neural Network Classifiers with Probabilistic Outputs. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 1109–1116. MIT Press, Cambridge, MA, USA, 7 1995. ISBN 0-262-20104-6.

[Przybocki and Martin 2006] M. Przybocki and A. Martin. The NIST Year 2006 Speaker Recognition Evaluation Plan. *NIST*, 2006. URL `http://www.nist.gov/speech/tests/spk/2006/`.

[Przybocki and Martin 2004] M. Przybocki and A. Martin. NIST Speaker Recognition Evaluation Chronicles. In *Proceedings of ODYSSEY - The Speaker and Language Recognition Workshop*, 2004.

[Rabiner and Juang 1993] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall Signal Processing Series. Prentice Hall, Englewood Cliffs, NJ, USA, 1993. ISBN 0-13-015157-2.

[Reynolds 1995] D. Reynolds. Speaker Identification and Verification Using Gaussian Mixture Speaker Models. *Speech Communication*, 17:91–108, 1995. ISSN 0167-6393.

[Reynolds 2003] D. Reynolds. Channel Robust Speaker Verification via Feature Mapping. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 53–56, 2003.

[Reynolds et al. 2000] D. Reynolds, T. Quatieri, and R. Dunn. Speaker Verification Using Adapted Gaussian Mixture Models. *Digital Signal Processing*, 10:19–41, 2000.

[Robinson 1989] A. Robinson. *Dynamic Error Propagation Networks.* PhD thesis, Cambridge University Engineering Department, Cambridge, UK, 1989.

[Rosenblatt 1958] F. Rosenblatt. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, 65(6):386–408, 1958.

[Roth 2004] V. Roth. The Generalized LASSO. *IEEE Transactions on Neural Networks*, 15(1):16–28, 2004.

[Schafföner et al. 2006] M. Schafföner, E. Andelic, M. Katz, S. E. Krüger, and A. Wendemuth. Limited Training Data Robust Speech Recognition using Kernel-Based Acoustic Models. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 1137–1140, 2006. ISBN 1-4244-0469-X.

[Schölkopf 1997] B. Schölkopf. *Support Vector Learning.* PhD thesis, Technische Universität Berlin, 1997.

[Shawe-Taylor and Cristianni 2004] J. Shawe-Taylor and N. Cristianni. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004. ISBN 0-52181-397-2.

[Shriberg and Ferrer 2007] E. Shriberg and L. Ferrer. A Text-Constrained Prosodic System for Speaker Verification. In *Interspeech*, pages 1226–1229, 2007.

[Smith and Gales 2002] N. Smith and M. Gales. Speech Recognition Using SVMs. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.

[Smola and Schölkopf 2000] A. J. Smola and B. Schölkopf. Sparse Greedy Matrix Approximation for Machine Learning. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pages 911–918, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-707-2.

[Snir et al. 1998] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra. *MPI – The Complete Reference*, volume 1. MIT Press, Cambridge, MA, 2nd edition, 1998. ISBN 0-262-69215-5.

[Solomonoff et al. 2005] A. Solomonoff, W. Campbell, and I. Boardman. Advances In Channel Compensation For SVM Speaker Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 629–632, 2005.

[Sterling et al. 1999] T. L. Sterling, J. Salmon, D. J. Becker, and D. F. Savarese. *How to build a Beowulf — A Guide to the Implementation and Application of PC Clusters*. Scientific and Engineering Computing. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-69218-X.

[Stolcke et al. 2007] A. Stolcke, E. Shriberg, L. Ferrer, S. Kajarekar, K. Sonmez, and G. Tur. Speech Recognition as Feature Extraction for Speaker Recognition. In *Proceedings of the IEEE Workshop on Signal Processing Applications for Public Security and Forensics*, pages 39–43, 2007.

[Stroustrup 2000] B. Stroustrup. *Die C++-Programmiersprache*. Addison-Wesley, München, 2000. ISBN 3-8273-1660-X.

[Stuhlsatz et al. 2006] A. Stuhlsatz, H.-G. Meier, M. Katz, S. E. Krüger, and A. Wendemuth. Support Vector Machines for Postprocessing of Speech Recognition Hypotheses. In *Proceedings of International Conference on Telecommunications and Multimedia (TEMU)*, 2006.

[The MathWorks 1997] The MathWorks. *Using MATLAB*, 1997. URL `http://www.mathworks.com`.

[Tibshirani 1996] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.

[Tsuda et al. 2002] K. Tsuda, M. Kawanabe, G. Rätsch, S. Sonnenburg, and K.-R. Müller. A New Discriminative Kernel From Probabilistic Models. *Neural Computation*, 14(10):2397–2414, oct 2002. ISSN 0899-7667.

[Vapnik 2000] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Information Science and Statistics. Springer, Berlin, 2nd edition, 2000. ISBN 0-387-98780-0.

[Vapnik 1998] V. N. Vapnik. *Statistical learning theory.* Adaptive and learning systems for signal processing, communications, and control. John Wiley & Sons, Inc., New York, NY, 1998. ISBN 0-471-03003-1.

[Wendemuth 2004a] A. Wendemuth. *Grundlagen der Stochastischen Sprachverarbeitung.* Oldenbourg, München, 2004a. ISBN 3-486-27465-1.

[Wendemuth 2004b] A. Wendemuth. *Grundlagen der Digitalen Signalverarbeitung.* Springer, 2004b. ISBN 3-540-21885-8.

[Wendemuth 1994] A. Wendemuth. Training of Optimal Cluster Separation Networks. *Journal of Physics A*, 27:L387–L390, 1994.

[Wendemuth 1995a] A. Wendemuth. Performance of Robust Training Algorithms for Neural Networks. *Journal of Physics A*, 28:5485–5493, 1995a.

[Wendemuth 1995b] A. Wendemuth. Learning the Unlearnable. *Journal of Physics A*, 28:5423–5436, 1995b.

[Zadrozny and Elkan 2002] B. Zadrozny and C. Elkan. Transforming Classifier Scores Into Accurate Multiclass Probability Estimates. In *International Conference on Knowledge Discovery and Data Mining*, 2002.

[Zhu and Hastie 2005] J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14:185–205, 2005.

[Zou and Hastie 2005] H. Zou and T. Hastie. Regularization and Variable Selection Via the Elastic Net. *Journal of the Royal Statistical Society*, 67 (2):301–320, 2005.