Wearable Sound-Based Recognition of Daily Life Activities, Locations and Conversations

A dissertation submitted to

ETH ZURICH

for the degree of

Doctor of Sciences

presented by

MIRCO ROSSI

Dipl. El.-Ing., ETH Zürich
born October 1, 1982
citizen of Brusino Arsizio TI, Switzerland

accepted on the recommendation of

Prof. Dr. Gerhard Tröster, examiner
Prof. Dr. Albrecht Schmidt, co-examiner

2014
Mirco Rossi
Wearable Sound-Based Recognition of Daily Life Activities, Locations and Conversations
Diss. ETH No. 21553

First edition 2014
Published by ETH Zurich, Switzerland

Printed by Reprozentrале ETH

Copyright © 2014 by Mirco Rossi
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior permission of the author.
To my family
Acknowledgments

First of all, I would like to thank my advisor, Prof. Dr. Gerhard Tröster, for giving me the opportunity to write this dissertation at the Wearable Computing Lab. I thank him for his support and for providing me with excellent research facilities. Special thanks go to Prof. Dr. Albrecht Schmidt for co-examining my PhD dissertation.

I am particularly grateful to Dr. Oliver Amft for the valuable discussions, inputs and advices during my Master’s thesis and PhD studies. He helped shaping my research ideas and with whom sharing of authorship in many papers reflects the closeness of our cooperation.

To all members of the IfE, thank you for the great environment you provided. Especially I would like to thank my office mates and all who worked together with me on the SENSEI project: Amir, Andreas, Bernd, Christina, Claudia, Clemens and Sebastian. Thank you also to Alberto, Bert, Burcu, Christian, Daniel R., Daniel W., Franz, Giovanni, Julia, Lars, Long-Van, Luisa, Martin W., Michael, Niko, Rolf, Silvan, Simon, Sinziana, Thomas H., Thomas K., Thomas S., Tobias, Ulf and Zack. A special thanks to Ruth Zähringer and Fredy Mettler for their help with technical and administrative problems.

I also owe thanks to all the students that worked with me on the project and contributed to its results, namely, Nils Braune, Seraina Buchmeier, Jeremy Constantin, Jan Haller, Adrian Hess, Jonas Huber, Sandro Martis, Christian Käslin and Benedikt Köppel.

Outside my research, I would like to thank all my sports mates who shared the passion for BodyAttack and regularly practiced sport with me. A special thank goes to Davide-Dan Margiotta for sharing the positive attitude and for the mental support.

Finally, I want to thank the most important people, my family. Especially I wish to thank my parents Doris and Ivan for always supporting me in every aspect and for making it possible to fulfill my dreams in my life.

Zurich, January 2014

Mirco Rossi
Contents

Acknowledgments v

Abstract xi

Zusammenfassung xiii

1. Introduction 1
   1.1 Context Awareness and Recognition 2
   1.2 Sound as a Context Provider 3
   1.3 Speaker Sensing 4
   1.4 Ambient Sound Sensing 7
   1.5 Research Contributions 10
   1.6 Approach of the Thesis 11
   1.7 Thesis Outline 14

2. Speaker Identification for Wearable Systems 19
   2.1 Introduction 20
   2.2 Related Work 20
   2.3 Unsupervised Speaker Identification System 22
   2.4 System Evaluation 25
   2.5 System Deployment on DSP 30
   2.6 Conclusion and Future Work 34

3. Collaborative Speaker Identification 37
   3.1 Introduction 38
   3.2 Collaborative Speaker Identification Concept 39
   3.3 Collaborative Identification Algorithms 43
   3.4 Evaluation Dataset 48
   3.5 Results 49
   3.6 Discussion 54
   3.7 Conclusion and Future Work 57

4. Speaker Identification in Daily Life 59
   4.1 Introduction 60
   4.2 Related Work 60
   4.3 Architecture 61
4.4 Visualization .................................................. 64
4.5 Evaluation ..................................................... 65
4.6 Conclusion .................................................... 72

5. Activity and Location Recognition for Smartphones 73
5.1 Introduction .................................................. 74
5.2 Related Work ................................................ 74
5.3 AmbientSense Architecture ............................... 75
5.4 Implementation ............................................. 77
5.5 Evaluation .................................................... 79
5.6 Conclusion .................................................... 85

6. Indoor Positioning System for Smartphones 87
6.1 Introduction .................................................. 88
6.2 Related Work ................................................ 88
6.3 RoomSense Architecture ................................. 90
6.4 Evaluation Study .......................................... 94
6.5 Evaluation .................................................... 97
6.6 RoomSense Implementation ............................. 102
6.7 Conclusion and Future Work ......................... 103

7. Crowd-Sourcing for Daily Life Context Modelling 105
7.1 Introduction .................................................. 106
7.2 Related Work ................................................ 106
7.3 Concept of Web-Based Sound Modelling ............. 107
7.4 Results ........................................................ 110
7.5 Conclusion .................................................... 113

8. Daily Life Context Diarization using Audio Community Tags 115
8.1 Introduction .................................................. 116
8.2 Related Work ................................................ 117
8.3 Context Diarization based on Community Tags .... 118
8.4 Daily Life Evaluation Study ............................ 124
8.5 Evaluation Approach ...................................... 126
8.6 Results ........................................................ 132
8.7 Discussion ..................................................... 138
8.8 Conclusion .................................................... 139
9. Conclusion and Outlook 141
  9.1 Conclusion ........................................ 142
  9.2 Outlook ............................................ 143

Glossary 147

Bibliography 149

Curriculum Vitae 163
Abstract

Driven by advances in mobile computing, sensing technologies, and signal processing, context-awareness emerged to a key research area. Context-aware systems targets to automatically sense the user context and based on this information, support the user in his daily life. Context is typically sensed by sensors attached at the user’s body or in his environment. In wearable context recognition motion modalities have been traditionally used to recognize patterns in user’s movements, postures, or gestures. Improvements in recognition accuracy have been achieved by including additional modalities such as location, interaction with persons, or object use. However motion based systems are limited in recognizing user’s context where distinct movement patterns are present.

Sound is a rich source of information that can be used to improve user context inference. Almost every user’s activity (e.g. teeth brushing) and location (e.g. restaurant) produces some distinct sound patterns. Also social interactions of a person can be revealed by speech and voice patterns in the auditory channel. Additionally, compared to other modalities like motion, sound offers some advantages for wearable context recognition, e.g. the robustness against sensor displacements and the user-independency.

In this thesis we envisioned a personal wearable sound based recognition system which provides continuous real-time context information of the user throughout his day. We investigated in wearable context recognition systems focusing on two categories of context: speaker sensing and ambient sound sensing. For both categories we designed and implemented wearable pattern recognition systems, we proposed techniques to increase recognition performance and reduce the process of manually collecting training data, and we analysed the performance of our prototypes in daily life environments.

In speaker sensing we investigated in an unsupervised speaker identification system. The system identifies known speakers and detects unknown speakers using pattern recognition based on the extraction of well-known audio features (e.g. Mel Frequency Cepstrum Coefficients). If an unknown speaker is detected, speech data is used to dynamically enrol a new speaker model for future identification. For speaker modelling Gaussian Mixtures Model and Vector Quantization was tested.
Our evaluation indicated a recognition performance in meeting rooms conditions of up to 81% recognition rate for 24 speakers. Besides the standalone mode of the speaker identification system a collaboration mode was proposed where two or more identification systems can build an ad-hoc collaboration to exchange speaker information. Evaluation showed that with four collaborating systems recognition rate increases compared to the standalone mode for up to 9% for 4 speakers and up to 21% for 24 speakers. Additionally, an evaluation study was performed where we tested the system on full-day recordings of person’s working days. Results showed identification accuracies in daily life situations from 60% to 84%, depending on the background noise during conversation.

In ambient sound sensing we proposed an activity and location recognition system for smartphones, running in two operating modes: autonomous mode where the system autonomously runs on the smartphone, and server mode where recognition is done in combination with a server. For 23 ambient sound classes, the recognition accuracy of the system was 58.45%. Analysis of runtime showed similar results for both modes (e.g. 12 h of continuous runtime), whereas prediction time was for the autonomous mode in average 200 ms faster. Furthermore, we investigated in two crowd-sourcing methods to model user’s context based on web collected audio data and tags. One method provides the opportunity to reduce the process of manually collecting training data. Core component of our system was the proposed outlier filters, to remove outlier samples from the collected audio data. In an evaluation study of participant’s full-day recordings recognition rates for 23 high-level contexts between 51% and 80% could be achieved. The proposed outlier filtering methods yielded a recognition accuracy increase of up to 18%. The second crowd-sourcing method uses a new approach to generate a descriptive daily life diary using crowd-sourced audio tags. Audio models of over 600 tags were used to describe person’s daily life audio recordings. In a user study with 16 participants our system produced daily life diaries with up to 75% meaningful tags. Finally, we investigated in a smartphone-based method for indoor positioning using an active sound fingerprinting approach. An active acoustic fingerprint of a room or a position in a room was derived by emitting a sound chirp and measuring the impulse response. Based on the fingerprint, pattern recognition was used to recognize the indoor-position. Our system achieved excellent recognition performance (accuracy > 98%) for localize a position in a set of 20 rooms.
Zusammenfassung


Umgebungsgeräusche beinhalten Informationen, die verwendet werden können, um die Erkennung von Kontext zu verbessern. Fast jede Aktivität (z. B. Zähne putzen) und jeder Ort (z. B. Restaurant) produzieren ein charakteristisches Geräuscht muster. Auch soziale Interaktionen können, basierend auf Sprache und Sprachmuster, durch die Audiomodalität erkannt werden.

Das Ziel dieser Arbeit war ein tragbares audiobasiertes Erkennungssystem für den persönlichen Gebrauch, das kontinuierlich und in Echtzeit Kontext des Benutzers während seines Alltags erkennt. Wir konzentrierten uns auf zwei Kategorien von Kontext: Sprechererkennung (Speaker Sensing) und Umgebungsgeräuscherkennung (Ambient Sound Sensing).

Im Bereich Sprechererkennung entwickelten wir ein Sprecher-identifikationssystem, welches selbständig neue Sprecher erlernt. Das System identifiziert bekannte Sprecher und detektiert unbekannte Sprecher mittels Mustererkennung. Die vorhandenen Sprechdaten eines unbekannten Sprechers werden benutzt, um dynamisch ein neues Sprechermodell zu erstellen, welches dann für die künftige Identifikation benutzt wird. Unsere Auswertung ergab eine Erkennungsrate von bis zu 81% für 24 Sprecher in Konversationen mit typischen Geräuschkampanien von Tagungsräumen. Neben dem autonom laufenden Sprecheridentifikationssystem wurde ein Kollaborationsmodus vorgeschlagen, bei welchem zwei oder mehr Identifikationssysteme eine ad-hoc Zusam-
menarbeit zum Austausch von Informationen der Sprecher aufbauen können. Unsere Auswertungen zeigten, dass mit vier kollaborierenden Erkennungssystemen die Erkennungsrate gegenüber dem autonom laufenden System bis zu 9% für 4 Sprecher und bis zu 21% für 24 Sprecher gesteigert werden kann. Zusätzlich wurde eine Evaluationsstudie durchgeführt, in welcher das System mit ganztägigen Audioaufnahmen von 6 Probanden getestet wurde. Die Ergebnisse zeigten Identifikationsraten in Alltagssituationen von 60% bis 84%.

In this chapter sound-based context recognition for wearable systems is motivated and the challenges are presented. Furthermore, the research objectives of this dissertation are summarized and the approach to design, implement, and evaluate wearable sound-based context recognition systems is explained. The chapter finishes with an outline of the thesis.
1.1 Context Awareness and Recognition

In the last decade, mobile systems such as smartphones have overtaken traditional desktop computers. Especially smartphones are changing the usage of personal computing. In combination with advances in signal processing and sensing technology context-aware applications have arisen. A context-aware application aims to sense user’s personal and environmental factors and to use this information to automatically adjust its configuration and functionality [1]. Typical examples for context awareness are smartphone apps helping the user to find local points of interests by using the device positioning system. Besides location, user’s activity is one of the most important contextual cues [2]. However the typical definitions of context are broad and concrete interpretations may depend on the application.

Context recognition is the crucial building block to enable context awareness. The goal of context recognition is to infer user context by observing a person’s action and her/his environment in daily life situations. To achieve this goal, context recognition systems use sensors placed on the user’s body and in her/his environment. Different sensor modalities can be used for recognition including motion (e.g. accelerometer, gyroscope), location (e.g. GPS), sound (e.g. microphone), and video (e.g. camera). As an example, a video camera installed in a room can be used by computer vision to recognize user’s actions [3].

Stationary systems, however, are limited to one location and thus are not able to capture user’s context throughout the day. In contrast, wearable systems enable continuous context recognition during daily life by using on-body sensors. In wearable computing, context recognition research widely covered inertial sensors, i.e. accelerometers, gyroscopes to recognize body movement [4], postures [5], or gestures [6]. User’s motion is captured by sensors attached on the body [7] or by using the smartphone’s internal sensors [8]. Improvements in recognition accuracy have been achieved by including additional modalities such as location [9], interaction with persons [10] or object use [11].

There are, however, limitations to current wearable context recognition systems. Inertial sensors are limited in recognizing user context where distinct user’s movement patterns are present. Additionally, movement patterns are usually user dependent and thus a motion-based
recognition system would need to be personalised. Furthermore, many recognition systems need multiple sensor units attached to different parts of the body, which can be impractical for daily usage.

1.2 Sound as a Context Provider

Sound is a rich source of information that can be used to make accurate inferences of a person’s context in her/his daily live. Almost every activity of a person produces some characteristic sound pattern, e.g. walking, eating or using the computer. Most locations have usually a specific sound pattern, too, e.g. restaurants or streets. Furthermore, social interactions of a person can be revealed by speech and voice patterns in the auditory channel. It has been shown that pattern recognition can be used on sound data to automatically infer user’s context focusing on the user’s activities [12], locations [13, 14], and on speakers and conversations [15]. However, the sound modality has its limitation, too. As an example, for obvious reasons user context which does not provide any acoustical information cannot be recognized.

Compared to other modalities like motion, sound offers some advantages for wearable context recognition. A person’s environmental sound can be captured and recognized with a single microphone, modalities like motion usually need more than one sensor to recognize complex activities [7]. Microphones are cheap and already available in almost any wearable device. Compared to other sensors like accelerometers, gyroscopes, or cameras, microphones are not only integrated in high-end smartphones, but also in more basic wearable devices like PDAs and cell-phones. Moreover, recognition performance of sound based context recognition systems is less dependent on the sensor’s position and more robust to sensor displacement compared to other modalities: vision-based context recognition using the smartphone’s camera is not possible when the smartphone is in user’s pocket. In motion-based recognition systems small on-body displacements of the acceleration sensors can render the recognition systems useless [16]. However, the information available from an audio stream degrades more gracefully [17]: ambient sound can still be recognized based on the captured sound of a smartphone in the pocket of the user. Another advantage of sound-based recognition is that many user activities and locations can be better detected with sound compared to other modalities. Examples are locations with background noises (e.g. street, restaurant) or activities in
which machines are involved (e.g. coffee machine, washing machine). In this context distinct sound patterns are given and sound-based recognition can outperform other modalities. User independent operation can be considered as another advantage of sound-based context recognition. Motion based context recognition is usually related to the user. Since a user can perform an activity in a variety of ways, activity recognition systems based on motion sensors need to take this into account. Sound-based context recognition is based on sound which is generally dominated by environmental sound sources and not by sounds generated by the user himself and thus, recognition tends to be user-independent.

Sound can be used to infer a broad spectrum of user’s context. In this thesis two sound-based context recognition approaches have been investigated: speaker sensing and ambient sound sensing. In the following sections (Sec. 1.3 and 1.4), we introduce the approaches, present the state of the art, and detail problems and opportunities which were tackled in this thesis.

1.3 Speaker Sensing

Speaker sensing targets to recognize speaker related information. Different types of speaker information can be inferred by sound, ranging from speech detection, sex identification, language recognition, speech recognition, speaker identification and verification [18], to mood detection of a speaker [19].

In speaker sensing we focused on text-independent speaker identification targeted for use in mobile systems. Speaker identification is the task of recognizing a speaker (e.g. using an unique speaker id) by his voice. In text-independent speaker identification, speakers are identified based on an arbitrary speech segment. In contrast, text-dependent identification is performed on a pre-defined sentence which has to be uttered by the speaker. The identification of speakers in meetings and conversations is a valuable user context information which opens opportunities to analyse user’s social relations, capture interesting moments in daily life, and can provide personal annotations of timing and content of conversations.
1.3. Speaker Sensing

1.3.1 State of the Art

Automated speaker identification has been investigated from both application and technical perspectives for several years (e.g. [20, 10, 15]). These systems are either stationary installed in rooms to annotate meetings or - as it is aimed at in this work - the system can be worn as a daily personal accessory. The latter case allows us to identify interaction partners, annotate conversations, and build a personal diary of social activities.

Several smart meeting rooms have been proposed, such as at Dalle Molle Institute [20] and at Berkeley [21]. Both rooms were equipped with a set of microphones, typically a microphone array at the table centre and individual microphones worn by each participant (e.g. headset or lapel microphone). To identify a speaking person, the lapel microphone having the highest input signal energy was chosen. The approaches are by far not restricted to monitoring using acoustic means alone. Approaches have been made to combine sensor information from multiple sources, including vision and audio [22, 23]. As the systems are stationary their use is restricted to meetings and conversations held in the particular room. Wearable systems can capture conversations as they happen outside of these smart spaces.

Several procedures intended for speaker identification have been developed. Anliker [15] proposed an online speaker separation and tracking system based on blind source separation. The task of identifying speakers is facilitated by source separation, for which reason it had been used in many works. However, at least two microphones are required to perform a source separation. This property imposes extended processing and power consumption requirements, which contradict to the viability of a wearable system implementation. Other algorithms that operate without speaker separation and, therefore, need one microphone only, have been proposed by Charlet [24], Lu and Zhang [25], Kwon and Narayanan [26], and Lilt and Kubala [27]. In the last year, advances of computational power integrated in smartphones opened the door for real-time sound processing locally on the mobile device. In parallel to this thesis, smartphone-based solution for speaker recognition was proposed by Lu et al. [28].
1.3.2 Speaker Sensing in Daily Life Situations: Problems and Opportunities

Our literature review revealed that speaker identification based on sound is feasible, however, challenges remain in identifying speakers in daily life situations: identification systems found in literature are either restricted to distinct rooms or are not designed for daily life usage. A personal wearable speaker identification system requires to cope with a number of problems that affect the identification performance. The system has to be able to detect and learn new speakers as conversations may involve new co-workers, friends, or strangers. The wide variety of scenarios where a personal identification system can be used renders a general system solution challenging. For example, identification systems could be used by team workers, where members of a conversation are rather static and thus known in advance. Alternatively, systems may be used in an ad-hoc meeting with strangers, who do not use an identification system. There, speakers must be learned before they can be identified. In this section we describe problems in identifying speakers in daily life situations and present new concepts which are further investigated in this thesis.

**Unsupervised Speaker Identification:** Sound-based speaker identification enables us to identify speakers without any further infrastructure (e.g. [24, 25, 26]). However, proposed systems used pre-trained speaker models to recognize speakers. Thus, all speakers which should be identified had to be manually trained before the recognition system is started. This is not feasible for real-life applications targeting to recognize conversation partners of a person during his daily life. An idea to overcome this problem is proposed by Lu et al. [28]. They investigated in acquiring training data from phone calls and in a semi-supervised segmentation method for training speaker models based on one-to-one conversations. However, this methods imply that before identification is started, every conversation partner has to talk with the user at least one time at the phone or within a one-to-one conversation. One approach to solve this problem is to use an unsupervised identification method: using new speaker detection, speakers unknown by the identification system are detected and dynamically enrolled by training a new speaker model which is then used by the system for future identification.
Collaboration in Speaker Identification: Performance of a wearable identification system can be limited if no additional information on the use condition is available. Using collaboration between two or more personal speaker identification systems is an opportunity to improve the identification in many scenarios. In collaboration mode wearable systems exchanges information of the current speaker or available speaker model to improve standalone identification system performance. For example, personal speaker identification systems start with a speaker model for their owner only. When jointly exposed in a meeting, they perform weakly in identifying other participants and in acquiring further speakers from the conversation. However, in this collaborative scope, relevant speakers are known already by each individual system, which provide a crucial benefit for all participants.

1.4 Ambient Sound Sensing

Ambient sound sensing targets to recognize user’s context which is recognizable by sound patterns produced in the user’s environment. This context can be divided in user activities and locations. ”Brushing teeth” or ”showering” are examples of user activities with distinct sound pattern. Especially activities involving a machine are hearable and recognizable by sound (e.g. ”making coffee” with a coffee machine). Many types of locations have distinct sound patterns, too, e.g. ”street” or ”restaurants”.

Additionally to the recognition of user activities and locations, we investigated in indoor positioning based on ambient sound. The aim of sound-based indoor positioning is to localize a person in an indoor environment based the person’s ambient sound. Indoor positioning can be done on different granularities, e.g. localizing in which room the person is or localizing a person within a room.

1.4.1 State of the Art

In the last decade several publications addressed the recognition of ambient sounds (e.g. [29, 30, 17]). In general these works focused on recognizing locations. Starting from 2004 works on sound-based recognition of activities of daily living (ADL) came up. Sound-based ADL recognitions have been investigated for different locations like bathroom [12],
Chapter 1: Introduction

office [31, 32], kitchen [31, 33], workshop [31, 34], and household for health care applications [35, 36, 37]. Smartphone based ambient sound recognition systems were published by Miluzzo et al. [38], and Lu et al. [39].

In the field of indoor positioning various approaches exist. Methods to localize wearable devices were proposed which use additional infrastructure in the environment such as sensors or transmitters [40, 41, 42]. Alternative approaches without the need of dedicated infrastructure are using already existing wireless infrastructure such as cellular network and Wi-Fi information for the localization task [43, 44, 45]. However, these positioning approach is less suitable where station coverage is unknown or sparse. Recently sound-based positioning approaches have been proposed that require no additional infrastructure to perform indoor positioning [46, 47, 48].

1.4.2 Ambient Sound Sensing in Daily Life Situations: Problems and Opportunities

In the last section we showed the potential of using ambient sound sensing to infer user context. However, in daily life situations ambient sound sensing remains a challenge: the presented recognition system are either constraint to small sets of sound contexts and well-defined recording locations, or not providing context recognition in real-time. In this section we describe problems of ambient sound sensing in daily life situations and present opportunities which we investigated in this thesis.

Crowd-Sourcing for Context Modelling: For wearable ambient sound recognition systems, identifying complex daily life situations, such as office work or commuting is challenging due to variations in the acoustic context and limited availability of training data. Consequently, most existing recognition systems suffer from incomplete modelling of the user contexts. While appropriate training data is essential for ambient sound recognition systems, it is laborious to obtain sufficient amounts with annotations that represent these daily life situations. Sound data can be sourced from the web to derive acoustic pattern models of daily life context. This approach is inspired by the idea of crowd-sourcing [49]: Web audio data is generated by the web community. Web audio data is heterogeneous, available in large quantities, and provides annotations, e.g. in the form of ‘tags’.
Daily Life Context Diarization using Audio Community Tags: Context recognition systems are usually limited to recognize a small set of context categories (e.g. 24 sound categories [13]), usually defined by the researcher or a small group of persons. However, sound includes information which can be used to infer context with different point of views. As an example, from ambient sound of a street we could infer that the user is near a street. However, we could also infer other events like "approaching car", "bus", "footsteps" or "singing birds". Another crowd-sourced inspired method can be used to overcome this limitation. Based on an open-source web audio community a daily life context diary using crowd-sourced audio tags can be generated. Audio models of a large set of tags (> 100) can be used to automatically describe person’s daily life audio recordings.

Indoor-Positioning using Active Sound Probing: Indoor positioning is still an actively researched field. Recently sound-based positioning approaches have been proposed. Passive sound fingerprinting uses ambient sound to generate position estimates, whereas active fingerprinting approaches emit and then record a specific sound pattern for the positioning. Wirz et al. [46] proposed an approach to estimate the relative distance between two devices by comparing ambient sound fingerprints passively recorded from the devices’ positions. The distance was classified in one of the three distance regions (0 m, 0 m – 12 m, 12 m – 48 m) with an accuracy of 80%. However, no absolute position information was obtained by this method. Tarzia et al. [48] proposed a method based on passive sound fingerprinting by analysing the acoustic background spectrum of rooms to distinguish different locations. The location was determined by comparing the measured sound fingerprint for a room with fingerprints from a database. A room’s fingerprint was created by recording continuous ambient sound of 10 s length. The evaluation of this method showed a room recognition accuracy of 69% for 33 rooms, however, room recognition dropped when people were chatting or when the background spectrum had large variations. Azizyan et al. [47] used a combination of smartphone sensor data (WiFi, microphone, accelerometer, colour and light) to distinguish between logical locations (e.g. McDonalds, Starbucks). Their passive acoustic fingerprints are generated by recording continuous ambient sound of 1 min and extracting loudness features. In contrast, an active acoustic fingerprint can be derived by emitting a sound chirp and measuring the impulse responses. We expect an active sound fingerprinting approach
to reduce recognition time compared to the passive approach and to be robust against noise. However, to our knowledge yet no approach exist using active sound fingerprinting.

1.5 Research Contributions

The goal of this thesis was to design, implement and evaluate a personal wearable sound-based recognition system for daily life activities, locations and conversations. The work includes the following contributions:

- **Sound based pattern recognition for wearable systems:** We designed and implemented wearable recognition systems for both speaker and ambient sound sensing. In speaker sensing we proposed an unsupervised speaker identification system using unknown speaker detection and online speaker enrolment. In ambient sound sensing we proposed a system recognizing user’s activities and locations using real-time cloud computing, and a system for the recognition user’s indoor position using active sound fingerprinting. Recognition systems have been implemented as a wearable prototype for real-time performance testing.

- **Exploiting collaboration and crowd-sourcing:** We studied the use of collaboration and crowd-sourcing to increase recognition performance and reduce the process of manually collecting and annotating training data. In speaker sensing we exploited collaboration between two or more wearable speaker identification systems to improve identification accuracy. In ambient sound sensing we used the concept of crowd-sourcing by exploiting audio data of an open-source web audio community enabling automatic collection of training data.

- **Evaluation studies in daily life environments:** We evaluated our recognition prototypes in daily life environments. In speaker sensing we evaluated our identification system in distinct real-life situations (e.g. in a bar or near a street) and in full-day experiments during users’ daily life routines. In ambient sound sensing our crowd-sourced recognition system was used to generate a descriptive daily life diary of user’s full-day ambient sound.
1.6 Approach of the Thesis

This section describes the approach we used in this thesis to design, implement, and evaluate our wearable sound-based context recognition systems. Figure 1.1 depicts the wearable sensing and inference system architecture used throughout this thesis.

![Wearable Sensing and Inference System Architecture](image)

**Figure 1.1:** Overview of the wearable sensing and inference system architecture used throughout this thesis.

Conceptually, our proposed recognition systems consisted of the following components:

**Active Probing and Sensing:** These components are responsible for acquiring the audio data. The sensing component directly samples the audio data either from the device’s integrated microphone or an external microphone (e.g. a headset). Typically for speaker and ambient sound sensing audio is sampled at 8kHz or 16kHz at 16 bit depth (e.g. [13, 28]). In contrast to the sensing component, active probing uses additional to the microphone the device’s integrated loudspeaker. In this thesis, we used active probing for indoor positioning. By emitting a sound chirp the room impulse response was measured (see Chapter 6 for details).

**Front-End Processing:** targets to extract audio features which characterize either a speaker or an ambient sound category. Using feature extraction size of sensor data is reduced by discharging information irrelevant for recognition. A considerable work on the design of acoustic features for sound classification can be found in the literature (a review can be found in [50]). Typical feature sets include both time- and frequency-domain features, such as spectral skewness, energy envelope, harmocity and pitch. The most used audio features in audio classifica-
tion are the Mel cepstral frequency coefficients (MFCC). Throughout the thesis we used MFCC feature extraction and compared the performance with other audio feature sets. In Section 7 were a data volume of more than 700 GB was modelled, extracted features were further processed to audio fingerprints for additional data reduction (see Section 8.3.1).

**Modelling and Recognition:** Modelling and recognition uses machine learning algorithms to do pattern matching. Context categories are represented by a model trained to detect certain patterns in the data. Training models is usually done in an offline phase prior recognition. In this case only the recognition algorithm has to work in real-time on the wearable device. However, in an unsupervised setting new context categories are detected and learned online (e.g. as in our unsupervised speaker identification system presented in Chapter 2) and thus both modelling and recognition is done in real-time on the wearable device. The most used technique for both speaker and ambient sound modelling is the Gaussian Mixture Model (GMM) (e.g. [51, 13]). In this thesis we used GMM to model both speakers and ambient sounds and compared the performance with two other techniques also used in audio modelling: Vector Quantization (VQ) (e.g. [52]) and Support Vector Machine (SVM) (e.g. [53]).

**Performance Evaluation:** The recognition performance of our recognition systems was evaluated using an annotated sound dataset. Depending on the recognition system freely available or self-collected sound datasets of speakers or ambient sounds were used. In general we measured recognition performance by the normalized accuracy $acc = mean(acc_c)$, which is the mean of all class relative accuracies:

$$acc_c = \frac{TP_c + TN_c}{N_c}$$

$TP_c$ is the number of true positive of class $c$, $TN_c$ the number of true negatives of class $c$, and $N_c$ is the number of all test instances of class $c$. The set of classes $c \in C$ depends on the evaluated recognition system and was either a set of speaker identities or set of ambient sound classes. The normalized accuracy is further referred just as accuracy ($acc$).
1.6. Approach of the Thesis

1.6.1 Wearable Platforms for Prototyping

As introduced in Section 1.5 all our designed recognition systems have been implemented as a wearable prototype for real-time performance testing or for real life evaluation studies. Figure 1.2 shows the wearable platforms we used for our prototype implementations. Our first prototype was the real-time speaker identification system implemented in 2007 (presented in Chapter 2). For this prototype we used the ColdCruncher DSP board (see Figure 1.2(a)). ColdCruncher is a custom system, including the TI TMS320C67 DSP, audio interface, USB host connection, and power supply to attach a battery. Moreover, the system included 64 MB SDRAM memory and 16 MB flash. The system was designed to wear it attached to a belt. Our further prototypes have been implemented and tested on Android smartphones: the enhanced speaker identification system (see Chapter 4), the activity and location recognition system (see Chapter 5), and the indoor positioning system (see Chapter 6). For evaluation of these prototypes we used two Android smartphone models: the Google Nexus One (see Figure 1.2(b)) and the Samsung Galaxy SII (see Figure 1.2(c)).

![Figure 1.2:](image)

Table 5.2 shows the specifications of the used wearable devices. The devices are sorted by their release date and reflect the hardware advances achieved in wearable systems over the last years.
<table>
<thead>
<tr>
<th></th>
<th>ColdCruncher</th>
<th>Google Nexus One</th>
<th>Samsung Galaxy SII</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU</strong></td>
<td>TI TMS320C67 DSP</td>
<td>1 GHz ARM Cortex A8</td>
<td>1.2 GHz Dual-core ARM Cortex-A9</td>
</tr>
<tr>
<td><strong>RAM</strong></td>
<td>64 MB</td>
<td>512 MB</td>
<td>1024 MB</td>
</tr>
<tr>
<td><strong>Battery</strong></td>
<td>external</td>
<td>1400 mAh Lithium-ion</td>
<td>1650 mAh Lithium-ion</td>
</tr>
<tr>
<td><strong>Wi-Fi</strong></td>
<td>none</td>
<td>IEEE802.11g, 54Mbit/s</td>
<td>IEEE802.11n, 300Mbit/s</td>
</tr>
<tr>
<td><strong>Dimensions</strong></td>
<td>40x55x22</td>
<td>119x60x12</td>
<td>125x66x8</td>
</tr>
<tr>
<td><strong>Release Date</strong></td>
<td>2006</td>
<td>2010</td>
<td>2011</td>
</tr>
</tbody>
</table>

**Table 1.1:** Specifications of the used wearable platforms

### 1.7 Thesis Outline

This thesis is based on seven scientific publications. Figure 1.3 depicts the thesis research contributions (as described in Section 1.5) and the chapters that cover them. Table 1.2 maps these chapters to the related publications.

![Thesis Outline Diagram](attachment:thesis-outline.png)

**Figure 1.3:** Outline of this thesis according to the research contributions presented in Section 1.5 and the definition of the terms *speaker sensing* and *ambient sound sensing* detailed in Section 1.3 and 1.4.
Contributions focusing on speaker sensing are presented first, followed by contributions in ambient sound sensing. In detail, the work is organized as follows:

- Chapter 2 presents our unsupervised speaker identification system. The system’s architecture, the prototypical wearable DSP implementation and the evaluation of the system are detailed.

- Chapter 3 introduces collaborations between multiple personal speaker identification systems. Besides the standalone mode introduced in Chapter 2, a collaboration mode is presented. A generalized description of collaboration situations are presented and evaluated.

- Chapter 4 introduces MyConverse, a personal conversation recognizer and visualizer for Android smartphones. MyConverse is based on the speaker identification system presented in Chapter 2 including improvements in speaker recognition and modelling. A user study is presented, to show the capability of MyConverse in daily life situations to recognize and display user’s communication patterns.

- Chapter 5 presents AmbientSense, a personal activity and location recognition system implemented as an Android app. AmbientSense is implemented in two modes: in autonomous mode the system is executed solely on the smartphone, whereas in the server mode the recognition is done using cloud computing. Both modes are evaluated and compared concerning recognition accuracy, runtime, CPU usage, and recognition time.

- Chapter 6 presents RoomSense, a new method for indoor positioning using active sound fingerprinting. RoomSense is implemented as an Android app works autonomously on a smartphone. An evaluation study was conducted to analyse the localization performance of RoomSense.

- Chapter 7 presents an approach to model daily life contexts from crowd-sourced audio data. Crowd-sourced audio tags related to individual sound samples were used in a configurable recognition system to model 23 ambient sound context categories.

- Chapter 8 introduces a daily life context diarization system which is based on audio data and tags from a community-maintained
audio database. We recognised and described acoustic scenes using a vocabulary of more than 600 individual tags. Furthermore, we present our daily life evaluation study conducted to evaluate the descriptiveness and intelligibility of our context diarization system.

- Chapter 9 concludes the work by summarizing and discussing the achievements.
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Publication</th>
</tr>
</thead>
</table>
| 2       | Collaborative Real-Time Speaker Identification for Wearable Systems  
M. Rossi, O. Amft, M. Kusserow, and G. Tröster  
| 3       | Collaborative Personal Speaker Identification: A Generalized Approach Pervasive and Mobile Computing  
M. Rossi, O. Amft, and G. Tröster  
| 4       | MyConverse: Personal Conversation Recognizer and Visualizer for Smartphones  
M. Rossi, O. Amft, S. Feese, C. Käs lin, and G. Tröster  
| 5       | AmbientSense: A Real - Time Ambient Sound Recognition System for Smartphones  
M. Rossi, S. Feese, O. Amft, N. Braune, S. Martis, and G. Tröster  
| 6       | RoomSense: An Indoor Positioning System for Smartphones using Active Sound Probing  
M. Rossi, J. Seiter, O. Amft, S. Buchmeier, and G. Tröster  
In Proceedings of the 4th Augmented Human International Conference, 2013 |
| 7       | Recognizing Daily Life Context using Web-Collected Audio Data  
M. Rossi, O. Amft, and G. Tröster  
| 8       | Recognizing and Describing Daily Life Context using Crowd-Sourced Audio Data and Tags  
M. Rossi, O. Amft, and G. Tröster  
Submitted to Pervasive and Mobile Computing, July 2013 |

Table 1.2: Chapters and publications presented in this thesis.
This chapter presents our unsupervised speaker identification system for personal annotations of conversations and meetings. The system’s architecture and the prototypical wearable DSP implementation are detailed. We evaluated our system on the freely available 24-speaker Augmented Multiparty Interaction dataset. For 5s recognition time, the system achieves 81% recognition rate. Additionally, the prototypical wearable DSP implementation could continuously operate for more than 8 hours from a 4.1 Ah battery.

*This chapter is based on the following publication:
M. Rossi, O. Amft, M. Kusserow, and G. Tröster. Collaborative Real-Time Speaker Identification for Wearable Systems. (PerCom 2010) [54]
2.1 Introduction

Identifying a speaker during meetings or conversations allows to annotate communication, determine social relations, capture interesting moments in daily life. Moreover, it can enable many further applications, such as recognising speech and analysing interactions.

In this chapter we present the design and implementation of our unsupervised speaker identification for wearable systems. In our approach, a speaker is modelled dynamically from voice data and subsequently identified. In particular the following contributions are presented:

1. We present an unsupervised, text-independent speaker identification system using only one microphone. We study its performance using a freely available dataset that had not been investigated for speaker identification before. The results demonstrate that our system can cope even with large meeting sizes of 24 speakers.

2. We discuss the real-time system design with regard to constraints in wearable systems. To this end, we present and evaluate a complete implementation and deployment on a DSP platform prototype, show that the system can operate in real-time, and a wearable identification system can be built.

2.2 Related Work

A literature review of sound-based text-independent speaker identification systems was presented in Section 1.3.1. In this section we detail related work focusing on wearable systems enabling to monitor speech behaviour of individual persons.

An initial wearable system was the Sociometer developed by Choudhury and Pentland in 2002 [10]. This system can be attached to a person’s shoulder. It includes an IR transmitter and receiver to communicate with persons nearby. A microphone was used to separate speech segments from non-speech segments. The Sociometer is used for different kinds of social network analysis and organizational behaviour, including analysis of social behaviour in a research group [55], modelling of group discussion dynamics [56], and prediction of shopper’s interest [57]. As the speaker identification with the Sociometer is achieved through IR communication, only individuals wearing this system can be recognized.
The works cited above impressively demonstrate the broad application potential of speaker identification. Nevertheless, these systems are limited by the prior knowledge and configuration required to operate them, such as the number and identity of speakers, and their location. Since those approaches did not use a speaker modelling, the monitoring devices depend essentially on exchanging information on the current speaker. However, the availability of speaker models would allow to use identification system while roaming between locations and continuously identifying speakers that have been modelled before. Subsequently, adding the capability to detect a new speaker allows to learn speakers dynamically and unsupervised.

Several procedures intended for unsupervised speaker recognition have been developed. Anliker [15] proposed an online speaker separation and tracking system based on blind source separation. The task of identifying speakers is largely facilitated by source separation, for which reason it had been used in many works [58, 59, 15]. However, at least two microphones are required to perform a source separation. This property imposes extended processing and power consumption requirements, which contradict to the viability of a wearable system implementation.

Other algorithms that operate without speaker separation and, therefore, need one microphone only, have been proposed by Charlet [24], Lu and Zhang [25], Kwon and Narayanan [26], and Lilt and Kubala [27]. These works utilized different speech features including linear predictive cepstrum coefficients (LPCC), mel-frequency cepstrum coefficients (MFCC), and line spectrum pair (LSP). For modelling speaker these systems typically use Gaussian Mixture Models (GMMs). It is known that GMMs may not be stably derived from small training data sizes. For unsupervised operation during conversations, however, only small data amounts may be available to learn a new speaker online. In contrast, Vector Quantization (VQ) handles small training data sizes more effectively [52, 60]. Although these presented systems were based on one microphone only, no investigation in wearable and real-time speaker identification has been done.
2.3 Unsupervised Speaker Identification System

Our unsupervised speaker identification system incorporates two operations: recognition and online learning. For recognition, the system identifies a speaker by matching phoneme models from a speaker database to the continuous audio stream. In addition, the system can identify new speakers that do not sufficiently match with the existing model database. These new speakers are automatically added to the system using online learning. Figure 2.1 illustrates main components of our identification system for both operations.

Figure 2.1: Concept of unsupervised speaker identification system supporting recognition and online learning operations.

Rejecting an utterance that does not belong to any existing speaker model in the database, is a core design element of an unsupervised speaker identification system. We study here three variants for discriminating between known and unknown speakers that yield different identification performance.

2.3.1 Front-End Audio Processing

Front-end processing targets to extract speaker-dependent and text-independent features from the audio signal using pre-processing, feature extraction, and channel compensation.

Most speaker related information in speech is inside a frequency band of 0-4 kHz. To minimize system complexity, we chose an 8 kHz sampling and 16 bit quantization rate. During pre-processing we filtered the raw audio signal with a transfer function \( H(z) = 1 - \alpha z^{-1} \), where \( \alpha = 0.97 \). This filter emphasizes higher frequencies bands and removes speaker independent glottal effects [51].

Subsequent to pre-processing, a feature vector \( \mathbf{x} = (x_1, \ldots, x_N) \) was derived from the audio signal. Two frequently used audio feature sets
have been evaluated: linear predictive cepstrum coefficients (LPCC) and mel-frequency cepstrum coefficients (MFCC). Both concepts capture phonetic speaker properties. Since phonemes are speech segments of about 20-30 ms [51] we used a sliding window with 30 ms length and 20 ms step size to derive these features. Different numbers of coefficients \((N)\) have been evaluated for both feature sets (see Section 2.4.2).

We utilized a linear channel compensation approach to minimize device-dependent effects. We used here the short-term cepstral mean subtraction [51]. However, we applied it on sliding windows: \(\tilde{x}^t = x^t - \bar{x}^t\), with \(\bar{x}^t = \frac{1}{T} \sum_{j=t-T}^{t} x^j\). This corresponds to subtracting the feature vector average of the last \(T\) features from feature vector \(x^t\) generated at time \(t\). \(T\) was set to the recognition epoch size, as described below.

### 2.3.2 Speaker Modelling and Matching

During recognition mode, feature vectors of a speech segment are compared with stored database models to identify a known speaker. While GMMs can outperform VQ in text-independent speaker recognition performance [61], they require a complex model learning phase using the expectation maximization algorithm. However, VQ can outperform GMMs at small amounts of training data and when fast modelling time is required [62]. With regard to our real-time speaker recognition and learning system, we chose VQ as a short training time was desired. In addition, algorithm complexity is a critical concern for the DSP implementation.

With VQ, speaker models are formed by clustering a set of training feature vectors \(\{x_i\}_{i=1}^{L}\) in \(K\) non-overlapping clusters. Each cluster is represented by a code vector \(c_i\) of the cluster centroid. A set of code vectors (codebook) \(C = \{c_i\}_{i=1}^{K}\) serves as speaker model during recognition.

Several clustering algorithms can be used to derive a codebook, however, with marginal performance differences [63]. For this work we used the Generalized Lloyd algorithm (GLA) [64], which has low complexity compared to the other known algorithms. The modelling procedure parameters (codebook size \(K\), number of feature vectors \(L\)) determine system complexity. These have been further evaluated in Section 2.4.

To identify a speaker during recognition we used the quantization distortion between a set of test feature vectors \(X = \{x_i\}_{i=1}^{M}\) and a speaker codebook \(C\). The quantization distortion \(d_q\) of \(x_i\) with respect to \(C\) was defined as \(d_q(x_i, C) = \min_{c_j \in C} d(x_i, c_i)\). Here \(d(x_i, c_i)\) is a dis-
tance measure defined between two feature vectors for which we used the Euclidean distance. The average of all individual distortions was used as matching metric of a speaker model during recognition (Equation 2.1).

\[ D(X, C) = \frac{1}{M} \sum_{i=1}^{M} d_q(x_i, C) \] (2.1)

Speaker identification is done by calculating the mean distortion of every code of every codebook stored in the system’s database. The speaker is then identified with the best matching speaker model \( C_{\text{best}} \), which is the codebook with the smallest \( D \).

The recognition performance is proportional to the length of a recognition epoch, hence, the number of feature vectors \( M \) considered for each recognition. Nevertheless, long epochs will prevent the system to identify rapid speaker changes in conversation and meetings. We evaluate \( M \) in Section 2.4.

### 2.3.3 New Speaker Detection

In unsupervised open-set operation, a speaker may be initially unknown to the system. Consequently, we developed a procedure that determines whether the analysed observation belongs to a known or unknown speaker. For this purpose we defined a decision function shown in Equation 2.2.

\[ f_d(X, C_{\text{best}}) = \begin{cases} 1, & \text{if } \text{score}(X, C_{\text{best}}) \geq \Delta \\ 0, & \text{else} \end{cases} \] (2.2)

\( X \) is the set of feature vectors of the tested person, \( C_{\text{best}} \) is the best matching speaker model, \( \text{score}(X, C_{\text{best}}) \) is a score function, and \( \Delta \) is a threshold. If the score of a tested speaker is equal or larger than \( \Delta \), the tested speaker is classified as the best matching speaker \( C_{\text{best}} \). However, if \( \text{score}(X, C_{\text{best}}) \) is smaller than \( \Delta \), the observation will be classified as unknown speaker.

We analysed three variants for the score function, one of these is the impostor cohort normalization (ICN) [65, 66]. The two alternatives were developed for this work and compared to ICN in Section 2.4.

1. The score function corresponds to the negated best matching speaker model distortion (compare Equation 2.1):

\[ \text{score}_{\text{score}}(X, C_{\text{best}}) = -D(X, C_{\text{best}}) \] (2.3)

where \( C_{\text{best}} \) is the model of speaker \( C_{\text{best}} \). 
2.4 System Evaluation

2. The score function corresponds to the negated $D(X, C)$, normalized by distortions of a set of other speaker models ("impostor speakers"):

$$\text{score}_{ICN}(X, C_{\text{best}}) = -\frac{D(X, C_{\text{best}}) - \mu_I}{\sigma_I},$$  \hspace{1cm} (2.4)

with mean $\mu_I$ and standard derivation $\sigma_I$ of the impostor distortions. This score function corresponds to the impostor cohort normalization (ICN).

3. The score function corresponds to the feature vectors in $X$ with minimum distance to the best matching speaker model $C_{\text{best}}$ ($N_{\text{win}}$), normalized by the total number of feature vectors in $X$ ($N_{\text{all}}$):

$$\text{score}_{\text{win}}(X, C_{\text{best}}) = \frac{N_{\text{win}}}{N_{\text{all}}}. \hspace{1cm} (2.5)$$

**Online learning procedure**

When an unknown speaker is detected, as described in Section 2.3.3 above, this new speaker is enrolled in the system using online learning. All feature vectors that have been collected during recognition and new speaker detection are reused to derive the new speaker model.

For a real-time operation, timing constraints exist between recognition, new speaker detection, and online learning. Since an identified new speaker is instantly enrolled, the new speaker detection epoch was set to equal the training set size. Figure 2.2 illustrates these timing relations. The current speaker is recognized in every recognition epoch of length $t_{\text{rec}} = 5\text{sec}$ (see Section 2.3.2), whereas new speakers are detected on a sliding window of length $t_{\text{NSD}} = t_{\text{train}} = 20\text{sec}$ and a window shift of one recognition epoch. If the current speaker is classified as known, a speaker ID is returned every $t_{\text{rec}} = 5\text{sec}$, whereas if the current speaker is classified as unknown, online learning of a new model is triggered.

2.4 System Evaluation

To confirm the robust system operation and to select parameters for efficient online performance, we evaluated the system performance. In particular, we analysed system performance for LPCC and MFCC with different coefficient counts, the number of centroids to model speakers, the effect of training and recognition times, and the three score metrics
Figure 2.2: Illustration of timing relations between recognition, new speaker detection, and online learning in the real-time system (see Figure 2.1).

for our new speaker detection. These parameters influence complexity and performance of the resulting system regarding both online learning and recognition, and hence determine viability of real-time operation on a DSP system.

2.4.1 AMI speaker corpus and evaluation procedure

To ensure reproducibility of all analysis results, we selected the freely available Augmented Multiparty Interaction (AMI) corpus [67] for our evaluation. This dataset provides more than 200 individual English speakers and contains $\sim$100 hours of conversation/meeting scenes recorded from ambient far-field microphones and close-talk lapel microphones, worn by each participant. Each meeting had four participants. Two meeting types were recorded and transcribed: actual ad-hoc meetings and scenario-based meetings, where people had been briefed to talk about a particular topic beforehand.

For the standalone system performance analysis, we extracted speech data from the original corpus to evaluate performance for a set of 24 speakers (9 female, 15 male). From each speaker 5 minutes of speech out of two different meetings were used and annotated with a speaker ID. We used audio data recorded from all individual lapel microphones for this purpose\(^1\). As the audio files were originally recorded with 16 kHz, we resampled it to 8 kHz. An anti-aliasing FIR filtering was performed prior to downsampling. A two-fold cross-validation was applied to partition the speech data into training and evaluation set.

\(^1\)Described as lapel mix at the AMI website.
2.4.2 LPCC/MFCC Vector Dimension

We analysed the performance of LPCC and MFCC performance on the AMI dataset. For speaker enrolment a training time of 20 s was applied and the number of centroids per model was set to $K=16$. In recognition mode an epoch time of 5 s was used.

Figure 2.3: System performance for LPCC and MFCC modelling with different feature vector dimensions (cepstrum coefficients).

Figure 2.3 shows recognition performance of LPCC and MFCC algorithms for different feature vector sizes $N$ (number of coefficients). We observed that both modelling approaches yield similarly good results. Increasing the number of coefficients, increases recognition accuracy. For more than 12 coefficients, however, performance only marginally increased. Consequently, lower cepstral coefficients carry most of the speaker individuality. These results for the AMI dataset confirm earlier performance reports [68, 69].

As the MFCC algorithm uses FFT, its complexity is larger than that of LPCC [70]. Since the performance of both methods was similar, we used LPCC in further analysis steps and set the number of coefficients to $N=12$. 
2.4.3 VQ Codebook Dimension

Figure 2.4 shows the performance with regard to the codebook size $K$ per speaker model. We observed that accuracy improves with more centroids per model, however, with more than 16 centroids, performance increases only marginally. Nevertheless, recognition complexity using the VQ method depends linearly on the number of centroids. For further analysis we set $K = 16$.

![Figure 2.4: System performance with regard to the codebook size $K$ (numbers of VQ centroids).](image)

2.4.4 Training and Recognition Time

Due to the unsupervised online operation both learning and recognition must be performed with in size-constrained data. We analysed the number of feature vectors needed to train (parameter $L$) and recognize a speaker ($M$). Figure 2.5 shows the system performance with regard to training and recognition time. The results confirm that below 5 s of recognition time system performance decreases rapidly. In contrast, only marginal improvements are obtained for more than 6 s of recognition time. With 10 s of training data per speaker, recognition accuracy was below 50%, while >70 s did not further improve performance.

As it is desirable to recognize a speaker in short speech segments, recognition time must be short. In addition, there is potentially only lit-
2.4. System Evaluation

selected parameter set

Figure 2.5: System performance trade-off with regard to training and recognition time. The selected parameter set (marked point) corresponds to an accuracy of 0.81.

Figure 2.5: System performance trade-off with regard to training and recognition time. The selected parameter set (marked point) corresponds to an accuracy of 0.81.

2.4.5 Score Functions for New Speaker Detection

Initially unknown speakers are detected by the system using a score function as described in Section 2.3.3. Using the system parameters chosen before, we compared the ICN score ($\text{score}_{\text{ICN}}()$) to both alternatives. Figure 2.6 shows the result for all three score functions using Receiver Operating Characteristic (ROC) analysis and 20 s training time. The area under the curve (AUC) was used to compare the score functions performance. $\text{score}_D$ yields a low performance (AUC=0.71) compared to ICN with AUC=0.91. The best result was obtained for $\text{score}_{\text{win}}$, with AUC=0.94.

For our real-time system implementation it is important to avoid recalculating the best matching model that results from the 5 s recognition epochs for the total 20 s time frame. Hence, we applied a majority voting over the four frames obtained from recognition. With this scheme, AUC drops from 0.94 to 0.93 for $\text{score}_{\text{win}}$. In return, complexity of the algorithm is greatly reduced.
Figure 2.6: ROC performance for the new speaker detection using three score functions. Results for the classification of known and unknown speakers are shown. The curves were derived by varying a decision threshold on the score functions results. AUC measures the area under the curve.

2.5 System Deployment on DSP

We implemented the speaker identification system on a wearable DSP system using Matlab-Simulink\(^2\) with the goal to identify speakers in standalone operation, as described in Section 2.3.

While the performance results derived in Section 2.4 were obtained in simulations using a desktop workstation, the results in this section refer to the actual wearable DSP system implementation. The Matlab algorithms and Simulink models remained the same for both evaluations. Hence, the recognition performance results presented above are valid for the DSP system as well.

The computational performance analysis presented in this section is based on run-time tests executed on our DSP system. Theoretical complexity analysis of the LPCC and VQ algorithms have been detailed in other works [70].

\(^2\)Matlab-Simulink: http://www.mathworks.ch/products/simulink/
2.5. System Deployment on DSP

2.5.1 Implementation using Matlab-Simulink

The identification system was implemented in Matlab-Simulink. Pre-defined Simulink blocks from the library “Signal Processing Blockset” were used to facilitate system design. These included operations, such as “Autocorrelation LPC” and “LPC to Cepstral Coefficients”. The designed solution was subsequently evaluated on a desktop workstation as presented above.

In a second step, DSP-specific interface blocks were added to the design. We used the library “Target for TI C6000” to generate executable code for the DSP from Simulink. Audio signal inputs, LEDs, Switches, memory operations, and special routines for the DSP board were controlled by blocks included in the library.

Simulink uses “Matlab Real-Time Workshop” to generate C code supported by the DSP platform. This code is then transfered to the development application (Code Composer Studio for the TMS320 DSP processor family from Texas Instruments), to build an executable for the intended DSP processor. The Real-Time Workshop build process loads the specified machine code to the board and runs the executable file on the DSP system. The hardware evaluation was performed using a Texas Instruments TMS320C6713 DSP clocked at 225 MHz with 16 MB of memory.

Nevertheless, we had to optimize the automatically generated code so that a sufficient processing performance of the system was achieved. The changes involved implementing an additional DSP routine as special block in Simulink.

2.5.2 System Configuration

We selected a parameter set according to the evaluation results in Section 2.4, such that a trade-off between processing performance and recognition performance is achieved.

The speaker identification system was modelled with Simulink as a multirate system: every 20 ms a new 12-LPCC feature vector is extracted, every 5 s a speaker recognition is performed. The new speaker detection is done on a 20 s frame every 5 s. The decision is based on score\(_{\text{win}}\) with threshold \(TH_{\text{win}} = 0.107\). If a speaker is classified as unknown a new 16-VQ speaker model is created based on the 20 s decision frame. Simulink separated these rates in three synchronous, periodically scheduled tasks with fixed priorities. The task with smallest period has
the highest, whereas the task with the longest period has the lowest priority.

2.5.3 Optimization of the Implementation

The automatically generated code was further optimized manually to achieve optimal system performance. In particular, we used the dedicated DSP function for calculating the squared sum of vector elements, according to \( \text{vecsumsquared}(v) = \sum_i v(i)^2 \). This function permits an efficient processing of the squared Euclidean distance, while Simulink did not provide a predefined block for this purpose. The optimized code was imported as S-function to the Simulink design to avoid manual changes after Simulink code generation. Performance improvement due to optimization is discussed in the next section and summarized in Table 2.1.

2.5.4 Processing Performance Analysis

We analysed real-time processing performance of the implementation on the DSP system and compared this result to the host workstation. Using the implementation generated by Simulink without optimization, as detailed above resulted in a online learning time of 25 s and real-time recognition with up to 4 speakers in the system’s speaker model database without concurrent learning. Processing more than 4 speaker models was not possible resulting in a buffer overflow. These results are insufficient for the targeted real-time operation. Our analysis revealed that processing of the Euclidean distance was the limiting element. For every feature vector of 12 elements the distance to 16 centroids had to be determined, where 50 feature vectors were derived per second. This results in 9600 squaring operations per second.

Using the code optimization approach, clearly improved processing performance. The DSP system was able to identify speakers in real-time with up to 150 speakers in the system’s speaker model database. We derived this result by virtually increasing the number of speaker models in the system’s database. Furthermore, deriving a new speaker model for online learning required 5 s. The online learning could be initiated only after the 20 s of data have arrived. Hence in total 25 s were needed to enrol a new speaker until it could be identified from the database.

For comparison we evaluated the performance for a Intel Pentium 4, 3 GHz system. For this system a maximum of 70 speakers could be recognized in real-time. Table 2.1 summarizes the results.
Table 2.1: Processing performance of the implemented system.

<table>
<thead>
<tr>
<th>System</th>
<th>DSP</th>
<th>DSP</th>
<th>PC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TI TMS320C67</td>
<td>TI TMS320C67</td>
<td>Intel Pentium 4</td>
</tr>
<tr>
<td></td>
<td>unoptimized</td>
<td>optimized</td>
<td>(3 GHz)</td>
</tr>
<tr>
<td>Speakers in</td>
<td>≤ 4</td>
<td>≤ 150</td>
<td>≤ 70</td>
</tr>
<tr>
<td>recognition</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learning time</td>
<td>25 s</td>
<td>5 s</td>
<td>16 s</td>
</tr>
</tbody>
</table>

2.5.5 Wearable DSP Prototype Device

We analysed the integration of the DSP system into a wearable device prototype to confirm the viability of our approach towards a personal speaker identification. For this purpose we designed a custom system, including the TI TMS320C67 DSP, audio interface, USB host connection, and power supply to attach a battery. Moreover, the system included 64 MB SDRAM memory and 16 MB flash. The system was designed to wear it attached to a belt. An external battery could be attached to the device.

The design was split into a main board, containing the DSP and memories, and a interface board, containing power supply, and interfaces. Both board can be stacked to minimize the design size. Figure 2.7 shows both boards. In stacked format the system had an outline of 55x40x22 mm. In our initial investigation we used an existing battery design that provided 4.1 Ah capacity.

We performed an initial power consumption analysis of this system with a power supply of 3.3 V and a DSP clock of 197 MHz. When capturing audio at 8 kHz the system consumed 928 mW. When the system executed additional processing algorithms in addition to audio capturing, and stored results to flash memory, consumption raised to 976 mW. However, when capturing audio at 48 kHz, 996 mW were required even without further processing. The latter result indicates that audio capture has an impact on consumption. Hence, processing two audio streams, as it would be required for source separation, or processing higher sampling rates increase the power consumption challenge for a wearable system. At standby the system consumed 308 mW. We expect that this standby consumption can be reduced by optimizing the power supply of the analysed design.
These consumption results cannot be compared to audio processing systems aiming at ultra low-power operation of 0.1W, such as the SoundButton [31]. In contrast, the design implemented here targets rapid prototyping, e.g. using the Matlab-Simulink toolchain. This concept allows to process complex algorithms such as the unsupervised speaker identification demonstrated in this work. Nevertheless, even at this current power consumption rate, the device had a measured battery operation time of 8.6 hours between recharges. This will be a sufficient runtime to further study the personal speaker annotation in various applications.

### 2.6 Conclusion and Future Work

In this chapter we presented our unsupervised real-time speaker identification approach intended for a personal wearable annotation system. The system provides recognition and an online learning functions that operate in parallel to identify speakers from a model database, detect unknown speakers, and enrol new speakers.

We evaluated our design decisions regarding the real-time implementation on the freely available AMI dataset that has not been used for speaker recognition before. Our results indicated an excellent perfor-
mance of up to 81% recognition rate for 24 speakers and a recognition time of 5 s.

Finally, we reported implementation results from deploying our speaker identification approach to a wearable DSP system using Matlab-Simulink. With manual optimizations, the implementation was able to process up to 150 speaker models on a DSP in real-time. Learning time for enrolling a new speaker was 5 s. Including the lead-time for new speaker detection, an unknown speaker could be enrolled (from initial voice samples to model in the database) within 25 s. Our subsequent evaluation of a wearable implementation prototype showed that the system could continuously operate for >8 hours using a 4.1 Ah battery. These results combined with the excellent recognition performance confirm the viability of our speaker identification approach on a wearable device.

We assumed in this work that the analysed audio data contains speech information only. We expect that a robust voice activity detection (VAD) can be added to the system to perform an a-priori speech segmentation. In chapter 4 we present a daily-life evaluation of our system extended with a voice activity detector and with additional improvements in speaker modelling and recognition.

While the system can operate robustly with the selected training time, a faster enrolment may be desirable. For this purpose the Generalized Lloyd algorithm (GLA, see Section 2.3.2) would need to be replaced by another clustering approach that permits an incremental model creation. A weaker model could then serve to recognize the speaker during the first few seconds already.
This chapter introduces collaborations between multiple personal speaker identification systems to improve annotation performance. Besides the standalone mode of the speaker identification system introduced in Chapter 2 a collaboration mode is presented. A generalized description of collaboration situations are presented and three use scenarios are derived. Further, an evaluation of the system in these use scenarios is presented.

*This chapter is based on the following publication:
Chapter 3: Collaborative Speaker Identification

3.1 Introduction

In the last chapter we showed how a wearable speaker identification system could be realized that supports real-time operation. We confirmed that a speaker identification could be efficiently performed on a wearable system. Ad-hoc collaboration between two or more wearable systems could help in many scenarios to improve standalone identification system performance. For example, personal systems could start with a speaker model for their owner only. When jointly exposed in a meeting, they would perform weakly in identifying other participants and in acquiring further speakers from the conversation. However, in this collaborative scope, relevant speakers are known already by each individual system, which could provide a crucial benefit for all participants.

In this chapter, we present a generalized approach to personal speaker identification that is independent of particular locations and can benefit from collaborative settings, in which multiple distributed systems share their recognition results. While our system can be used in standalone operation as introduced in Chapter 2, we foresee that systems exchange information to jointly recognize speakers and to decide whether a speaker is known to the collective. As our system concept supports learning of new speakers, collaboration is used as well to improve robustness for unsupervised speaker set extensions. In particular this chapter provides the following contributions:

1. We extend our speaker identification system presented in the last chapter. We show how our approach can be applied in different collaborative use scenarios, in which a personal speaker identification may be exposed. For this purpose, we introduce collaboration scenarios that account for unknown speakers and independent speaker model databases of the participating systems.

2. We study the performance of identification systems in collaborative operation, using the same corpus presented in the last chapter (see Section 2.4.1). Our results confirm clear benefits, (1) for collaboratively recognizing speakers, (2) for unsupervised systems that collaborate during identification of new speakers, and (3) for mixtures of systems collaborating and systems “knowing” conversation-relevant speakers.
3.2 Collaborative Speaker Identification Concept

The operation of a personal speaker identification system can change with the availability of collaboration partners and depends on the collaborative scenario. This section details our collaboration approach.

3.2.1 Collaborative Speaker Identification Architecture

The function of a personal speaker identification system is to continuously annotate the user’s conversations when the system is carried by the user. Speakers in conversations are recognized and their speech segments are annotated. In standalone mode, a personal identification system analyses the speech signal recorded from a worn microphone. The speaker is identified using a speaker model (Speaker recognition) and it is detected whether a current speaker is known (New speaker detection). Unknown speakers are then automatically learned by the system and stored as speaker model in a system’s database. This standalone mode does not involve collaboration with other systems at all and thus represents a baseline to study collaboration benefits.

In contrast, if two or more participants of a conversation use a personal speaker identification system, these systems could collaborate in their identification and detection tasks. In our approach systems periodically broadcast information of their current identification and detection results through an ad-hoc network, such as ZigBee [72]. An identification system can utilize information from other collaborative systems by fusing it with its own results. Thus, in collaborative mode, each system performs an individual speaker identification and new speaker detection as in standalone mode, while in addition, using information from others. To utilize information of others, a relation between the system’s speaker identity (Speaker ID) representation and that of other systems must be derived (Speaker ID mapping). Figure 3.1 illustrates the collaborative mode setting that is generally considered in this work.

In our implementation, the operation in standalone and collaborative mode can be switched at any time to ensure independence of a personal systems.

3.2.2 Collaboration Scenario Analysis and Use Cases

In collaborative mode, the state of each system’s speaker model database essentially determines its benefit for others. Here, we consider
Figure 3.1: Collaborative speaker identification architecture. Collaborating systems exchange information on speaker recognition, new speaker detection, and for speaker ID mapping. Both speakers with and without a personal speaker identification system are recognized.
all online information exchange between two or more speaker identification systems that participate in a conversation as a collaboration. In the most general case, no assumptions on the speaker model database can be made. However, as discussed below, specific collaboration applications exist, which could reduce identification uncertainty compared to this general case.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speaker model</td>
<td>A specific speaker model is used by an individual identification system to recognize speaker. Speaker models are stored in the speaker database of an identification system.</td>
</tr>
<tr>
<td>Speaker identity (Speaker ID)</td>
<td>Speaker IDs are generated by individual speaker identification systems corresponding to a speaker model. Speaker IDs are in general not compatible with IDs of other systems (see Section 3.3.3 for a description).</td>
</tr>
<tr>
<td>Relevant speakers $S_{Relevant}$</td>
<td>Denotes the speakers that actually participate in a conversation.</td>
</tr>
<tr>
<td>Local speaker set $L_n$</td>
<td>Contains speakers known by the system. A model for each of these speakers exists in the database of the identification system $n$.</td>
</tr>
<tr>
<td>Collaborative speaker set $S_{Collab}$</td>
<td>$S_{Collab}$ denotes the set of speakers known in the joint set of systems that participate in a collaboration. Thus, a model for the speaker exists in the database of at least one system in a collaboration.</td>
</tr>
</tbody>
</table>

Table 3.1: Speaker-related terminology and variables used in the collaboration scenario analysis.

Systems that hold at least their user’s speaker model in their speaker database can be of substantial benefit to others for recognizing this speaker. In contrast, systems that do not hold relevant speaker models for a conversation, cannot support the recognition step. In the worst case, a speaker is unknown by all systems in a collaboration (hence not in the speaker database of any collaborating system). This situation will be detected by each system, and corrected by learning a new model. Nevertheless, even in this situation systems can collaborate to ensure that the speaker is indeed unknown to all. Thus, in both situations, collaboration can support the operation of an individual system.
We structured the collaborative usage scenarios regarding properties of speaker sets jointly known by the systems in a collaboration and regarding the relation of speakers sets known by individual systems. Figure 3.2 provides an overview on the four different collaborative use scenarios that result from these categories. We denote the set of speakers known by a collaboration with $S_{\text{Collab}}$ and the total speaker set relevant in a collaboration with $S_{\text{Relevant}}$. Relevant speakers are speakers that actually participated in a conversation. The set of relevant speakers known by system $n$ is referenced with $L_n$, where $L_n \subseteq S_{\text{Collab}} \forall n = 1 \ldots N$, and $N$ is the total number of systems in the collaboration. These terms and variables are summarized in Table 3.1.

We refer to scenarios where all speakers are known in a collaboration, thus $S_{\text{Collab}} = S_{\text{Relevant}}$, as collaborative-closed (CC). In contrast, $S_{\text{Collab}} \neq S_{\text{Relevant}}$ describes collaborative-open (CO) scenarios. In the latter scenarios, unknown, but relevant speakers exist. For local-identical (LI)-set systems, where $L_1 = L_2 = \ldots = L_N = S_{\text{Collab}}$, all identification systems have the set of relevant speakers in their databases. In contrast, in local-nonidentical (LN)-set systems, $L_1 \neq L_2 \neq \ldots \neq L_N$, the databases do not contain the same speakers.

**CC-LI scenario.** In this scenario all system databases contain identical speaker sets and all speakers in a collaboration are known. A typical situation for this scenario is the use of identification systems by team members, where the members of a conversation are known. The scenario applies as well for meetings, where all speakers use a collaborative system that learnt all speakers.

**CC-LN scenario.** In this scenario databases contain different relevant speaker sets. The speakers are known by at least one of the collaborative systems. A typical situation occurs when identification systems start into a collaboration with a speaker model of their owner only. Thus, collectively, all relevant speakers are available.

**CO-LI scenario.** In this scenario all system databases contain identical speaker sets, but relevant speakers can be unknown to all systems. We have not found a typical application for this scenario and thus assume that it is less likely to occur in practice.

**CO-LN scenario.** In addition to the open-set speaker problem of CO-LI, the system databases contain different speakers in this scenario. This is the most challenging scenario and applies to arbitrary conversations or meetings, where either not all speakers use a collaborative system or systems have acquired relevant speakers independently before entering in the current collaboration.
3.3 Collaborative Identification Algorithms

This section details the system’s implementation of the collaborative mode operation. Figure 3.3 depicts the system architecture. Details of the components used for standalone mode (front-end processing, speaker recognition, new speaker detection, speaker modelling, and speaker model database) were presented in the last chapter in Section 2.3. This section details the algorithm implementation for collaborative speaker recognition and new speaker detection. In addition, we present a mapping method to exchange speaker IDs among collaborating systems.

### 3.3.1 Collaborative Speaker Recognition

The goal of collaborative speaker recognition is to improve recognition performance of individual identification systems (see Figure 3.3). For this purpose, systems in a collaboration exchange their locally obtained speaker recognition results at each recognition epoch $t_{rec}$ =

<table>
<thead>
<tr>
<th>Local–identical sets (LI)</th>
<th>Collaborative–closed set (CC)</th>
<th>Collaborative–open set (CO)</th>
</tr>
</thead>
<tbody>
<tr>
<td>All systems know the same speakers.</td>
<td>$L_n = S_{Collab} = S_{Relevant}$ ∀n = 1…N</td>
<td>Speakers unknown by a collaboration possible. $S_{Collab} \neq S_{Relevant}$</td>
</tr>
<tr>
<td>$L_1 = L_2 = \ldots = L_N = S_{Collab}$</td>
<td>CO-LI $L_n = S_{Collab} \forall n = 1 \ldots N$</td>
<td></td>
</tr>
</tbody>
</table>

| Local–nonidentical sets (LN) | CC-LN $L_1 \cap L_2 \cap \ldots \cap L_N = S_{Collab}$ | CO-LN $L_1 \cap L_2 \cap \ldots \cap L_N = S_{Collab}$ |
| Local speaker sets not identical. | $L_1 \neq L_2 \neq \ldots \neq L_N$ |

**Figure 3.2:** Collaboration scenarios structured regarding speaker sets jointly known by a collaboration $S_{Collab}$ compared to the relevant speaker set $S_{Relevant}$ (columns) and the relation of speakers sets known by individual systems $L_1, L_2, \ldots, L_N$ (rows).
Figure 3.3: Extended system architecture of our speaker identification system presented in the last chapter (see Figure 2.1) including collaborative mode.

More specifically, each identification system calculates the matching distances $D_{sys}^i(X, C_i)$ between speaker feature set $X$ and all models in the system’s model database $DB_{sys} = \{C_{sys}^1, \ldots, C_{sys}^{S_{sys}}\}$ (see Equation 2.1). To reduce channel differences of individual systems, the distances are normalized by subtracting the mean of all locally calculated distances $\mu_{D_{sys}} = \frac{1}{S_{sys}} \sum_{i=1}^{S_{sys}} D_{sys}^i$ for $X$:

$$DN_{sys}^i = D_{sys}^i - \mu_{D_{sys}}$$ (3.1)

This normalization is needed to create comparable distances among the systems. In addition, channel effects were minimized in the front-end processing, as described in Section 2.3.1. Each system broadcasts a set of normalized distances $\{DN_{sys}^i|i = 1, \ldots, N_{sys}\}$.

Every collaborative system sorts the shared distances according to the speaker IDs and calculates the mean distance for each speaker. A speaker is recognized as speaker ID exhibiting the smallest distance among all evaluated speakers:

$$ID_{ColRec} = \arg\min_{i \in S_{Collab}} \left( \frac{1}{N_{sys}} \sum_{sys=1}^{N_{sys}} DN_{sys}^i \right)$$ (3.2)
where \( N^{sys} \) is the number of collaborating systems. Since the collaborating systems exchanged sufficient information, the recognition can be performed by each system and the same speaker will be recognized.

Table 3.2 exemplarily illustrates the information sharing algorithm for a situation, in which a speaker with ID 3 is speaking. System 1 and 3 miss speaker models for ID 2 and 3, respectively. Overall, the collaboration recognizes speaker ID 3 as the speaker \((ID_{ColRec})\).

### 3.3.2 Collaborative New Speaker Detection

The collaborative new speaker detection aims at improving new speaker detection performance through collaboration with other systems (see Figure 3.3). In analogy to the collaborative speaker recognition, new speaker detection results of local and remote systems are fused to obtain a collective decision, if a speaker is known to the collective.

Collaborative new speaker detection is used in collaborative-open set scenarios, subsequently to a collaborative recognition (see Figure 3.3). In these scenarios both collaborative functions are performed consecutively at each recognition epoch \( t_{rec} \). In contrast, for
collaborative-closed set scenarios, all speakers are available in a collabor-
oration set already, and thus the new speaker detection is not needed.

During collaborative new speaker detection, the collaborating iden-
tification systems $sys \in Col_{sys}$ are broadcasting model scores of the
speaker $ID_{ColRec}$, which was obtained in the preceding collaborative
speaker recognition. For the model score, $score^{sys}(X, C^{sys}_{ID_{ColRec}})$, of
a system, $sys$, we used the impostor cohort normalization (ICN),
presented in the previous Chapter (see Equation 2.4). In a CO-LI sce-
nario, all models know the speaker and can share their model score.
However, in a CO-LN scenario, some collaborative systems might not
have a model for the speaker $ID_{ColRec}$ and thus will not share any
scores. Each system calculates the mean of the shared model scores:

$$score^{mean} = mean(SCORE_{shared})$$

where $SCORE_{shared} = \{score^{sys}(X, C^{sys}_{ID_{ColRec}}) | sys \in Col_{sys}\}$ is the
set of shared scores of all collaborating systems. Finally, the collective
detection is calculated by using the decision function $f$ presented in
Equation 2.2:

$$NewSpeaker_{ColDet} = f_{ICN}(X, score^{mean})$$

(3.3)

The decision threshold $\Delta$ was set to 1.57, which was obtained in the
previous chapter by maximizing the new speaker detection accuracy for
a given dataset (see Section 2.4.5). If an unknown speaker is detected, all
systems train a new model. Additionally, in a CO-LN scenario, systems
having no speaker model for speaker with $ID_{ColRec}$ train a new model
independently of the collaborative new speaker detection outcome.

### 3.3.3 Speaker ID Mapping for Collaboration

Speaker IDs are created locally by a personal identification system to
uniquely identify speaker models. Since they are generated locally on
every system, speaker IDs are not compatible between systems. For col-
laborative recognition and new speaker detection, a relation of speaker
IDs between systems of a collaboration is nevertheless needed.

Our algorithm approach targets to obtain a mapping between
speaker IDs of one system and those of another one. The algorithm
compares all speaker models of these systems to create the ID map-
ing. For this purpose, distances between two models are computed
and used as a metric denoting model similarity.
As detailed in Section 2.3.2, we model speaker phonemes using a codebook \( C = \{c_i\}_{i=1}^{K} \), which is a set of \( K \) code vectors \( c_i \). The distance between two models \( C_1 = \{c_{1i}\}_{i=1}^{K} \) and \( C_2 = \{c_{2i}\}_{i=1}^{K} \) is

\[
D(C_1, C_2) = \frac{1}{2} \left\{ \sum_{i=1}^{K} \min(c_{1i}, C_2) + \sum_{i=1}^{K} \min(C_1, c_{2i}) \right\}. \tag{3.4}
\]

Subsequently we distinguish two mapping types for local-identical (LI) and local-nonidentical (LN) speaker sets. For LI-sets, the two systems are assumed to have speaker models for the same speaker set. In contrast, LN-sets allow systems to have models for arbitrary independent subsets of relevant speakers.

**LI-set mapping** In LI-sets, there exists for each model of a system exactly one corresponding model in the other system, thus resulting in a one-to-one mapping. Algorithm 1 illustrates the pseudo code for LI-set mapping. The algorithm first calculates all distances between models of both systems. Subsequently, models with minimum distances are determined under the one-to-one mapping constraint.

**Algorithm 1** Speaker ID mapping for local-identical (LI) speaker ID sets.

1. Create distance matrix \( D_{i,j} = D(C^1_i, C^2_j) \) of all distances between models of the first system \( (C^1_i \forall i = 1, \ldots N_1) \) and models of the second system \( (C^2_j \forall j = 1, \ldots N_2) \)

2. \( k = 1 \)

3. Search in matrix \( D_{i,j} \) for the indexes of the smallest element:

\[
(id_1, id_2) = \arg\min_{i,j}(D_{i,j})
\]

4. Store the indexes as a new mapping: \( \text{map}(k) = (id_1, id_2) \)

5. Remove all distances of models \( C_{id_1} \) and \( C_{id_2} \) from \( D_{i,j} \)

6. \( k = k+1 \)

7. If notEmpty\( (D_{i,j}) \) Then goto(3) Else return(map)
**LN-set mapping** In LN-sets, not every model of one system has a corresponding model in another system. Speaker models may be missing in one of the systems and thus no connection to another system’s models can be made.

For LN-sets we used a two step approach to derive a mapping. The first step consists of applying the LI-set algorithm (see Algorithm 1). As result, a mapping under the one-to-one mapping constraint $N_{\text{min}} = \min(N_1, N_2)$ is obtained, with $N_1$ and $N_2$ numbers of models in each systems’ databases. Thus, $|N_1 - N_2|$ models without a mapping can be identified by this step. In a second step, all $N_{\text{min}}$ mappings are tested. A mapping between $C^1_i$ and $C^2_j$ is accepted or rejected with the decision function:

$$f_{ICN}(C^1_i, C^2_j) = \begin{cases} 1, & \text{if } D(C^1_i, C^2_j) \leq \Delta \\ 0, & \text{else} \end{cases}. \quad (3.5)$$

Here $\Delta$ is the decision function’s threshold. If a mapping distance is less than or equal to $\Delta$, this mapping $(i, j)$ is accepted, otherwise it is removed from the mapping table. The collaborative identification algorithms can handle NL-set mappings as illustrated in the example shown in Section 3.3.1.

### 3.4 Evaluation Dataset

As in the evaluation of the standalone mode - presented in the Chapter 2 - we selected the freely available Augmented Multiparty Interaction (AMI) corpus [67].

We targeted to evaluate situations where up to 24 speakers are involved. Thus, to analyse the performance of our collaborative systems approach, we extracted speech data from six meeting sets of the original corpus, in total 24 speakers (9 female, 15 male). We used a mixture of ad-hoc and scenario-based meetings.

For each speaker we extracted 8 minutes of speech out of four meetings. We used audio data recorded from the lapel microphones of all meeting participants to realistically match the situation of a wearable speaker annotation system: one single lapel microphone was the input of one wearable recognition system. Since every meeting had four participants we could use the data of four lapel microphones to simulate up to four parallel collaborative identification systems.

The AMI corpus provides annotations for words and other sounds, however lacks ground truth information on the actual speaker. Conse-
3.5. Results

Consequently we annotated each individual speaker of the selected dataset. In total, our evaluation is based on 192 minutes of annotated speech data. Speech segments that were annotated by AMI as cross-talk and non-speech gaps of larger than 1 s were omitted. The audio files of AMI were originally recorded with 16 kHz. We downsampled the data to 8 kHz, since this band provides the most relevant speaker information. An anti-aliasing FIR filtering was applied prior to downsampling.

A new speaker model was trained with a speech segment of $t_{train} = 20\, \text{sec}$ (as described in Section 2.4.4). This results in 24 training segments for the 8 minutes speech data available from each speaker.

To evaluate the collaborative operation, speaker databases for $N_{sys}$ collaborative systems were generated. Each system database consisted of $N_{sp}$ models, where each model was created using training segments of the system-specific channel. The collaboration performance was then evaluated on the remaining data. Recognition was performed on speech segments of $t_{rec} = 5\, \text{sec}$ (as described in Section 2.4.4). The total accuracy for $N_{sys} = \{1, 2, 3, 4\}$ and $N_{sp} = \{4, 8, 12, 16, 20, 24\}$ was calculated by simulating all possible combinations.

3.5 Results

This section details evaluation results for collaborative speaker identification in the use scenarios, presented in Section 3.2. We utilized the approach described in Section 3.4 to evaluate the collaborative use scenarios.

For evaluation we used Matlab and Simulink as our simulation environment. The speaker identification system, as detailed in Section 3.3, was implemented in Simulink. Collaborations between the systems were simulated in Matlab using the Simulink model. We focus on evaluating performance bounds for these use scenarios. Section 3.5.1, 3.5.2, and 3.5.3 present the results for the use scenarios CC-LI, CO-LI, and CO-LN. In Section 3.5.3 performance of the three use scenarios are compared. Finally, in Section 3.5.5 results for acquiring a speaker ID mapping are presented.

3.5.1 Collaborative-Closed, Local-Identical (CC-LI) Analysis

In a CC-LI use scenario, collaborative systems share speaker matching distances during recognition for all relevant speakers. In a collaborative
Figure 3.4: Performance of multi-system collaboration in the three use scenarios, in comparison to a standalone system. Performance is shown for varying the number of relevant speakers. Figure 3.4(a) shows the results for the CC-LI case, Figure 3.4(b) for CO-LI, and Figure 3.4(c) for CO-LN.
3.5. Results

speaker recognition, these results are weighted, resembling a voting by all collaborative systems.

Figure 3.4(a) shows the collaborative speaker recognition performance in a CC-LI use scenario for 2 to 4 collaborating systems, and with the standalone system performance as reference. Relevant speaker sets were varied between 4 and 24. These results show that performance continuously increases with the number of collaborating systems. Hence, collaboration in a CC-LI scenario provides a clear benefit compared to a standalone system’s performance. Moreover, benefits of the collaboration are larger for settings with high number of relevant speakers. With four relevant speakers, a collaboration of four systems can increase accuracy from 0.88 to 0.97 (+9%), whereas with 24 relevant speakers an improvement from 0.70 to 0.90 was observed (+20%).

3.5.2 Collaborative-Open, Local-Identical (CO-LI) Analysis

For CO-scenarios, collaboration is performed in speaker recognition and new speaker detection functions. We evaluated two CO-scenarios regarding local speaker sets, thus CO-LI and CO-LN, as introduced in Section 3.2. In a CO-LI use scenario, collaborative recognition was performed as in the CC-scenario, presented above: all collaborating system share their matching distances during recognition. However, since tested speakers can be unknown to all collaborating systems, a collaborative new speaker detection was performed subsequent to the recognition step.

Figure 3.4(b) presents the results of CO-LI. Here, performance of 2 to 4 collaborating systems are compared to a standalone system, for varying the number of relevant speakers. We analyzed CO-LI condition by iteratively leaving each speaker out of the collaborative set once. Models of all other relevant speakers were maintained in the databases. This evaluation provided a worst-case performance of the new speaker detection, since each left-out speaker should be detected as a new one, while all other models were available.

Similar to the CC-scenarios presented above, collaboration in CO-LI use scenario improves standalone system performance. Clear performance increases were observed for large numbers of relevant speakers. Four collaborative systems in a setting with 4 relevant speakers improved accuracy from 0.87 to 0.96 (+9%). In a setting with 24 relevant speakers, the improvement was from 0.69 to 0.90 (+21%).
3.5.3 Collaborative-Open, Local-Nonidentical (CO-LN) Analysis

CO-LN is the most general collaboration use scenario. Here, the collective collaborates for speaker recognition as well as for new speaker detection. As opposed to CO-LI, systems miss relevant speaker models to fully collaborate. Consequently, collaborative recognition and new speaker detection need to rely on unbalanced voting. A distance normalization helped to reduce system dependency, as described in Sections 3.3.1.

Figure 3.4(c) presents the results for CO-LN. In this analysis, the probability that a test speaker is known by $N_{\text{known}}$ systems was assumed to be $p_{\text{known}}(N_{\text{known}}) = \frac{1}{N_{\text{sys}}+1}$ for $N_{\text{sys}} = \{1, 2, 3, 4\}$.

In a setting with 4 relevant speakers, 4 collaborative systems improve accuracy from 0.87 to 0.92 (+5%), whereas with 24 relevant speakers, accuracy is improved from 0.69 to 0.85 (+16%). We attributed lower performance of CO-LN as compared to CO-LI to the reduction of collaboration information.

3.5.4 Collaborative Scenario Comparison

A comparison of collaboration performance among all use scenarios is shown in Figure 3.5. Here, performance gains for 4 collaborating systems compared to standalone mode are shown for varying the number of relevant speakers. It can be observed that gains for both LI-set based use scenarios are similar. In contrast, gains for CO-LN are \(\sim 4\%\) lower.

Figure 3.6 presents a detailed performance analysis for collaborations in CO-based use scenarios with 24 relevant speakers. The identification performance was analyzed here regarding the number of collaborative systems knowing the tested speaker, $N_{\text{known}}$, and the total number of collaborating systems $N_{\text{sys}}$. We observed that performance improvements of collaborating systems strongly depends on $N_{\text{known}}$. With 4 collaborative systems in CO-LI, only performances with $N_{\text{known}} = \{0, 4\}$ are relevant. For $N_{\text{known}} = 0$ only the new speaker detection was activated, since no models existed. Thus, this condition does not reveal speaker IDs and can be seen as starting point to learn new models.

The CO-LI results show a performance boost through collaboration. In CO-LN, performance for $N_{\text{known}} : 0 < N_{\text{known}} < N_{\text{sys}}$ are relevant as well. Here, lower performance improvements can be observed due to the challenge of nonidentical system databases. For increasing $N_{\text{known}}$,
3.5. Results

Figure 3.5: Comparison of all three use scenarios with 4 collaborating systems. Performance is shown as accuracy gains through collaboration for varying the number of relevant speakers compared to standalone mode. Points are denoted by their absolute accuracy.

performance improves, as expected. Moreover, for constant \(N_{\text{known}}\), increasing the number of collaborating systems leads to performance gains as well. This is due to the fact that systems which do not know a speaker help indirectly to elevate the correct speaker model (see Section 3.3.1 for an example illustrating this system behavior).

3.5.5 Speaker ID Mapping

The developed speaker ID mapping algorithms are described in Section 3.3.3. To evaluate speaker ID mapping performance, two speaker model databases were created by using training segments of \(N_{sp}\) speakers from two different channels. These generated databases were used with the mapping algorithms to determine mapping error counts. We simulated every possible combination of pairs of databases to evaluate mapping performance for \(N_{sp} = \{4, 8, 12, 16, 20, 24\}\). For LN-set speaker ID mapping, we compared two model databases, once with all speakers available in both databases and subsequently with a missing speaker in one of the two databases, where each speaker was removed once.

Figure 3.7 shows the mapping performance obtained between two systems for LI- and LN-sets and for varying the number of relevant
speakers. For both mapping algorithms, performance decreases with increasing relevant speakers in the databases. For LI-set, using one-to-one mapping, accuracy drops from 0.88 for 4 relevant speakers to 0.63 for 24 relevant speakers. LN-set mapping performance drops from 0.71 to 0.41 for 4 and 24 relevant speakers respectively. These results clearly show the benefit of using the less complex LI-set mapping, where an unknown model detection algorithm is not needed. It can be concluded that the LI-set mapping can certainly be used with 12–16 relevant speakers in one meeting at an accuracy $\geq 0.7$. Using the LN-set mapping, relevant speakers in one meeting are constrained to 4, at an accuracy $> 0.7$.

### 3.6 Discussion

Our evaluation revealed that a collaboration on speaker recognition and new speaker detection among personal speaker identification systems can substantially increase performance. Results for CC-LI and CO-LI show gains of $\sim 20\%$ at 24 relevant speakers.

**Figure 3.6:** Performance of multi-system collaboration in CO-based use scenarios with 24 relevant speakers, in comparison to a standalone system. Recognition accuracy is shown for 0 to 4 collaborative systems. The number of systems known the speaker was varied from 0 to 4.
3.6. Discussion

Figure 3.7: Performance analysis of speaker ID mapping between two databases for LI- and LN-set systems. The performance is shown for different numbers of relevant speakers in one meeting.

3.6.1 Information Exchange in Collaboration

Collaboration in mobile and wearable systems is constrained by wireless communication bandwidth and power consumption. To this end, collaboration could be performed by fusing information at different levels of the processing stack, including raw audio data, processed sound features, recognition and detection result, and speaker model levels. Clearly, a viable collaboration concept for mobile systems should make use of a compressed information exchange. However, this inherently limits collaborative information to improve performance.

For our system architecture, fusion at raw data, processed sound features, and speaker model levels would require a collaborating system to transmit net data rates of 128 kbit/s (8000 kHz · 16 bit), 38.40 kbit/s (100 frames · 12 LPCC · 32 bit), and 61.44 kbit/model (12 LPCC · 16VQ · 32 bit), respectively. In contrast, information fusion at the level of speaker recognition and new speaker detection requires 3.2 kbit (assuming 100 speakers: 100 · 32 bit) and 1 bit, respectively. As both were calculated every 5 s, a bandwidth far below the rates stated above is required.

3.6.2 Challenges in Collaborative Identification Systems

Although channel compensation was used to normalize the audio features between different systems (see Section 2.3.1, the different channel properties between collaborating systems are a critical concern. In combination with differences in speaker distance and room reflection effects, the recorded sound data and derived speaker models could differ
substantially. This condition critically constrains collaboration options and required a speaker ID mapping. E.g. it is not feasible to compare complete databases between systems. Our choice to solely exchange recognition and detection results, reflects these constraints.

When owners of personal speaker identification systems enter into a conversation or meeting, their systems need to perform an initial speaker ID mapping. In our approach, this mapping enables a collaboration. Our performance results show that this mapping is feasible. However, the mapping is more challenging for LN-set systems, in which individual systems have different speaker databases. While the implementations presented in this work would permit collaborations with 16 relevant speakers in LI-set systems, it is limited to 4 relevant speakers for LN-set systems at a bound of 70% accuracy. Although further work is needed to improve speaker ID mapping performance, this function is not often used. Typically, a speaker ID mapping would be performed upon initiating a collaboration only, e.g. at the beginning of a meeting. Any subsequent ID mapping, e.g. when a new speaker was detected, would use the collaborative new speaker detection to determine the ID mapping.

Background noise disturbing the speech signal (e.g. street noise) is a key challenge in speaker recognition. This work did not focus on analysing the effect of noise in particular. Our evaluation dataset was however composed from real indoor meetings, including typical noise levels (e.g. street, noise from participants). Thus, all performance results presented reflect natural meeting environments. A further, dedicated noise analysis could reveal additional benefits of our collaboration approach. Often audio channels of individual sound recording systems have different noise and channel properties. Thus, collaborative systems could improve individual identification performance in environments with high background noise even more than in rather silent settings.

3.6.3 Prototype Implementation

Our personal speaker identification and collaboration approach is designed to be used with standard smart phones. Such mobile and wearable devices are limited in processing capabilities and power consumption. Thus, minimizing algorithmic complexity and communication bandwidth between collaborative partners is essential.

In the last chapter we confirmed the feasibility of a speaker identification and learning system working in standalone mode. This system
was implemented on a custom wearable device prototype, based on a TI TMS320C67 DSP, audio interface, USB host connection, and battery power supply (see Figure 2.7). The system was designed to be worn as belt attachment. With this system we were able to train and recognize up to 150 speakers in real-time. The device could continuously operate for 8.6 hours between battery recharges.

We expect that this system could be extended to operate in collaborative mode as targeted in this work by adding the 'collaborative identification' function block and a wireless transceiver. Given that every system is sending 16 bit/s and conversation partners are in a range of typical meeting room sizes of about 15 meters, we expect that ultra low-power radio solutions, such as ZigBee are feasible. Since ultra low-power transceivers are not yet common for smart phones, Bluetooth could be used as intermediate alternative for collaborative communication.

3.7 Conclusion and Future Work

In this chapter we introduced a collaborative personal speaker identification approach that can be generally applied in different use scenarios. Due to the diversity of situations in which a mobile or wearable identification system can be used, operation conditions and collaboration options vary widely. For this purpose we introduced a collaboration use scenario concept that accounts for unknown speakers and independent speaker model databases of participating systems. Our analysis confirmed that the scenarios have practical applications in different conversation and meeting situations. Furthermore, evaluations of different use scenarios showed that our speaker identification system provides useful performance in standalone and collaborative operation modes.

When compared to a standalone operation, the collaboration among four personal identification systems increased system performance. Gains where up to 9% at 4 relevant speakers and up to 21% at 24 relevant speakers for systems with locally identical speaker sets. For the most challenging scenario of collaborative open and locally non-identical speaker sets, still gains of 5% and 16% at 4 and 24 relevant speakers respectively, were achieved. We concluded that both collaborations, to recognize known speakers and to detect new speakers, provide substantial benefits regarding system robustness.

From our performance comparison among collaboration scenarios we concluded that allowing unknown speakers in a conversation does
not hamper system performance and gains achieved through collaboration. In contrast, allowing systems to have nonidentical speaker sets clearly reduced collaboration gains. Moreover, we found that our collaborative fusion provides benefits even in situations, where only one system knows the actual speaker. In this situation, collaborating systems indirectly elevate the correct speaker by returning low matching scores for their models.

We specifically developed the system architecture to cope with all use scenarios considered during system evaluation. Moreover, system architecture and implementation considered the requirements of mobile and wearable systems regarding communication and algorithm complexity. In particular, efficient solutions were found to exchange collaboration information while minimizing bandwidth requirements. The choice for exchanging speaker recognition and detection results represents a tradeoff between system dependency, due to channel properties and information detail. We concluded that a collaborative personal speaker identification system can be realized with currently available audio, communication, and processing capabilities in mobile devices.

Further work should address speaker ID mapping approaches to optimize the performance of ad-hoc mappings when system owners enter into a conversation or meeting. Additionally, the benefit of collaboration in different noise environments should be investigated.
Speaker Identification in Daily Life *

This chapter introduces MyConverse, a personal conversation recognizer and visualizer for Android smartphones. MyConverse is based on the speaker identification system presented in Chapter 2 including improvements in speaker recognition and modelling. A user study is presented, to show the capability of MyConverse in daily life situations to recognize and display user’s communication patterns.

*This chapter is based on the following publication:
M. Rossi, O. Amft, S. Feese, C. Käsli, and G. Tröster. MyConverse: Personal Conversation Recognizer and Visualizer for Smartphones. (Recognise2Interact 2013) [73]
4.1 Introduction

In this chapter we present MyConverse, a personal conversation recognizer and visualizer for Android smartphones. MyConverse uses the smartphone’s microphone to continuously recognize the user’s conversation during its daily life autonomously on the smartphone. MyConverse identifies known speakers in conversations. Unknown speakers are detected and trained for further identification. MyConverse can be used as personal logging tool for daily life conversations. A user can review his conversations by e.g., analyse his speaking behaviour or look-up a forgotten name of a speaker. For privacy concerns, MyConverse never stores captured raw audio data on the smartphone’s storage. Audio data is immediately processed such that speaker related information is extracted, however speech content is removed. In particular this chapter makes the following contributions:

1. We present the system architecture of MyConverse. The system is based on the personal speaker identification system presented in Chapter 2 including optimizations in feature extraction and speaker modelling. An extensive study of the recognition performance is done based on the dataset already used to evaluate our previous system.

2. We discuss the implementation of MyConverse as an Android app. In particular we show how conversations can be visualized on the smartphone. We show that the app can continuously and unobtrusively run on a commercial available Android smartphone for more than one day.

3. We present our daily-life evaluation of MyConverse. MyConverse was evaluated in different real-life situations, e.g. in a bar or street. Additionally, an evaluation study was performed where MyConverse was tested on full-day recordings of person’s working days.

4.2 Related Work

Most related to our work are the following proposed systems also focusing on speaker identification on a smartphone: EmotionSense [74], Darwin [75], and SpeakerSense [28]. EmotionSense is a sensing platform for social psychology studies based on mobile phones including a speaker
recognition sub-system. Speaker training data is gathered offline in a setup phase. In contrast, we focused on unsupervised speaker identification avoiding an offline training phase. Darwin is a collaborative sensing platform for smartphones. Speaker identification was used as an example application to demonstrate the identification using multiple phones. We focus on improving speaker recognition using one independent phone without any further infrastructure. However, our work could contribute to overall improvements in collaborative approaches (as introduced in Chapter 3). SpeakerSense investigated in acquiring training data from phone calls and in a semi-supervised segmentation method for training speaker models based on one-to-one conversations. We focused on dynamic learning of new speakers, without the assumption of one-to-one conversations or prior phone calls for speaker training.

4.3 Architecture

The aim of MyConverse is to unobtrusively recognize the user’s interactions throughout the day. MyConverse runs on the user’s smartphone continuously detecting speech, identifying speaker, and record information of the user’s conversations on the smartphone. While MyConverse identifies known speakers (e.g. stored in the system’s speaker models database, see Figure 4.1) by their unique id and name, an unknown speaker is detected, subsequently a new speaker model is learned and stored with a unique speaker id for future recognition. MyConverse saves the following information of each user’s conversation: start and end time, position of the conversation, the identity of each speaker involved in the conversation, and the time segments, when the individual persons spoke. In this section, we detail the recognition architecture of MyConverse and its implementation on the Android platform. In the next section we present how MyConverse uses the recognition to visualize user’s communication behaviours. Figure 4.1 depicts the components of MyConverse. The architecture was implemented as an Android app and completely runs locally on a Android smartphone. The input of the system was the internal microphone of the smartphone, or the microphone of the connected headset. The microphone was continuously sampled with a sampling rate of 16kHz at 16bit depth and is then processed by the front-end processing.

The Front-end processing unit targets to extract speaker-dependent features from the audio signal using non-speech filter, pre-
emphasis, and feature extraction. The non-speech filter is a speech detector removing all audio segments containing no speech data. We used the non-speech filter proposed by Raj et al. [76]. Speech segments longer than 0.5s were further processed in the pre-emphasis step, smaller speech segments and non-speech segments were discharged. The pre-emphasis filter amplifies higher frequencies bands and removes speaker independent glottal effects. For filtering we used a commonly used filter transform function [51]: \( H(z) = 1 - \alpha z^{-1} \), with \( \alpha = 0.97 \). After pre-emphasis, speaker-dependent features are extracted from the audio signal. We evaluated the following audio feature set which have been previously used in other speaker identification tasks: MFCC (Mel-frequency cepstrum coefficients, e.g. [28]), MFCCDD (MFCC with first and second derivatives, e.g. [77]), LPCC (linear prediction cepstral coefficients, e.g [54]), AM-FM (e.g [78]), and wavelets (e.g. [79]). For feature extraction a commonly used framing method [28, 54] was used: a sliding window of 32 ms length with an overlap of 16 ms. Prior feature extraction, windows were filtered by a Hamming window [54]. The output of the front-end unit is a \( N \) dimension feature vector \( \vec{x} = [x_1, x_2, \ldots, x_N]^T \).

We implemented the front-end processing unit using the CMU Sphinx speech recognition framework\(^1\). Sphinx is a framework intended to build speech recognition systems. The framework was completely written in Java and its core library runs on Android systems. In Sphinx the front-

\(^1\)CMU Sphinx Open Source Toolkit For Speech Recognition: http://cmusphinx.sourceforge.net/
4.3. Architecture

End processing is built as a pipeline of processing units. Configuration of the pipeline is done in an xml file defining the sequence of units as described before.

The **Speaker modelling** unit generates speaker models for unknown speakers and stores them in the **Speaker models database**. Speaker models were created based training data consisting of a set of feature vectors \( X_m = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_M\} \) generated by the front-end processing unit. We defined the *training length* \( T_t \) as the length of the speech signal used to train a new speaker model. \( M \) corresponds to the amount of feature vectors extracted from the speech signal of length \( T_t \). For modelling, we used Gaussian Mixtures Models (GMM), a widely used modelling technique in speaker recognition (e.g. [80, 78]). Using Expectation-Maximization (EM) a GMM with \( L \) mixture components are mapped to fit the training data. Additional to GMM modelling we evaluated its extended approach GMM-UBM [81]. The difference to GMM is the additional Universal Background Model (UBM). UBM is an pre-generated GMM modelling the speech of multiple random speakers. To create a new speaker model the UBM is adapted to the new training data. The modelling procedure was done without EM according to Reynolds et al. [82]. Because of the lightweight modelling based on the UBM, GMM-UBM has the advantage that less training data is needed and computational complexity of training can be reduced compared to the EM algorithm. We compared GMM and GMM-UBM and evaluated different training length \( T_t \) and number of Gaussian mixture components \( L \), which are presented in the evaluation section. To implement the speaker modelling unit, we integrated the code of Mary TTS 5.0 library\(^2\) for GMM training and for GMM-UBM training we wrote our own code based on [81].

The **Speaker matching** unit compares a set of feature vectors \( X_r = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_R\} \) of a speech segment with stored database speaker models \( \{\lambda_{S_1}, \ldots, \lambda_{S_n}\} \) of speakers \( \{S_1, \ldots, S_n\} \) and identifies the best matching speaker model. Speaker matching was done based on speech signal with a *recognition length* \( R_t \). Similar to modelling, \( R \) corresponds to the amount of feature vectors extracted from the speech signal of length \( R_t \). We evaluated different recognition length presented in the evaluation section. The best matching speaker \( \hat{S} \) was selected by:

\(^2\)Mary TTS 5.0:https://github.com/downloads/marytts/marytts/marytts-5.0.zip
\hat{S} = \arg \max_{S_1 \leq k \leq S_n} p(X|\lambda_k), \text{ where } p(X_r|\lambda_k) \text{ is the probability of the model } 
\lambda_k \text{ given } X_r \text{ (see [83] for details).}

The **New speaker detection** unit detects if speech data is from a known speaker (e.g. already modelled and stored in the model database) or from an unknown speaker. New speaker detection was defined as a speaker verification problem: The hypothesis that the set of feature vectors $X_r$ belongs to the speaker $\hat{S}$ has to be verified. As proposed in [82], this is verified by comparing the probability of model $\hat{S}$ and the UBM model: $LLR_{\hat{S}} = \log p(X_r|\lambda_{\hat{S}}) - \log p(X_r|\lambda_{\text{UBM}})$. $LLR_{\hat{S}}$ is additionally normalized for better detection accuracy (see [82]): $\bar{LLR}_{\hat{S}} = \frac{LLR_{\hat{S}}(X) - \mu_{\text{LLR}}}{\sigma_{\text{LLR}}}$, where $\mu_{\text{LLR}}$ is the mean and $\sigma_{\text{LLR}}$ the variance of the set $\{LLR_k(X)|k = S_1, \ldots, S_n\}$. A new speaker is detected if $\bar{LLR}_{\hat{S}}$ is below the threshold $T_S$, else the speaker is identified by the speaker $\hat{S}$. $T_S$ was chosen such that detection accuracy is optimized over the training set. The new speaker detection unit outputs the identified speaker $S_{id}$, which corresponds either to the matched speaker $\hat{S}$ or a newly created speaker id for the new speaker. Additionally, in case of detection of a new speaker, the speaker modelling unit is activated to create a new speaker model.

The **Conversation logger** unit divides the recognized speaker information in conversations and stores it in the database. The start and stop time of conversations were defined by silent audio segments: If during 2 min no speech data was detected, the last detected speech segment was defined as the conversation’s end. The start of a new conversation was then defined by a new speech segment. For each conversation, a GPS location was stored. For energy-efficiency, GPS location was sampled only once at the beginning of a conversation. All the presented units are running in an Android Background Service. This enables to continuously recognize user’s conversations, even if other applications are in the foreground or the smartphone’s display is turned off. Only the **User Interface** (UI) is running as an Android Activity.

### 4.4 Visualization

Figure 4.2 shows the user interface of MyConverse. The app gives the user the possibility to start/stop the recognition and see real time infor-
4.5 Evaluation

Several aspects of MyConverse were evaluated. We present recognition performance of the system with different parameter sets (see Sec-
Figure 4.3: Visualization possibilities in MyConverse. A single conversation or all conversations together can be visualized.

4.5.1 Parameters of the Recognition System

We tested our recognition system with different parameter sets. For this evaluation, the freely available Augmented Multiparty Interaction (AMI) corpus\(^3\) was used. This dataset provides more than 100 hours of meeting scenes recorded from different microphones installed in the meeting room and worn by each participant. We extracted speech data from 24 speakers (9 female, 15 male). From each speaker 5 minutes of speech data was extracted.

System’s recognition accuracy for the different feature sets and the two modelling techniques (GMM and GMM-UBM) were tested. For this experiment, the new speaker detection unit was disabled and only the speaker matching unit was tested. Speaker models of all 24 speakers were trained with a training length \(T_t\) of 15 s and stored in the systems model database. The rest of the data was used to test the matching per-

\(^3\)AMI Meeting Corpus: https://corpus.amiproject.org/
4.5. Evaluation

Figure 4.4: Recognition accuracy of the 24 speakers using different feature sets and modelling techniques (GMM and GMM-UBM). Training length was set to 15 s, recognition length to 3 s. For this evaluation new speaker detection was disabled. The baseline denotes the accuracy of a random recognition system selecting each speaker with equal probability.

Performance on a recognition length $R_t$ of 3 s. For GMM and GMM-UBM $L = 16$ mixture components were used. The UBM was trained on one hour of speech data from over 100 speakers not included in the speaker corpus. Figure 4.4 shows the results. The highest accuracy was reached by MFCCDD feature set. GMM-UBM reaches higher recognition accuracy compared to GMM, except for LPC features. This was expected, since GMM needs more data for an accurate speaker modelling. Further analysis showed that if training length is smaller then 25 s, GMM-UBM outperforms GMM (using the MFCCDD feature set). Only with higher training length GMM exceeded the recognition performance of GMM-UBM. However, since for our system it was crucial that new speaker were modelled with small training length, GMM-UBM was selected. An additional benefit of GMM-UBM is the faster modelling compared to GMM. Moreover, the number of Gaussian mixture model components $L$ was analysed. $L$ of GMM and GMM-UBM was varied between 3 and 64. Using the MFCCDD feature set, recognition accuracy increased from 3 to 16 components. However, accuracy did not increase with more mixture components.
Figure 4.5: Recognition accuracy of the 24 speakers using MFCCDD feature set and GMM-UBM. The system performance trade-off with regard to training and recognition time is marked.

The training length $T_t$ and the recognition length $R_t$ are crucial parameters of the recognition system. As it is desirable to recognize a speaker in short speech segments, recognition time must be short. In addition, there is potentially only little training data available during conversations to learn a new speaker online. We analysed the number of feature vectors needed to train (training length $T_t$) and recognize a speaker (recognition length $R_t$). For this evaluation we used the MFCCDD feature set and the GMM-UBM modelling approach. Figure 4.5 shows the system performance with regard to training and recognition time. The results confirm that below 3 s of recognition time system performance decreases rapidly. In contrast, only marginal improvements are obtained for more than 3 s of recognition time. With 5 s of training data per speaker, recognition accuracy was around 50%, while $> 30$ s did not further improve performance.

We evaluated the recognition performance of the new speaker detection unit and the threshold parameter $T_S$. For this evaluation we again modelled all 24 speakers with a training length $T_t$ of 15 s. The rest of the data was used to test the new speaker detection on a recognition length $R_t$ of 3 s. The same UBM model was used as presented above. For each test segment of speaker $S$ we calculated the $\overline{\text{LLR}}_S$ and $\overline{\text{LLR}}_{S_{\text{best}}}$, where $S_{\text{best}}$ is the best matching speaker model excluding the speaker
model \( S \). In the optimal case, all \( \text{LLR}_{S_{\text{best}}} \) should be smaller than the threshold \( T_S \) and be detected as a new speaker, whereas \( \text{LLR}_S \) should be above \( T_S \) and be detected as known speaker. We selected \( T_S = 2.1 \) which optimizes the new speaker detection accuracy (88%).

For further analysis we used the following parameter configuration: as feature set the MFCCDD was selected, GMM-UBM with \( L = 16 \) was used for speaker modelling, training length \( T_t \) and recognition length \( R_t \) was set to 15 s and 3 s, respectively, and speaker decision threshold \( T_S \) to 2.1.

### 4.5.2 Recognition Performance in Real-Life Environments

We recorded conversations in different locations: quiet room, busy street, and bar. Each conversation consisted of three people either sitting at a small table (e.g. in quiet room and restaurant), or standing together in a pedestrian zone. The distance between the speakers was always smaller than 1 meter. A conversation lasted for 15 min and was recorded with a headset of an Samsung Galaxy S2 Android smartphone. The headset was worn by one of the participant such that the microphone was fixed near his neck pointing towards the other speakers. The conversation was not scripted, however, to ensure that the system can learn the speakers right from the beginning, each participant started to speak a segment of at least 20 s length. After recording, the conversation was manually labelled to create the ground truth: Speech segments of a speaker larger then 2 s were labelled by their starting and stopping time and the speaker id. In total 5 groups of 3 people recorded their conversations on the three locations. In total we collected audio data of 225 min.

Recognition performance of the system was calculated by comparing the system’s prediction with the manually labelled information. A speech segment was counted as correctly recognized, if the ground truth segment and predicted segment have the same speaker id and matches at least 80 % of the ground truth segment’s duration. The recognition accuracy of a conversation is the ratio of the number of correctly labelled segments divided by the number of ground truth labels. The recognition accuracy is shown in Figure 4.6. The accuracy of each location is an average over 5 conversations. For each location, the measured background noise level in dBA is annotated.

As expected, the conversation’s speaker recognition accuracy for individual speech segments in the quiet room showed the best results
Figure 4.6: Comparison of speaker identification in speech segments during real-life conversations. For each location, the background noise level is annotated.

(84%). Accuracy in the street location dropped to 67%, however the noise level increased to 65 dBA. Although the noise level in the bar was smaller (60 dBA), the accuracy dropped to 60%. This can be explained by speech signal from other people included in the background noise of the bar. Background noise from street was rather dominated by car noise.

4.5.3 Full-Day Evaluation Study

We investigated how well conversations were recognized in a person’s daily life routines. In this evaluation study we analysed how accurate the system can recognize a conversation and the involved speakers. Detailed speaker annotation during the conversation was not subject of this study. For this purpose we recorded full-day ambient sound of three persons. The participants were asked to record ambient sound during their day from morning, after getting dressed, until evening, when they came back from office. At least 8 h of audio data was recorded for each participant. The audio was recorded with a Samsung Galaxy S2 device and headset. Participants wore the headset such that the microphone was positioned near the neck. All conversations were annotated by the participants. Only the start and stop time of the conversations and the involved speakers were labelled. Conversations smaller than 1 min were ignored.

A conversation was counted as correctly recognized if a predicted conversation matches at least 80% of a ground-truth label. We additionally analysed the recognition of speakers within a specific conversation. If a speaker was involved in a conversation and the system correctly predicted a segment of this speaker within this conversation, the pre-
Figure 4.7: Recognition accuracy for conversations and speakers in the full-day evaluation study.

Additionally, we analysed the runtime performance of the MyConverse app. The CPU usage of the app during speech and non-speech was measured as follows: the MyConverse app was started on the testing device and other running apps were closed. During 5 min of continuous speech the CPU load of the app was measured every 5 s with the Android Task Manager. The same measure was repeated for continuous non-speech (e.g. office noise of 30dBA level). The measurement resulted in an average CPU load of 30% for continuous speech and 5% for non-speech. CPU load in the non-speech case is smaller, because non-speech segments are not further processed. We further investigated how long the app can continuously run on the smartphone in battery mode. For this test, MyConverse app was started on the test device with a fully loaded battery. Other running apps were closed and the display was switched off. The test device was then positioned in an environment either with continuous speech or continuous non-speech background sound. To generate a continuous speech environment, we used a loudspeaker to play meeting recordings of the AMI corpus presented above. Additionally, in both cases the app triggered the phone’s position every 10 min. The time was measured until the device automatically switched off due to low battery. The experiment was repeated for each environment three times. The measurement resulted that in an environment with continuous speech the device runs in average for 7 h. In the non-speech environment the average runtime was 25 h.
4.6 Conclusion

We presented MyConverse, a personal conversation recognizer and visualizer for Android smartphones. MyConverse provides real-time speaker identification and online new speaker training functions that operate in parallel to identify speakers from a model database, detect unknown speakers, and enrol new speakers. Additionally, MyConverse visualizes user’s daily life conversations on the smartphone. MyConverse was optimized such that new speakers are enrolled with a small amount of training data, and known speakers are recognized on small speech segments. Evaluation showed that ConverseSense can recognize conversations in different real-life situations throughout the day.
This chapter presents AmbientSense, a personal activity and location recognition system implemented as an Android app. AmbientSense is implemented in two modes: in autonomous mode the system is executed solely on the smartphone, whereas in the server mode the recognition is done using cloud computing. Both modes are evaluated and compared concerning recognition accuracy, runtime, CPU usage, and recognition time.

*This chapter is based on the following publication:
5.1 Introduction

Real-time sound based inference of a user’s context could be used for people-centric sensing applications. For example, a smartphone can automatically change its profile while in a meeting, refuse to receive calls, or it can provide information customized to the location of the user. New smartphones with high computational power and Internet connectivity enable to make such inferences wearable and in real-time, either directly on the phone or through a server.

In this chapter, we propose AmbientSense, a real-time ambient sound recognition system that works in a smartphone setting. We present the design of the system, and its implementation in the smartphone setting working in two modes: In the autonomous mode the system is executed solely on the smartphone, whereas in the server mode the recognition is done in combination with a server. Our evaluation compares the two running modes focusing on the recognition accuracy, runtime, CPU usage, and recognition time on different smartphone models.

5.2 Related Work

The problem of ambient sound classification has been an active research area. Sound has been shown to be a sensing modality to recognize activities of daily living in locations such as bathroom [12], office [85], kitchen [33], workshop [85], and public spaces [13]. Mesaros et al. evaluated an ambient sound classification system for a set of over 60 classes [14]. Unlike other modalities sound has been shown to be robust to a range of different audio capturing locations, like the pocket of a trouser [17].

Only a few works addressed the implementation of real-time sound classification on limited resources of wearable devices. SoundButton was one of the first dedicated hardware for ambient sound recognition proposed by Stäger et al. [85]. More recently three systems using smartphone-based solutions were proposed: Miluzzo et al. [38] presented a framework for efficient mobile sensing including sound, Lu et al. [39] proposed a sound recognition system for voice, music and clustering ambient sounds, and Lu et al. [28] presented a sound speaker identification system on mobile phone. The novelty of our work is the implementation and evaluation of a smartphone-based system predicting ambient sound classes in real-time. Furthermore, we compare the two approaches, run-
ning the system solely on the smartphone and with the support of a server.

5.3 AmbientSense Architecture

Figure 5.1 illustrates main components of AmbientSense. The system receives ambient sound data as input and produces a context prediction using auditory scene models every second. The models are created in a training phase based on an annotated audio data training set. This section gives a detailed description of the system architecture.

![AmbientSense Architecture Diagram](image)

**Figure 5.1:** AmbientSense architecture illustrating the main components of the system. The main components are either implemented on the smartphone or on the server, depending on the running mode (detailed in Section 5.4).

**Front-end processing:** This component targets to extract auditory scene-dependent features from the audio signal. It has as input either continuous sound captured from a microphone, or in a database stored audio data. Audio data with a sample rate of 16 kHz sampled at 16 bit is used. In a first step, the audio data is framed by a sliding window with a window size of 32ms with 50% of overlap (framing). Each
The window is smoothed with a Hamming filter. In a consecutive step, audio features are extracted from every window (feature extraction). We used the Mel-frequency cepstral coefficients (MFCC), the most widely used audio features in audio classification. These features showed good recognition results for ambient sounds [12, 13]. We extracted the first 13 Mel-frequency cepstral coefficients, removing the first coefficient which is energy-dependent, resulting in a feature vector of 12 elements. In a next step, the feature vectors extracted within one second of audio data are combined by computing the mean and the variance of each feature vector element (merging). This results in a new feature vector $f_i^s$ with elements $i = 1, \ldots, 24$ and for each second $s$ of audio data. In a final step the feature vectors are normalized with $F_i^s = \frac{f_i^s - m_i}{\sigma_i}$, where $m_i$ are the mean values and $\sigma_i$ are the standard deviation values of all feature vectors of the training set (norm.). The normalization avoids the domination of one feature element in the classification task. The front-end processing outputs every second $s$ a new feature vector $F^s$.

**Classification:** For the recognition a Support Vector Machine (SVM) classifier with a Gaussian kernel was used [86]. The cost parameter $C$ and the kernel parameter $\gamma$ were optimized with a parameter sweep as described later in the evaluation section 5.5.1. The one-against-one strategy was used and an additional probability estimate model was trained, which is provided by the LibSVM library [86].

**Training and testing phase:** In a training phase the feature vectors of the training set including all auditory scene classes are computed and the SVMTrain component creates all auditory scene models which are stored for the recognition. In the testing phase the feature vectors are generated from the continuous audio data captured from the microphone. The SVMPredict component uses the stored auditory scene models to classify the feature vectors. Every second a recognition of the last second of audio data is created.

**Training set:** We tested our system on a set of 23 ambient sound classes listed in Table 5.1. The classes were selected to reflect daily life locations and activities that are hard to identify using existing GPS and activity recognition techniques. We collected a training set of audio data by recording for each class 6 audio samples from different sound sources (e.g. recordings of six different types of coffee machines). To record the audio samples we used the internal microphone of an Android Google Nexus One smartphone. The samples were recorded in the city of Zurich and in Thailand in different buildings (e.g. office, home, restaurant), and locations (e.g. beach, streets). For each recording we positioned
the smartphone closely to the sound-generating source or in the middle of the location, respectively. Each sample has a duration of 30 seconds and was sampled with a sampling frequency of 16kHz at 16bit. The audio samples are stored including the annotated label in the training set database.

<table>
<thead>
<tr>
<th>beach</th>
<th>crowd football</th>
<th>shaver</th>
</tr>
</thead>
<tbody>
<tr>
<td>bird</td>
<td>dishwasher</td>
<td>sink</td>
</tr>
<tr>
<td>brushing teeth</td>
<td>dog</td>
<td>speech</td>
</tr>
<tr>
<td>bus</td>
<td>forest</td>
<td>street</td>
</tr>
<tr>
<td>car</td>
<td>phone ring</td>
<td>toilet flush</td>
</tr>
<tr>
<td>chair</td>
<td>railway station</td>
<td>vacuum cleaner</td>
</tr>
<tr>
<td>coffee machine</td>
<td>raining</td>
<td>washing machine</td>
</tr>
<tr>
<td>computer keyboard</td>
<td>restaurant</td>
<td></td>
</tr>
</tbody>
</table>

**Table 5.1:** The 23 daily life ambient sound classes used to test our system. For each class 6 samples with a fixed duration of 30 sec were recorded.

**Running modes:** Two running modes have been implemented. In the *autonomous mode* the whole recognition system runs independently on the smartphone without any Internet connection required. The recognition results are continuously displayed on the phone as an Android notification. In the *server mode*, ambient sound capturing and the front-processing is done on the phone. The resulting features are sent to a server where the classification takes place. After classification, the predicted result is sent back to the phone, where it is displayed as an Android notification. This mode requires either Wi-Fi or 3G Internet connectivity, but enables to use more complex recognition algorithms on the server.

### 5.4 Implementation

The AmbientSense system was implemented in an Android smartphone setting. The main components (see Section 5.3) were implemented in Java SE 7 and are running on an Android smartphone or PC environment. For the implementation we used the FUNF open sensing framework [87] to derive MFCC features and the LibSVM Library [86] for the SVM modelling and recognition. In the rest of this section the specific Android and server implementations are explained in detail.
Figure 5.2: AmbientSense user interface on an Android smartphone. During recognition the predicted ambient sound class is displayed on the top as an Android Notification.

**Android implementation details:** Figure 5.2 shows an illustration of the user interface (UI). The application can either be set to build classifier models using training data stored on the SD card, or to classify ambient sound in the two running modes. The UI runs as an *Activity*, which is a class provided by the Android framework. An Activity is forced into a sleep mode by the Android runtime every time the UI of the Activity is not in the foreground. For the main components of the recognition system a continuous processing is needed. Thus, the main components of the application were separated from the UI and implemented in an Android *IntentService* class. This class provides a background task in a separate thread continuously running even in sleep mode, when the screen is locked, or the interface of another application is in front. The recognition result is shown as an Android Notification, which pops up on top of the display (see Figure 5.2). We used the *Notification* class of the Android framework to implement the recognition feedback. With this, a minimal and non-intrusive way of notifying the user about the current state is possible, while the ability to run the recognition part in the background is kept.
Server mode implementation details: In server mode, the phone sends every second the computed feature vector to the server for classification. This is implemented on the phone side with an additional IntentService handling the HTTP requests as well as the notifications on the screen. Therefore, communication with the server is running asynchronously enabling a non-blocking capturing of audio data and front-end processing. The feature vector is sent in a Base64 (included in the Android standard libraries) encoded JSON\(^1\) object as a JSONArray. On the server side, we used the Apache HttpCore NIO\(^2\) library, which provides HTTP client (as used by Android itself) and server functionality. The server listens for requests on a specified port, parses the data, computes the recognition and returns the predicted value.

5.5 Evaluation

We evaluated the autonomous- and server mode and compared both modes concerning recognition accuracy, runtime, CPU usage, and recognition time. For all the tests we used two Android smartphone devices: the Samsung Galaxy SII and the Google Nexus One. Table 5.2 shows the specifications of both phones. In all the tests a system with the 23 pre-trained classes (see Section 5.3) was used. For the autonomous mode Wi-Fi and 3G were deactivated, whereas for the server mode Wi-Fi was activated and 3G was deactivated on the smartphone. The server part was installed on an Intel Athlon 64 PC, with 4GB RAM and an Ethernet 100Mbit connection running on a Windows 7-64bit version.

<table>
<thead>
<tr>
<th></th>
<th>Samsung Galaxy SII</th>
<th>Google Nexus One</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU</strong></td>
<td>1.2 GHz Dual-core ARM Cortex-A9</td>
<td>1 GHz ARM Cortex A8</td>
</tr>
<tr>
<td><strong>RAM</strong></td>
<td>1024 MB</td>
<td>512 MB</td>
</tr>
<tr>
<td><strong>Battery</strong></td>
<td>1650 mAh Lithium-ion</td>
<td>1400 mAh Lithium-ion</td>
</tr>
<tr>
<td><strong>Wi-Fi</strong></td>
<td>IEEE802.11n, 300Mbit/s</td>
<td>IEEE802.11g, 54Mbit/s</td>
</tr>
</tbody>
</table>

Table 5.2: Specifications of the two testing devices: Samsung Galaxy SII and Google Nexus One.

\(^1\)http://www.json.org/java/index.html
\(^2\)http://hc.apache.org/httpcomponents-core-ga/httpcore-nio/index.html
5.5.1 Recognition Accuracy

The recognition accuracy was calculated by a six-fold-leave-one-audio-sample-out cross-validation. The SVM parameter \( C \) and \( \gamma \) were optimized with a parameter sweep. For \( C \), a range of values from \( 2^{-5} \) to \( 2^{15} \) was evaluated, while the range for \( \gamma \) was given between \( 2^{-15} \) to \( 2^5 \). This resulted in a recognition accuracy of 58.45\% with the parameter set \( C = 2^7 = 128 \) and \( \gamma = 2^{-1} = 0.5 \), which meets the result of previous work with a similar number of ambient sound classes [13]. Figure 5.3 displays the confusion matrix of the 23 classes. 12 classes showed an accuracy higher than 80\%, whereas 3 classes showed accuracies below 20\%. Since the recognition algorithm is identical for both running modes, the recognition accuracy holds for both modes.

![Confusion matrix of the 23 auditory sound classes.](image)

**Figure 5.3:** Confusion matrix of the 23 auditory sound classes.
5.5.2 Runtime

Runtime was tested by measuring application running time for five repetitions, each starting from fully charged phones. During all the tests, the display was turned off and no other tasks or applications were running. Figure 5.4 shows the average measured runtime of both running modes and for both testing devices. The Galaxy SII showed an average runtime of 13.75h for the autonomous mode and 11.633h for the server mode, whereas the Nexus One showed an average runtime of 11.93h for the autonomous mode and 12.87h for the server mode.

In server mode classification is done on the server which reduces power consumption for audio processing on the smartphone. In contrast, server mode uses the Wi-Fi adapter for communication using additional power. The Galaxy SII showed higher runtimes in autonomous mode compared to server mode. We conclude that for this smartphone the additional power needed for Wi-Fi connection was higher than the saved power in audio processing. The results of the Nexus One smartphone showed the opposite.

![Figure 5.4: Runtime average and the corresponding standard deviation of the testing devices in autonomous and server mode.](image)
5.5.3 CPU Usage

The CPU usage of the two modes was measured with the Android SDK tools. Figure 5.5 shows that the Galaxy SII used less CPU power in server mode. Since in server mode SVM recognition is not calculated on the phone, CPU power consumption is lower than in the autonomous mode. On the other hand, the Nexus One used more CPU in server mode. The reason for this is the additional Wi-Fi adapter which increased CPU usage of the Nexus One for the transmission task. The Galaxy SII has a dedicated chipset for the Wi-Fi communication processing. Furthermore, the Galaxy SII showed a higher fluctuation in processor load (standard deviations $\sigma_{\text{autom}}^g = 7.02\%$ and $\sigma_{\text{server}}^g = 6.58\%$) as the Nexus One ($\sigma_{\text{autom}}^n = 1.31\%$ and $\sigma_{\text{server}}^n = 1.69\%$). This is due to the fact that the Galaxy SII reduces the clock frequency of the CPU when the load is low$^3$.

![CPU usage graph](image)

**Figure 5.5:** CPU usage average and corresponding standard deviation of the testing devices for autonomous and server mode.

---

For a CPU profiling of the app we used the profiling tool included in the debugger of the Android SDK. We logged the trace files for both running modes to compare the different CPU time allocations. Figure 5.6 shows the profiling of the autonomous and the server mode on both testing devices for the execution steps Framing, FFT, Cepstrum, SVM, HTTP, and Rest.

![CPU profiling chart]

**Figure 5.6:** Comparison of the CPU profile between autonomous and server mode of the two testing devices. 100% CPU time corresponds to the data processing for one recognition.

The Rest includes operations like array copying, the computation of mean and variance and the audio recorder itself. The different execution steps are ordered in the way as they occur in the recognition chain (see Figure 5.1). The CPU time of one single task is measured as a percentage of the time it takes to complete the processing chain. The results show that server mode needed less time than in autonomous mode, also for the Google Nexus One (note that in contrast to Figure 5.5 the CPU profiling includes just the CPU usage of the processing chain). The FFT used to derive the MFCC features used up almost 50% and the calculation of the cepstrum used about 35% of the CPU time. The SVM predict used about 14% of the CPU time in autonomous mode and none in server mode, as in server mode the recognition is not done
Similarly there is no CPU time used for the HTTP client in autonomous mode, because the features do not have to be sent to a server. CPU load could be decreased by \( \sim 80\% \) moving the front-end processing to the server. However, in this case the raw audio data has to be sent to the server increasing the data rate from \( \sim 3\text{kbit/sec} \) to \( 256\text{kbit/sec} \).

### 5.5.4 Recognition Time

We define the recognition time as the time the system needs to calculate one recognition (see Figure 5.7). This includes in the autonomous mode just the execution time of the SVM recognition. In the server mode the recognition time includes the execution time of sending the feature vector (\( \sim 370\text{Bytes} \)) to the server, the SVM recognition on the server, and sending the result (\( \sim 5\text{Bytes} \)) back to the smartphone. The evaluation was done with both devices in autonomous mode, and in the server mode over the Wi-Fi as well over the 3G network.

![Figure 5.7: Definition of recognition time for autonomous and server mode.](image)

The measurement of the latency over Wi-Fi connection has been done from a LAN outside the server’s local network. The phones have been connected to a D-Link DI-524 Wi-Fi router following the 802.11g / 2.4GHz standard. For each mode we ran the experiment for 10min (600 recognitions). In Figure 5.8, the latency in the 3G network shows...
a higher mean and standard deviation for both phones which is comparable to other 3G latency measurements [88]. However, a 3G latency of approximately 260ms does not limit the usability of the application as this is still within the one second interval in which the request packets are sent.

![Recognition time of the testing devices in autonomous and Wi-Fi server mode, and 3G server mode.](image)

**Figure 5.8:** Recognition time of the testing devices in autonomous and Wi-Fi server mode, and 3G server mode.

## 5.6 Conclusion

We presented AmbientSense, a real-time ambient sound recognition system in a smartphone setting. The system can run autonomously on an Android smartphone (autonomous mode) or in combination with a server (server mode). For 23 ambient sound classes, the recognition accuracy of the system was 58.45%, which meets the result of previous work with a similar number of ambient sound classes [13]. Analyses of runtime and energy consumption showed similar results for both modes. In particular, runtime in server mode was $\sim 2$ hours shorter than in autonomous mode for the Galaxy SII, which is explained by the network communication usage. Further analysis revealed that $\sim 80\%$ of the total processing time was spent for feature computation (Framing, FFT and spectrum), where the server mode cannot gain advantages. In contrast, only $\sim 14\%$ of CPU time are required for computing classification re-
results using a SVM. Combined with the communication overhead, the server mode cannot gain advantages in our configuration. However, a server mode implementation could be beneficial if more computational power is needed for more complex classification models (e.g. modelling the MFCC distribution with a Gaussian Mixture Models or Hidden Markov Models). Another advantage of the server mode is the possibility to add crowd-sourced online learning allowing users to upload their own annotated ambient sound samples to improve the auditory scene models or to extend the model set.
Indoor Positioning System for Smartphones *

This chapter presents the design and implementation of RoomSense, a new method for indoor positioning using smartphones on two resolution levels: rooms and within-rooms positions. Our technique is based on active sound fingerprinting and needs no infrastructure. Rooms and within-rooms positions are characterized by impulse response measurements. Using acoustic features of the impulse response and pattern classification, an estimation of the position is performed. An evaluation study was conducted to analyse the localization performance of RoomSense.

*This chapter is based on the following publication: M. Rossi, J. Seiter, O. Amft, S. Buchmeier, and G. Tröster. RoomSense: An Indoor Positioning System for Smartphones using Active Sound Probing. (AugmentedHuman 2013) [89]
6.1 Introduction

Indoor positioning is an essential part of context information and useful for various location-based services that can augment human capabilities, including indoor way-finding in buildings, patient localization, and tour guides [43, 41, 90]. In this chapter we present RoomSense, a smartphone-based system to quickly determine indoor location. Our approach considers standard phones, thus the default smartphone microphone and speaker were used. Using the acoustic impulse response, we recognize room location within a building floor, similar rooms at different building levels, and different positions within a room. We selected 20 rooms and a total of 67 positions according to locations visited in the typical daily life of a university student. In particular, this chapter provides the following contributions:

1. We present the system architecture of RoomSense, which is designed to provide instantaneous indoor position estimates on two resolution levels: rooms and within-rooms positions. We use the Maximum Length Sequence (MLS) impulse response and a Support Vector Machine (SVM) based position recognition techniques to realise RoomSense. We identify the best-performing audio feature sets and further parameters to obtain robust estimates.

2. We evaluate RoomSense in a study comprising of recordings of impulse response measurements from 20 rooms and totally 67 positions using a standard smartphone. Besides performance of room and within-rooms positioning, we vary the number of trained positions per room area. Finally, we evaluate accuracy when the signal-to-noise-ratio (SNR) was reduced.

3. We describe the implementation of RoomSense as an Android app. The app was designed to recognize a room or position within rooms in less than one second. The app can be used to learn new rooms or positions within rooms instantly.

6.2 Related Work

Indoor positioning is an actively researched field. Various approaches have been proposed using additional ambient infrastructure such as sensors or transmitters that were installed in buildings to localize a wearable device [40, 41, 42]. Infrastructure-based methods have a
typical location error of less than one meter. However, a dedicated
technical infrastructure is needed for the localization, which is not
always practical or affordable.

Alternative approaches are using already existing wireless infras-
tructure such as cellular network and Wi-Fi information for the local-
ization task [43, 44, 45]. Here, the signal strength of cellular or Wi-Fi
station is used to determine location of a mobile device and the po-
sition of cells and network stations is known in advance. When using
wireless infrastructure, localization accuracy depends on the density of
cellular/Wi-Fi stations in the environment. E.g., Haeberlen et. al. re-
ported an accuracy of 95% over a set of 510 rooms [43]. However, at
least five Wi-Fi stations were in range at all measurements. The posi-
tioning approach is less suitable where station coverage is unknown or
sparse.

Recently sound-based positioning approaches have been proposed
that require no additional infrastructure to perform indoor positioning.
Passive sound fingerprinting uses ambient sound to generate position
estimates, whereas active fingerprinting approaches emit and then
record a specific sound pattern for the positioning. Wirz et al. [46]
proposed an approach to estimate the relative distance between two
devices by comparing ambient sound fingerprints passively recorded
from the devices’ positions. The distance was classified in one of
the three distance regions \((0 \, m, \, 0 \, m - 12 \, m, \, 12 \, m - 48 \, m)\) with an
accuracy of 80%. However, no absolute position information was
obtained by this method. Tarzia et al. [48] proposed a method based
on passive sound fingerprinting by analysing the acoustic background
spectrum of rooms to distinguish different locations. The location was
determined by comparing the measured sound fingerprint for a room
with fingerprints from a database. A room’s fingerprint was created
by recording continuous ambient sound of 10 s length. The system was
implemented as an iPhone app to localise between different rooms.
The localization performance was high for quiet rooms, but dropped
when people were chatting or when the background spectrum had large
variations. Azizyan et al. [47] used a combination of smartphone sensor
data (WiFi, microphone, accelerometer, colour and light) to distinguish
between logical locations (e.g. McDonalds, Starbucks). Their passive
acoustic fingerprints are generated by recording continuous ambient
sound of 1 \(min\) and extracting loudness features.
We expect an active sound fingerprinting approach to reduce recognition time compared to the passive approach and to be robust against noise. Zhang et al. [91] proposed an approach to estimate the relative distance between two devices with active sound fingerprinting. Their method was tested in a measurement range of 2 m and had a median distance error of 2 cm. So far, only Kunze and Lukowicz presented an absolute positioning approach where active sound fingerprinting was considered [92]. Their system could recognize specific as well as more abstract locations where a phone was placed (e.g. table, floor) when combining information from acceleration and sound sensors. However, no localization on a room resolution level was considered in their work.

Different methods have been proposed to measure room impulse responses however these techniques were not applied for indoor position estimation and pattern recognition. Stan et al. [93] compared different impulse response measurement techniques, including maximum length and inverse repeated sequences, time-stretched pulses and sine sweeps. They considered the room impulse response to be one of the most important acoustical characteristics of a room. Furthermore, the MLS measurement technique showed several advantages compared to the other methods: MLS is perfectly reproducible and immune to various noise types. Furthermore, MLS is deterministic and hence allow summing and averaging of multiple repetitions to improve the signal-to-noise ratio.

In this work, we propose to use room impulse response based on MLS and pattern recognition for indoor positioning on a smartphone at room and within-room position resolutions. Instead of relying on the acoustic background spectrum as in passive fingerprinting, we characterize room acoustics using impulse response measurements.

6.3 RoomSense Architecture

The RoomSense system emits a short acoustic wave and measures the impulse response. This response is further processed as sound fingerprint of a within-room position and eventually the extracted sound features are classified to estimate room and within-room position. The sound pattern models of room positions are derived in a training phase based on annotated acoustic impulse response data. Figure 6.1 illus-
trates the RoomSense system architecture comprising impulse response measurement, front-end processing, and classification components. This section presents the RoomSense system architecture in detail.

![RoomSense Architecture Diagram]

**Figure 6.1:** RoomSense architecture illustrating the main components of the system.

### 6.3.1 Impulse Response Measurement

The first main component of the system is the impulse response measurement for an indoor position. The impulse response is a response of a dynamical system to a Dirac input impulse. It is a time-dependent function. The behaviour of a linear and time-invariant system can be obtained by a convolution of the input signal with the impulse response [94]. Assuming that loudspeaker and microphone setup are motionless, the sound propagation and reflections within a room can be regarded as a close approximation to a linear and time-invariant system [95]. Room impulse responses can therefore be used to completely describe the acoustic characteristics of a position in a room. Common measurement techniques for acoustic room impulse responses are maximum length sequence (MLS), time-stretched pulses, and sine sweeps [93]. For our system we used the maximum length sequence measurement technique as described below.

**MLS Measurement Technique**

The Maximum Length Sequence (MLS) measurement technique is based upon the excitation of the acoustical space by a periodic pseudo-random signal having almost the same stochastic properties as a pure
white noise [93]. Maximum length sequences are binary, periodic signals. They are characterised by their order $M$. The period length of the MLS is $L = 2^M - 1$. A possible method to generate a MLS signal is to use maximal feedback shift register. The shift register can be represented by the following recursive function:

$$a_m[n + 1] = \begin{cases} a_0[n] \oplus a_1[n], & m = 3 \\ a_{m+1}[n], & \text{otherwise} \end{cases}$$

where $\oplus$ denotes the XOR operation. Let the MLS signal with order $M$ be $x[n] = a_M[n]$ and the impulse response of the LTI system be $h[n]$. The output $y[n]$ of the system stimulated by $x[n]$ can be denoted by:

$$y[n] = x[n] * h[n].$$

Since the auto correlation of pseudo-random maximum length sequence $\phi_{xx}$ has approximately the shape of a delta pulse, the room impulse response can be obtained by circular cross-correlation between the determined output signal and the measured input signal. Or in other words: Taking the cross-correlation of $y[n]$ and $x[n]$, we can write:

$$\phi_{yx} = h[n] * \phi_{xx} = h[n],$$

with the assumption $\phi_{xx}$ is a Dirac impulse.

**System Parameters**

For our system, we chose the MLS measurement technique with a common parameter set [95]. The order $M = 15$ was set and the sampling frequency was configured to $f_s = 48 kHz$. A MLS sequence with a length of 0.68 s was played by the loudspeaker and recorded by the microphone with the same sampling frequency $f_s$. The played MLS sequence is hearable as a short noisy sound. With this parameter set a impulse response of the time interval $t = [0, 0.68] s$ and frequency interval $f = [0, 24] kHz$ is generated. Since time-synchronisation between loudspeaker and microphone is not supported by a common smartphone’s hardware, the first arriving impulse - assumed to be the largest peak in the impulse response - is considered to a fixed time $t_{fa} = 45 ms$ within the response. An illustration of a measured impulse response is shown in Figure 6.3.

**6.3.2 Front-End-Processing**

Front-end processing steps aim at extracting position- and room-dependent audio features from the impulse response. Initially, the im-
pulse response is processed in frames with a sliding window with a window size of 32ms and 50% overlap. Each window is smoothed with a Hamming filter. In a pre-evaluation, we found that this framing parameter setting resulted in the largest recognition performance. Similar settings could be found in other audio recognition systems, e.g. in [13]. Subsequently, audio features were extracted for each frame. Common audio features as well as specific room acoustic features have been evaluated (see Table 6.1). The performance results of the feature sets are presented in the evaluation (Section 6.5). Feature vectors $f_i$ were extracted from each frame $i$. In a next step the feature vectors were normalised with $F_i = \frac{f_i - m_i}{\sigma_i}$, where $m_i$ are the mean values and $\sigma_i$ are the standard deviation values of all feature vectors of the training set. After this step, all feature vectors $F_i$ with $i = \{1, 2, ..., n\}$ were concatenated to one feature vector $F_{All} = \{F_1, F_2, ..., F_n\}$. Finally, the Minimum-Redundancy-Maximum-Relevance (MRMR) [96] feature selection was used to select the $M_{sel}$ most relevant features $F_{SEL}$. In our evaluation the number of selected features, $M_{sel}$, was tuned to maximise recognition performance.

<table>
<thead>
<tr>
<th>type</th>
<th>feature names</th>
<th>coef</th>
</tr>
</thead>
<tbody>
<tr>
<td>room</td>
<td>Reverberation Time (T)</td>
<td>3</td>
</tr>
<tr>
<td>room</td>
<td>Early Decay Time (EDT)</td>
<td>1</td>
</tr>
<tr>
<td>acoustic</td>
<td>Clarity (C)</td>
<td>2</td>
</tr>
<tr>
<td>acoustic</td>
<td>Definition (D)</td>
<td>2</td>
</tr>
<tr>
<td>acoustic</td>
<td>Center Time (CT)</td>
<td>1</td>
</tr>
<tr>
<td>common</td>
<td>Auto Correlation Function (ACF)</td>
<td>12</td>
</tr>
<tr>
<td>common</td>
<td>Linear Bands (LINBANDS)</td>
<td>10</td>
</tr>
<tr>
<td>common</td>
<td>Logarithmic Bands (LOGBANDS)</td>
<td>10</td>
</tr>
<tr>
<td>common</td>
<td>Linear Predictive Coding (LPC)</td>
<td>12</td>
</tr>
<tr>
<td>common</td>
<td>Mel-Freq. Cepstral Coefficients (MFCC)</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 6.1: Common audio features and specific room acoustic features considered in the evaluation with their number of coefficients $coef$.

### 6.3.3 Position Classification

The classification aims to generate an estimation of room and within-rooms position based on the generated feature vector $F_{SEL}$. We used the Support Vector Machine (SVM) classifier with a Gaussian kernel [86], which include the cost parameter $C$ and the kernel parameter $\gamma$. 
These parameters were optimized with a parameter sweep as described later in the evaluation section 6.5. The one-against-one strategy was used, which is provided by the LibSVM library [86].

In a training phase, the training set feature vectors including the position and room labels were derived and $SVMTrain$ was used to create pattern models for all rooms and within-rooms positions. In the testing phase $SVMClassify$ used the stored models to classify a new feature vector $F_{SEL}$ regarding room and within-room position.

### 6.4 Evaluation Study

An evaluation study has been conducted to analyse the recognition performance of RoomSense. An impulse response dataset of 67 positions within 20 rooms was collected. The impulse response measurements of our dataset were collected with a Samsung Galaxy SII Android smartphone. Figure 6.2 illustrates the locations of loudspeaker and microphone at the smartphone. Distance between loudspeaker and microphone was 2.4 cm.

**Figure 6.2:** Samsung Galaxy SII smartphone used during the evaluation study. The distance between loudspeaker and microphone was 2.4 cm.
Table 6.2: Overview on rooms and within-rooms positions included for the impulse response dataset. Rooms were selected according to the frequently visited places of a university student, including Work and Home buildings.

<table>
<thead>
<tr>
<th>ID</th>
<th>Rooms</th>
<th>Size $[m^2]$</th>
<th>Positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Work coffee room</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Work corridor 1</td>
<td>65</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>Work corridor 2</td>
<td>65</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>Work entrance 1</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Work entrance 2</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>Work lab</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>Work lecture room</td>
<td>50</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>Work meeting room 1</td>
<td>50</td>
<td>6</td>
</tr>
<tr>
<td>9</td>
<td>Work meeting room 2</td>
<td>25</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>Work office 1</td>
<td>25</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>Work office 2</td>
<td>25</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>Work office 3</td>
<td>30</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>Work toilet</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>Home bathroom</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>Home bedroom</td>
<td>25</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>Home corridor</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>17</td>
<td>Home entrance</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>18</td>
<td>Home kitchen</td>
<td>25</td>
<td>3</td>
</tr>
<tr>
<td>19</td>
<td>Home living room</td>
<td>30</td>
<td>4</td>
</tr>
<tr>
<td>20</td>
<td>Home office</td>
<td>25</td>
<td>3</td>
</tr>
</tbody>
</table>

6.4.1 Recording Procedure and Dataset

Table 6.2 lists the rooms of the compiled impulse response dataset. The rooms were chosen to cover regularly visited rooms of a university student during one working day. Rooms from two different buildings were selected, marked as ‘Work’ (denoting an office building) and ‘Home’ (denoting the participant’s home). Some rooms in the dataset are very similar: work corridor 1 and work corridor 2 have the same floor plan and furniture arrangement, whereas work office 1 and work office 2 have the same floor plan but a different set of furniture.

To investigate within-rooms position estimation, we selected a position for approximately every $9 \, m^2$. It is conceivable that in larger rooms,
the localisation service need to give more detail than in smaller ones. Depending on the size of the room, 2 to 6 recording positions were selected (see Table 6.2). For each within-room position, two orientations were chosen. One orientation was determined by pointing loudspeaker and microphone to the middle of the room. The second orientation was chosen in the opposite direction, thus rotated by 180°.

Since time-synchronisation between loudspeaker and microphone is not supported by a common smartphone’s hardware, the first arriving impulse is fixed at time $t_{fa} = 45\, ms$ (see Section 6.3.1).

All impulse response measurements were carried out with the Samsung Galaxy SII smartphone. During the measurement the smartphone was held with one hand at approximately 1.20 m over ground in an ergonomic posture (see Figure 6.9). Neither hands, other body parts nor objects covered the loudspeaker and microphone (see Figure 6.2). An example of a measured impulse response is shown in Figure 6.3.
Additionally, during all measurements the state of the room has been kept unchanged: Windows and doors were closed, no furniture has been moved, and recording were performed only in quite conditions ($\sim 30$ dB).

For each orientation, 40 measurements were carried out. Since every position has two orientations, 80 measurements per position were gathered. Overall, 67 position within 20 rooms were defined which corresponds to 5360 impulse response measurements.

6.5 Evaluation

We evaluated the recognition performance of the RoomSense system using the impulse response dataset introduced in Section 6.4. This section presents the results of the evaluation. In Section 6.5.1 the recognition performance of different feature sets is compared. The recognition accuracy for the room localization and for the within-rooms position estimation is presented in Section 6.5.2 and 6.5.3, respectively. Furthermore, the effect of noise is analysed in Section 6.5.4. Note that for all evaluation results the SVM parameters $C$ and $\gamma$, and the feature selection parameter $M_{SEL}$ (see Section 6.3) were swept to reach the best recognition performance.

6.5.1 Feature Comparison

Figure 6.4 depicts the room localization accuracy of the system for different feature sets (as introduced in Table 6.1). Positions are classified as one of the 20 rooms (see Table 6.2). The accuracy was computed with a leave-one-sample-out cross validation, where the tested sample is left out from the training set. Using all features the highest recognition accuracy was reached (ALL, 98.7%). A similar result was achieved by the MFCC features (98.2%), followed by the other common audio features LINBANDS (94.4%), ACF (93.5%), LOGBANDS (92.5%), and LPC (82.4%). With the acoustic room features the lowest accuracy was reached (ACOUSTIC_ROOM, 60.3%). Since the MFCC is the best performing feature set, this set was used for the following evaluations and for the app implementation of Room-Sense.
6.5.2 Room Localization

As already presented in the Section 6.5.1 the system’s room localization performance using MFCC features was 98.2%. In this evaluation it is assumed that the system is trained with impulse responses of all positions and orientations. Thus, for a room characterization impulse responses of each tested position and orientation would be needed. For the assumption that the system is trained by an orientation-independent training set, we carried out a leave-one-orientation-out cross validation, where the orientation of the tested sample is not trained. In this case the recognition performance dropped to 85.1%. This performance drop shows that impulse responses depend not only on the measured room but also on the measurement’s position and orientation within the room. Nevertheless, the similarity of room impulse responses within a room compared to other rooms still enables to perform room localization. Figure 6.5 depicts the confusion matrix of this evaluation.

We further analysed the system’s performance for different densities of training positions per room. We varied the number of training positions per room-area from one training position per 9 m² to one training position per 63 m². Additionally, positions of the test samples were never trained (expect for 9 m² where all positions were used for training). Figure 6.6 shows the result of this evaluation. The results show the dependency of recognition accuracy and density of the training positions per room: Performance dropped from 98.2% for one
training position per 9 $m^2$ to 49.8\% for one training position per 63 $m^2$. We conclude that impulse responses are dependent of the position of the measurement equipment. Variations of impulse responses in larger rooms are higher than in smaller rooms. Thus, to reach a room localization accuracy of $\sim 80\%$ at least one position every $\sim 18 m^2$ should be trained.

### 6.5.3 Within-Rooms Position Estimation

In this section the system’s within-rooms position estimation is analysed. Figure 6.7 shows the result in comparison to the room localization performance. For the assumption that all tested positions are exactly known by the system, we trained the system with all 67 positions on both orientations. Tested samples were then classified as one of the 67 positions. We performed a leave-one-sample-out cross vali-
6.5.3 Area with one trained position

Figure 6.6: Room localization performance with different densities of training positions per room. The training position per room-area was varied from one position/9 m² up to one position/63 m².

dation, where the tested sample is left out from the training set. This evaluation resulted in an accuracy of 96.4 %, which is similar to the room localization performance. In a second evaluation, we analysed the orientation-dependency of the within-rooms position estimation. A leave-one-orientation-out cross validation leaving out the test sample’s orientation from the training set was carried out. In this case the accuracy dropped to 51.3 %. We conclude that the orientation-dependency of the impulse response is high. Within-rooms position estimation is possible, however, the orientation of the tested measurement has to be trained in advance.

6.5.4 Noise Robustness

Figure 6.8 shows the noise robustness of the room localization and within-rooms position estimation. Additive white Gaussian noise was added to the recorded maximum length sequence of the tested measurement samples. The recorded maximum length sequence is assumed to be noiseless and SNR was varied between 10 and 50 dB. Since the loudness of the Galaxy SII speakers is 67 dB, an SNR of 50 dB can be compared to the noise of falling leaves, 30 dB to a talking person, and 10 dB to cars on a main road. For both localization levels, a leave-one-sample-out cross validation was performed, were test sample was left out from the training set. Noise robustness is similar for both localization lev-
Figure 6.7: Accuracy of within-rooms position estimation compared to room localization. Leave-one-out cross validation was performed, where either the test sample or the test sample’s orientation was left out from the training set.

The recognition performance constantly drops while decreasing the SNR from 98.2% and 96.4% for an SNR of 50dB to 66.6% and 65.9% for an SNR of 10dB. We conclude that localization is possible (accuracy of > 80%) in environments with an SNR of > 30dB.

Figure 6.8: Noise robustness of room localization and within-rooms position estimation. Measurement samples were corrupted with white Gaussian noise. Noise level was varied between 10 and 50 dB.
6.6 RoomSense Implementation

The RoomSense system was implemented in an Android smartphone setting. The main components (see Section 6.3) were implemented in Java SE 7 and are running on an Android smartphone or PC environment. For the implementation we referred to [97] to implement the MLS impulse response measurement, we used the MFCC implementation of FUNF open sensing framework [87] to derive the features and the LibSVM Library [86] for the SVM modelling and prediction.

Figure 6.9 shows an illustration of the user interface (UI). The app can be used to recognize a location by pressing ’Test Now’: The impulse response measurement is immediately started, the generated signal is processed (front-end-processing), and a location prediction is generated (classification). On the Samsung Galaxy SII the duration of the overall process is about one second whereas IR measurement requires most of the time (0.68 s). Additional the app enables to extend the training set by pressing ’Add to DB’. Training data for new or existing room positions can be recorded and integrated in the room/position models.

![RoomSense user interface (UI) on an Android smartphone and its usage. Both recognizing a location and training new location is possible.](image)
6.7 Conclusion and Future Work

In this chapter we presented a new method for indoor positioning using an active sound fingerprinting approach. After characterising rooms according to the impulse response using acoustic features, pattern classification was used to estimate positions on a room and within-rooms position level.

Our evaluation study showed that our system achieved excellent recognition performance (accuracy > 98\%) for localize a position in a set of 20 rooms with low background noises (> 25) dBA. Our evaluation of different feature sets revealed that MFCC features outperform any other feature group, including the specific room acoustics features in the classification task. Even in a more challenging setting in which a position is localized in a set of 67 within-rooms positions an excellent accuracy (> 96\%) can be achieved. For room localization the positioning performance depends on the density of positions trained. Larger rooms could still be identified (accuracy ~ 80\%), if at least one position is trained for every ~ 18 m$^2$ of room area. Additionally, the orientation of a measurement also effects the performance of room localization. Nevertheless, the similarity of room impulse responses within a room compared to other rooms still enables to perform accurate room localization. For within-rooms positioning the orientation-dependency is higher. If the tested orientation is not trained, within-room positioning performance drops to 51.3\%.

Overall, the sound localization approach presented in this work has large application potential for indoor location-based services as it requires very short measurement times until a robust position estimate can be derived. In our study, less than 1 s was required to obtain the presented estimation performances. Moreover, the impulse response measurement showed robustness against noise. We consider that the short estimation time and noise-robustness can be advantageous over passive fingerprinting approaches. Due to the use of different study conditions, our performance results are however not directly comparable with the passive approach presented by Tarzia et al. [48]. Our active sound localization method may however be unsuitable for continuous use or applications where the user is unaware of the position estimation, due to the hearable probing. Nevertheless, we expect that our method and results open opportunities for indoor location estimation applications using smartphones (e.g. using an ultrasonic frequency range, which is not hearable for humans). The smartphone implementation and system parameters proposed in this work could serve as reference.
Crowd-Sourcing for Daily Life Context Modelling *

This chapter presents an approach to model daily life contexts from crowd-sourced audio data. Crowd-sourced audio tags related to individual sound samples were used in a configurable recognition system to model 23 ambient sound context categories.

*This chapter is based on the following publication:
7.1 Introduction

We investigated an approach to source sound data from the web and derive acoustic pattern models of daily life context. Our approach is inspired by the idea of crowd-sourcing: web audio data is generated by many users. It is heterogeneous, available in large quantities, and provides annotations, e.g. in the form of ‘tags’. However, the web is not a source of perfectly labelled training data. Users generate web audio annotations by following personal interpretation and preferences. In some cases, even erroneous annotations could occur. Thus, web search results include also audio samples with unexpected acoustic content. We refer to these audio samples as outliers. Including outliers in training data affects the quality of the acoustic pattern model and the recognition performance.

We present an approach and system architecture to use data subsets from the open web database Freesound [99] - an audio database consisting of more than 120’000 audio samples freely annotated with tags and uploaded by around 6’000 contributors. To investigate our approach we used an example configuration of 23 sound context categories to derive a recognition system. We demonstrate that the web data can be used to discriminate daily life situations recorded from microphones of commonly available smart phones. We evaluated the system with dedicated recordings of all the 23 categories, and in a study with full-day recordings of 10 participants. We furthermore investigated different automatic outlier filtering strategies and compare them to a manually derived baseline performance.

7.2 Related Work

A common approach to build a recognition system is to manually collect and label training data. Most auditory scene recognition systems used this approach in the past decade. For example, Eronen et. al. [13] focused on recognition systems for environmental sounds, such as “restaurant” or “street”, and Stäger et al. [31] recognized a set of activities of daily living (ADL) based on sound data. Wearable systems for sound recognition have been proposed as dedicated hardware [31] and more recently using smart phones-based solutions, e.g. Lu et al. [39]. However, many activity and environmental sound recognition solutions are yet constraint to small sets of sound contexts and well-defined recording
7.3 Concept of Web-Based Sound Modelling

Our approach is based on context category descriptions, which could be provided by a user. Context category descriptions are used for collecting audio data from the web. Subsequent steps of our architecture include extracting audio features, filtering outliers, and modelling context categories. Filtering the collected audio samples for outliers is essential to derive a robust recognition system. This section details our web-based sound modelling and outlier filtering as shown in Figure 7.1.

Context category descriptions provide a textual description of the set of context categories \( C \). Each category \( c_i \in C \) is described by one or more descriptive terms, which are subsequently used to retrieve sound samples from the web. In this work, we used an example configuration of 23 context categories, listed in Table 7.1. We compiled this set of categories such that a wide range of complex daily life situations were covered, including categories characterizing locations, sounds of objects, persons and animals.

<table>
<thead>
<tr>
<th>Context categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>objects: brushing teeth, bus, car, chair, coffee machine, dishwasher, phone ring, raining, shaver, sink, toilet flush, vacuum cleaner, washing machine</td>
</tr>
<tr>
<td>locations: beach, crowd football, forest, office, restaurant, street, railway station</td>
</tr>
<tr>
<td>animals and persons: bird, dog, speech</td>
</tr>
</tbody>
</table>

Table 7.1: Context category set \( C \). In this work, an example of 23 context categories were used. The category names are directly used as the descriptive terms.
Figure 7.1: Overall architecture of our sound context recognition based on web-collected sound data using Freesound.

Collecting audio data: According to the context category descriptions the sound samples were retrieved from the Freesound database. Sound samples having a set of tags that matches all terms in a category description of our example configuration were downloaded and labelled with the corresponding category. All the retrieved audio samples were transcoded to WAV format with a sampling frequency of \( f_s = 16\, \text{kHz} \) and bit depth of \( b_s = 16\, \text{bits/sa} \).

Extracting audio features: Audio features were extracted from the retrieved audio samples. We used the mel-frequency cepstral coefficients (MFCC), the most widely used audio features in audio classification. These features showed good recognition results for environmental sounds [13]. The feature vectors \( \{s_f\} \) of an audio sample \( s \) were generated by extracting MFCC’s (12 coefficients) on a sliding window of 32\,ms length, with an overlap of 16\,ms between consecutive windows. The same method was used to extract audio features from the smart phone.
Filtering outliers: When collecting audio from the web, outliers with regard to the targeted context category can be expected. Training models with data containing outliers negatively effects the recognition performance of the system. Thus, the goal of filtering is to remove outliers from the correctly labelled data, before models are trained. In our approach to remove outliers, we assumed that correctly labelled samples of a category sound similar than outliers. To measure sound similarity of two sound samples, $s^{(1)}$ and $s^{(2)}$, we used the Mahalanobis distance measure:
\[
D(s^{(1)}, s^{(2)}) = (\mu_{s^{(1)}} - \mu_{s^{(2)}})^T \Sigma^{-1} (\mu_{s^{(1)}} - \mu_{s^{(2)}}),
\]
where $\mu_{s^{(1)}}$ and $\mu_{s^{(2)}}$ are the mean feature vectors of the audio samples $s^{(1)}$ and $s^{(2)}$, and $\Sigma$ is the covariance matrix of the features across all samples. This distance measure has low computational costs and showed competitive results compared to more complex modelling schemes [102]. Based on $D(s^{(1)}, s^{(2)})$, we propose two outlier filtering methods using semi-supervised and unsupervised concepts. Both methods use an approach presented in Algorithm 2, where a filtered set of audio sample set $S$ is created from the retrieved audio sample set $F$. An initial set of correct samples $S_{\text{init}}$ is required. For semi-supervised filtering the initial set must be provided by the user, who selects for each category $k$ correctly labelled samples in $F$. For unsupervised filtering, the initial set is formed by selecting for each category $k$ samples with the smallest inter-category distance. In our evaluation we included the following two methods and used them as comparative baselines: no filtering in which we used all samples in $F$ for training ($S = F$), and manual filtering in which we manually filtered outliers ($S = S_{\text{hand}}$). For manual filtering a sample is listened by a person. If the person does not recognize the sample’s labelled context category (e.g. restaurant) the sample is detected as an outlier.

Modelling context categories: The extracted features of the sample set $S$ were used to train models of the 23 context categories in our example configuration. We separately modelled the feature space of each category $c$ with a Gaussian Mixture Model $GMM_c$. The number mixture components was fixed after a small-scale experiment to 16.

Recognition system: The web-trained GMM models were used to classify audio data recorded from the smart phone. The probability that an audio test sequence $t$ belongs to the category $c$ is calculated by: $p(t|GMM_c) = \prod_f p(t_f|GMM_c)$, where $t_f$ is a feature vector of the test sequence $t$. In our evaluation we varied the length of the test sequence $t$ between 1 and 30 s. As our approach produces a term-based description of the context, it is conceivable that several sound context
Algorithm 2 Outlier Filtering. $F^{\text{cat}}(s)$ is the set of all samples in $F$ belonging to the same category as sample $s$.

**Algorithm 2**

```
inputs: $S_{\text{init}}$, $F$
$S = S_{\text{init}}$
repeat
  for $s$ in $S$ do
    $f = \arg\min_{i \in F^{\text{cat}}(s)} D(i, s)$
    if $\arg\min_{i \in S} D(i, f)$ is $s$ then
      add $f$ to the set $S$
    end if
  remove $f$ from the set $F$
end for
until $F$ is empty
output: $S$
```

could be used simultaneously to describe the situation. An example with the selected set of context categories (see Table 7.1) is when a user is in the location *office* and has a conversation (e.g. *speech*) with his colleagues. Thus, the recognition system generates a top-$n$ classification by selecting $n$ categories with the highest probabilities. Top-$n$ classifiers with $n \in \{1, 2, 3\}$ has been evaluated.

**Performance evaluation:** The system’s recognition performance was measured using the normalized accuracy (mean over all class-relative accuracies). We accounted the classification as correct, if the annotated context category was within the top-$n$ categories.

### 7.4 Results

In total 4678 audio samples (114 hours of audio data) were retrieved from the Freesound database for the 23 context categories. In average, a category had 203 samples. Manually filtering outliers from the samples (as described in the last section in *filtering outliers*) showed that 38% of the samples were outliers. The performance of the system (see Figure 7.1) was analysed within two evaluations. Firstly, we analysed the performance of all category models (see Table 7.1) using dedicated sound data recorded for each context category. Subsequently, we evaluated the performance and system operation in a study using daily life audio recordings from smart phones of 10 participants.
7.4. Results

7.4.1 Evaluation by Dedicated Recordings

The context category models were tested with isolated sound data recorded for each context category using a smartphone (Google Nexus One). Sound samples of at least four different entities per sound category (e.g., four different dishwashers) had been recorded using the phone’s integrated microphone. The samples were recorded in the city of Zurich and in Thailand. For each class, we recorded 6 min of audio data. This test allowed us to assess the system’s performance for the complete set of 23 context categories using self-recorded data and compare the benefit of the different outlier filtering methods. Moreover, the goal of this test was to confirm that the microphone and electronics of a commonly available smartphone suit for the recognition of daily life contexts.

Figure 7.2 shows the recognition accuracy of the top-1 classifier for the different outlier filtering methods. The length of the test sequence has been varied between 1 and 30 s. As expected, increasing the length of the test sequence improved the overall recognition accuracy. Models trained with no outlier filtering performed at lowest accuracy (38% with a 30 s test sequence). Using unsupervised and supervised filtering, the accuracy increased to 46% and 53%, respectively. The best performance was reached using manual filtering (57%). These performance results are comparable to studies considering similar large number of sound categories, e.g., the work of Eronen et al. [13]. In their work, the authors used an audio-based recognition system for 24 environmental categories and obtained a recognition performance of 58% using MFCC features. Their training dataset was compiled manually from dedicated recordings and included few locations only. In contrast, the web-based audio data consists of diverse field recordings acquired using different recording systems.

Figure 7.3 shows the confusion matrix of the top-1 classifier trained on data filtered with the semi-supervised method and using a test sequence length of 30 s. The category beach, railway station, and speech showed the best class-relative accuracies (100%). In contrast, the category vacuum cleaner was not recognized by the system. For the vacuum cleaner, semi-supervised filtering failed to remove outliers. Some confusions could be explained by the similar context in which the sounds were recorded, e.g., restaurants was confused with speech and bus. All the three categories included recordings of talking people. Brushing teeth was confused with shaver, since some electronic toothbrushes
and shaving machines produced similar sounds.

Using a top-3 classifier on the same data set with a test sequence length of 30 s the recognition accuracy improved for all four methods: 51% without any outlier filtering, 69% with unsupervised filtering, 79% with semi-supervised filtering, and 80% with manual filtering.

### 7.4.2 Evaluation of Daily Life Study

To investigate the web-based recognition approach in real-life data, we performed a study using smartphones for continuous environmental sound recording of 10 participants aged between 24 and 40 years. Participants were asked to record two full working days in one week. Recordings were done using the same phone model as in Section 7.4.1 but with a headset microphone. During the recordings participants attached the headset to the upper body clothing between waist and collar. The recordings had been performed using our specialized Android application “AudioLogger”. The application allowed us to store continuous audio data on the SD card of the smart phone. In addition, the application provided an annotation tool in which the user could select current contexts from a selection list providing all context categories shown in Table 7.1. For each recording day at least 8 hours of audio data were obtained. In total, more than 230 hours of audio data were collected in this study.

During the study, participants used only a subset of the annotations provided. The categories used by all participants were: bus, car,
7.5 Conclusion

Using web-collected audio data to construct a context recognition system showed to be a promising approach. It provides opportunities to
reduce the process of manually collecting training data as it is available in large quantities from the web. However, this investigation also showed that web-mined information can be ambiguous or even wrong. No strong semantical rule exists in the web, which defines how data should be described or tagged. Nevertheless, our results showed that the retrieved web information can be used for context modelling: the proposed outlier filtering methods yielded a recognition accuracy increase of up to 18%. Practical recognition rates for high-level contexts between 51% and 80% could be achieved. We expect that the presented recognition system could be implemented on a cloud server to operate in real-time, as introduced in Chapter 5. Based on such a mobile system, the idea of crowd-sourcing could be extended giving the users an opportunity to share personal auditory scenes directly using their smart phone.

**Figure 7.4:** Recognition performance of the web-based recognition in daily life recordings. The recognition system was trained on the 23 context categories, however, tested only on the participant’s used seven categories (see Section 7.4.2). The test sequence length has been set to 30 s.
This chapter introduces a daily life context diarization system which is based on audio data and tags from a community-maintained audio database. We recognised and described acoustic scenes using a vocabulary of more than 600 individual tags. Furthermore, we present our daily life evaluation study conducted to evaluate the descriptiveness and intelligibility of our context diarization system.

*This chapter is based on the following publication:
8.1 Introduction

Existing context recognition systems that attempt to describe daily life situations often suffer from two specific constraints: firstly, it is laborious to obtain sufficient amounts of training data, representative for the targeted daily life situations. The fundamental challenge underlying this constraint is the heterogeneity of activities and environmental variability that could be observed. As an example, the acoustic scene of a street is composed of various intermixed sources, including car, busses, trams, footsteps, or even tweeting birds. Secondly, the description of a particular moment in time may involve semantic and intelligibility issues related to differences in terminology, culture, and personal preferences. In classic recognition systems, designers select the terms and context classes based on the sounds modelled, which can limit context descriptiveness. To address the modelling and intelligibility challenges, context recognition could be based on extensible, open databases that not only provide sensor data but also suitable descriptions, i.e. tag words, that could be used to construct context diaries. Besides the absolute system recognition performance, descriptiveness and intelligibility should be assessed in such a context diarization system to evaluate the information that is accessible to users.

In this chapter we present a smartphone-based daily life context diarization system that uses crowd-sourced audio data and tags to provide a comprehensible textual description of activities and environmental situation. Instead of directly classifying environmental sound from a once-recorded dataset of sounds, our approach is based an open, community-maintained audio database that provides crowd-sourced audio tags to describe sounds. By using the databases’ sound and tagging information, we recognised and described acoustic scenes using a vocabulary of more than 600 individual tags. This work provides the following contributions:

1. We present the concept and system architecture of the context diarization system. We show how the audio database tag vocabulary is derived and how large sound-derived feature sets could be used for context modelling and recognition.

2. We present an evaluation study of continuous environmental sound recordings and context annotations of 16 participants during their daily life. Here, we measure system performance in post-recording analyses by (1) analysing participant ratings of the gen-
8.2 Related Work

An approach to overcome the heterogeneity challenge is to mine information from the web. Perkowitz et al. [100] presented a first approach for activity discovery using web-based data sources. The authors extracted information from textual descriptions in large “How-to”-databases to model human activities. These databases provided step by step descriptions for various activities ranging from “How to make a tea” to “How to Ice Skate Backwards”. Activities were then modelled as dynamic Bayesian networks and reused for activity recognition with RFID-tagged objects. Wyatt et. al. [105] extended this work by mining the web for new activities instead of specific how-to websites. Pentney et al. [106] used the web to extract common sense information and used this knowledge to improve sensor-based recognition of daily life activities. Zheng et al. [107] described how labelled data of an activity in one domain (e.g. hand-washing-dishes) can be reused for a similar activity in another domain (e.g. hand-washing-laundry). Similarities between activities were learned by mining knowledge from the web. A similar methodology was used for image recognition too: Ferrari and Zisserman [108] showed how Google image search can be used to automatically learn specific object or persons. Nevertheless, these
works did not mine sensor data, or more specifically, acoustic contexts from the web.

Characterizing person’s context during his daily live can be done at different abstraction levels. One approach is to use unsupervised clustering methods on the continuous audio data, as e.g. in [109, 110, 111], where audio data is divided in segments representing specific context of person’s daily life. However, to describe such audio segments, a defined set of sound classes and annotated training data is needed. Furthermore, describing daily life context, such as “office work” and “conversation” using mutual excluding categories can be insufficient to fully describe the context. In contrast, our approach is to characterize the environmental sound based on audio community tags. Here, we exploit the online crowd-sourced audio database “Freesound” by using the tag vocabulary and the associated audio data created by the vast database user community. Thus, acoustic contexts could be recognised using a rich set of descriptive tags.

Freesound was used for information indexing and retrieval before. Chechik et al. [53] and Roma et al. [112] presented retrieval systems that indexed sound samples based on the sample’s audio content rather on the sample’s meta data. Chechik et al. modelled audio data by the individual audio tags, whereas Roma et al. used a taxonomy from ecological acoustics for modelling and retrieving. These content-based retrieval systems were extended by taking the semantic similarities of tags into account [113]. Wichern et al. [114] and Martínez et al. [115] exploited Freesound for automatic tagging of audio samples. Tagging was done by using an audio similarity measure to compare tagged audio samples with non-tagged samples. In our previous work (see Chapter 7), we presented an approach for recognising daily life context based on data sourced from Freesound. The mined sounds were used to model context classes and recognition performance was analysed. However, the recognition system was limited to 23 manually selected sound classes. In our present work, we introduce the concept of a context diarization system, able to produce continuous tag-based descriptions of the recognised context. Moreover, we evaluate our approach by assessing intelligibility of the system’s output.

8.3 Context Diarization based on Community Tags

In this section we present our context diarization system based on crowd-sourced audio data and tags. In particular, we detail the sound
tag modelling based on sourced sounds and describe how tag models were used to generate textual diaries from environmental sound recordings done with smartphones. Figure 8.1 depicts the architecture of our system, consisting of the tag sound modelling using Freesound and the context diarization components.

**Figure 8.1:** Architecture of our context diarization system. Our approach is based on audio data and tags sourced from the Freesound database. Our system consists of a context modelling component (*tag sound modelling using Freesound*) to create tag models, and a diarization component (*context diarization*) to create a textual sound diary from smartphone recordings.
8.3.1 Tag Sound Modelling using Freesound

The web offers various sources for sound samples and associated descriptions that could be used for context recognition purposes. While our implementation described below makes use of Freesound, the approach can be used with other sources that link sound samples and textual metadata.

The Freesound Project [99] is a collaborative database that aims at creating a large repository for non-music sound data, including audio snippets and recordings. Audio data is associated with descriptions and user defined tags. The database provides a wide range of audio data qualities (regarding sampling rate and bit depth), and coding (including MP3, WAV, and FLAC). The data is released under the “Creative Commons Sampling Plus License” [116]. In the modelling phase of our approach, Freesound is exploited to create the diarization tag vocabulary and the corresponding models from sound data, which are further used in the context diarization phase to create textual diaries (see Figure 8.1).

Creating Tag Vocabulary

The vocabulary $V_d$ formed the set of tags which were further modelled in the audio domain and then used for textual diarization. The aim of this component was to select meaningful tags from the crowd-sourced tag vocabulary $V_f$ which had a sufficient representation in the audio domain and discard all other tags. The creation of the vocabulary $V_d$ included the following steps:

1. All tags $t_i \in V_f$ used to annotate less than 200 sound samples were discarded. This step ensured that enough audio data was available to model the tag in the audio space.

2. Remaining tags used more than 2000 times to describe sound samples were discarded. This step ensured that general tags like ”field recording” were not used for the textual diarization.

3. Remaining tags were transformed to lower case and tags including numbers were removed. Additional, hyphens in tags were replaced with a space character.

4. In a last step we discarded all tags not included in the WordNet. WordNet is a large lexical database of English which is further described in Section 8.5.2.
Collecting Audio Data

This system component collected data from Freesound based on the diarization tag vocabulary $V_d$. For each tag $tg$ in the vocabulary, audio samples $s_i^{tg}$ annotated with $tg$ were downloaded:

$$\hat{S}_{raw} = \{s_i^{tg} | tg \in V_d \text{ and } i = \{1, 2, \ldots N_{tg}\}\}$$

where $N_{tg}$ is the number of audio samples annotated with the tag $tg$. Samples shorter than 1 s were not downloaded since the included environmental sound information is constricted. For the download we used the RESTful API provided by the Freesound platform\(^1\). The remaining recordings were transcoded using the FFmpeg tool\(^2\) to WAV format with a sampling frequency of $f_s = 16$ kHz, a bit depth of $b_s = 16$ bits/sa and reduced to one audio channel (mono), resulting in a retrieved audio sample set $S_{raw}$.

Audio Fingerprinting

The component audio fingerprinting aims at compressing the raw audio samples by extracting audio features characterizing the specific tag and discard tag-unrelated information. A considerable work on the design of acoustic features for sound classification can be found in the literature (for a review see [50]). Typical feature sets include both time- and frequency-domain features, such as spectral skewness, energy envelope, harmocity and pitch. The most used audio features in audio classification are the Mel cepstral frequency coefficients (MFCC). MFCCs showed to be a sufficient representation for environmental sound classes [13], and in some cases adding additional features did not improve the classification accuracy cases [117].

In this work we chose to use MFCC feature extraction. For each audio sample $s \in S_{raw}$ MFCC features vectors (12 coefficients, without the (first) energy coefficient) are extracted together with their first and second derivatives on a sliding window $w$ of 32 ms length, with no overlap between consecutive windows. This resulted for each audio sample $s$ in a set of 36-dimensional feature vectors $M^s$:

$$M^s = \{m^s_w | w = \{1, \ldots, N^s_w\}\}$$

\(^1\)Freesound RESTful API: http://www.freesound.org/docs/api/

\(^2\)FFmpeg: http://www.ffmpeg.org/
where $N^s_w$ is the number of sliding windows of the audio sample $s$. For the extraction we used the Matlab toolbox Voicebox\(^3\) with the given standard parameters.

Typically these feature vectors were directly used for modelling sound classes, e.g. by generating a Gaussian Model Mixture (GMM) for a tag $tg \in V_f$ based on the combined set of feature vectors of all audio samples \([13]\). However, this modelling technique does not scale up for large data sizes (see \([53]\)). Thus, we further reduced feature data of an audio sample $s$ to a single sparse feature vector $r^s$ using the concept of *acoustic words representation* as described in \([53]\): We clustered the feature space of the MFCCs feature vectors by using k-mean clustering with $k = 2048$. Clustering is done on a randomly chosen subset of all feature vectors extracted from the audio samples $s \in S_{raw}$. The generated centroids $C = \{c_1, c_2, \ldots, c_k\}$ are then treated as *acoustic words* and each audio sample $s \in S_{raw}$ is viewed as a *bag of acoustic words*. More precisely, each feature vector $m^s_w \in M^s$ was mapped to the nearest acoustic word $c_{map}$:

$$c_{map} = \arg \min_{c \in C} d(c, m^s_w)$$

where $d(c, m^s_w)$ is the Euclidean distance between the centroid and the feature vector. An audio sample $s$ can then be represented by the distribution of the acoustic words:

$$\hat{f}^s = [tf^s_{c_1}, \ldots, tf^s_{c_i}, \ldots, tf^s_{c_k}]^T$$

where $tf^s_{c_i}$ is the number of occurrence of acoustic word $c_i$ in audio sample $s$. We normalized this vector by the term frequency-inverse document frequency (tf-idf) normalization used in text mining \([118]\), resulting in the final audio fingerprint of an audio sample $s$:

$$f^s = [n^s_{c_1}, \ldots, n^s_{c_i}, \ldots, n^s_{c_k}]^T, \quad \text{with } n^s_{c_i} = \frac{tf^s_{c_i} \cdot idf_{c_i}}{\sqrt{\sum_{c \in C} (tf^s_c \cdot idf_c)^2}}$$

The term $idf_c$ is the inverse document frequency of the acoustic word $c$ defined as $-\log(r_{at_c})$, with $r_{at_c}$ being the fraction of all samples $s \in S_{raw}$ containing at least one occurrence of the acoustic word $c$. The tf-idf normalization weights acoustic words frequently used in many audio samples smaller than words found in just a few sound samples. The idea behind this normalization is that words used only in a small set of sound samples characterize this samples better than words frequently used in many samples.

\(^3\)Voicebox: http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html
8.3. Context Diarization based on Community Tags

Sound Modelling

For modelling the tags $tg$ based on the extracted audio fingerprints $S_{fin}$ we used Support Vector Machines (SVM) [119]. The advantage of SVM is the efficient model capability of sparse vector data (e.g. the audio fingerprints) and thus, more scalable compared to GMM’s: In [53] SVM showed similar classification performance as GMM for environmental sounds, however, for SVM the creation of the models was up to 190 times faster than for GMM.

Each tag $tg$ in the vocabulary $V_d$ was modelled with a separate SVM with a linear kernel using the LibSVM library [86] implementation. The SVM for a tag $tg$ was trained with the one-against-all strategy: all audio fingerprints of audio samples relevant to the tag $tg$ were used as positive examples, and all others as negatives. To be able to compare the predictions of each SVM we additionally trained a probability estimator proposed in [120]: The SVM alone only classifies a test sample $s_{test}$ belonging to the tag $tg$ or not, however, using the estimator we get a score $(score(svm_{tg}, s_{test}))$ ranging from 0 to 1 how well the test sample matches to the tag $tg$. Finally all trained SVM $D = \{svm_{tg}\}$ are stored in the tag sound model database.

8.3.2 Context Diarization

In a context diarization phase the tag vocabulary $V_d$ along with the sound models $T = \{svm_{tg}\}$ were used to create a textual diary based on users’ daily life continuous environmental sound data. Audio data recorded with a mono microphone at a sampling frequency of $f_s = 16$ kHz and a bit depth of $b_s = 16$ bits/sa was used.

Segmentation and Audio Fingerprinting

We used a sliding window of 4 min length and no overlap between consecutive windows to segment the data. This window length enabled to capture interesting events in user’s daily life ensuring enough data for testing. For each segment $u \in U_{seg}$ an audio fingerprint $f^u$ was created. The same method for audio fingerprinting as in the modelling phase along with the same set of acoustic words $C$ was used (see Sec. 8.3.1).

Automatic Tagging

The automatic tagging maps to each segment $u \in U_{seg}$ a fixed number $n$ of tags $tg \in V_d$. To select the tags for a segment $u$ we calculated
score\((svm_{tg}, f^u)\) for all models \(svm_{tg} \in D\) (see Section 8.3.1) and selected the set of tags \(TG^u = \{tg_1, tg_2, \ldots, tg_n\}\) with the highest scores. Tags appearing in consecutive audio segments were merged together to longer segments (see an example with \(n = 3\) in Figure 8.1). This procedure generated the final textual diary of a user’s daily life, consisting of \(\leq 15 \cdot n\) tags for each recorded hour.

8.4 Daily Life Evaluation Study

We evaluated our context diarization system in a daily life evaluation study. The purpose of this study was to analyse the diarization system’s intelligibility to describe daily life context from sound recordings. We captured continuous environmental sound of users during their working days. After the recordings we generated the textual diary with our system. To evaluate the quality and descriptiveness of the automatically generated diary, we used two methods: participants’ feedback analysis and the WordNet analysis. In this section we describe how the data was collected.

8.4.1 Audio Recordings

We recruited participants to collect continuous environmental sound during their working days. Before the recordings, participants received oral and written information about the recording study, procedure, and study goals. After agreeing to participate, participants were asked to record two working days in one week.

In total 16 participants (5 females and 11 males) aged between 24 and 60 years were recruited. Recordings were performed in Zurich, Switzerland and Eindhoven, the Netherlands. In total, more than 360 hours of audio data were collected and annotated by the participants.

Recordings were performed using an Android phone (Samsung Galaxy SII), where the headset microphone was used to capture participant’s environmental sound. We implemented an Android app “SenseLogger” for this study (see Fig. 8.2). The app enabled to store audio and location data to the SD card of the smartphone. Before starting recordings, participant details were entered (see Fig. 8.2(c)). During the recordings, participants could start and stop audio capturing and monitor the storage usage (see Fig. 8.2(a)). Audio data was stored with a sampling rate of \(f_s = 16\) kHz and bit depth of \(b_s = 16\) bits/sa. A 16 GB
SD card was sufficient to store multiple day recordings. SenseLogger continuously recorded audio, even if the app was not in foreground or the display was turned off. The battery lifetime of the phone while recording was \( \sim \)8 hours. To ensure a continuous audio recording during the day, participants were asked to recharge the phone, e.g. when working with a computer.

The recordings were carried out as follows: after getting dressed in the morning, participants attached the headset to the upper body clothing between waist and collar, and start the SenseLogger app (see Figure 8.3). During the entire day, participants could annotate their activities and environmental contexts using the app as described in the next section. In the evening, after getting home from work, the recording was stopped by the participants. Participants could stop the recordings at any time during the day. For each recording day, at least 8 hours of audio data were recorded. After the two-day recording, the recognition system was used to generate the textual diary based on the captured environmental sound data.

### 8.4.2 Context Annotation

In addition, the app provided an annotation tool to describe the context during recordings (see Figure 8.2(b)). A standard list of context categories was provided (see Figure 8.3(b)), which were used to describe context by the type of location (e.g. home), for journeys by the type of transportation (e.g. car), and social interactions (e.g. in a conversation). Participants could add further annotations, if none of the initial ones suited a particular situation. Moreover, we provided participants with an option to add freetext comments in the app. Context categories could be activated by clicking on the item on the list. Activated categories were marked with a check and remain activated until the items were re-clicked. More than one context category could be activated at a time (e.g. office and conversation). The participants were asked to use at least one context category that described their current situation best. These user annotations were later used during participant’s feedback and evaluation. Furthermore, participants were encouraged to make detailed annotations, either by specifying an additional context category for repeated use or adding comments. This additional annotation support was an important tool helping participants to remember the recorded events in the feedback phase (see Section 8.5.1).
8.4.3 Dealing with Privacy Issues

Acquiring raw audio data of a person’s daily life involves critical privacy issues. In our study, the participants’ environmental sound data could include sensitive information e.g., private conversations. We addressed this problem by randomly permuting audio frames of 32 ms within a section of 1024 ms, immediately before storing them to the SD card. Since audio fingerprinting (see Sec. 8.3.1) had the same frame length, this approach did not effect recognition. It should be clarified that the original frame ordering may still be recoverable by testing all possible combinations of frames in a section. Nevertheless, the permutations were sufficient, such that recorded conversations and other activities could not be retrieved from listening to recordings.

8.5 Evaluation Approach

Using the data collected from our daily life evaluation study (see Section 8.4) we measured the intelligibility of the diary to describe daily life
8.5. Evaluation Approach

Figure 8.3: The SenseLogger app for audio recording and annotation options. (a) recording setup, (b) list of initial context categories used for annotation.

context. Here we detail the two methods, which were used: participants’ feedback analysis and WordNet analysis. Furthermore, we present an analysis of the Freesound database, characterizing the structure of the audio data and tag vocabulary as it was available for our evaluation.

8.5.1 Participants’ Feedback Analysis

We asked all participants of the daily life evaluation study to rate the descriptiveness of their diaries as they were generated by our system. To ensure high recall, participants were asked one day after the study data recording to complete feedback forms. The feedback consisted of a detailed tag rating and an overall rating of the textual diary.

Detailed Tag Rating

For their ratings, participants received the generated textual diary together with their annotations presented in a web application (see Section 8.4.1). Figure 8.4 shows the user interface with diary and annotations in a scalable time line. The Web application allowed participants to rate the generated tags. Each tag rating consisted of the following

<table>
<thead>
<tr>
<th>Context categories</th>
<th>Locations</th>
<th>Transports</th>
<th>Interactions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>restaurant</td>
<td>bus</td>
<td>conversation</td>
</tr>
<tr>
<td></td>
<td>home</td>
<td>car</td>
<td></td>
</tr>
<tr>
<td></td>
<td>kitchen</td>
<td>train</td>
<td></td>
</tr>
<tr>
<td></td>
<td>market</td>
<td>tram</td>
<td></td>
</tr>
<tr>
<td></td>
<td>office</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>street</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>toilet</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table: Initial context categories used for annotation.

(a) recording setup, (b) list of initial context categories used for annotation.
information: start and end time of the segment, the tag word, and the corresponding user rating. Additionally, participant comments were depicted in bullets at specific times when they were entered. Participants could browse through the diary by shifting the time line with the mouse pointer. The user interface was implemented for this daily life evaluation study using HTML5, JavaScript, and the Timeline framework.

While reviewing the generated tags, participants were asked to rate all tags in the textual diary by clicking on a tag (see Fig. 8.4(b)). Table 8.1 shows the rating scale used. In addition, participants were asked to revise their annotations: During the recordings, participants sometimes forgot an context annotation or annotated with the wrong context category. In the feedback phase they had the opportunity to correct wrong or missing information by using the edit feature in the user interface (see Fig. 8.4(c)). The annotation was needed for the WordNet analysis presented in Section 8.5.2. The corrected annotations and tag ratings were sent to a server. For further evaluation, each tag segment was split again into 4 min segments (as generated by the system earlier, see Sec. 8.3.2). These tags are further denoted as tag samples.

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>not understandable</td>
<td>tag is a word which is unknown by the participant</td>
</tr>
<tr>
<td>1</td>
<td>not occurred</td>
<td>tag describes a sound or an event which did not occur at this time.</td>
</tr>
<tr>
<td>2</td>
<td>similar sound</td>
<td>tag describes a sound or an event which sounds similar to a occurred event (e.g. raining and showering).</td>
</tr>
<tr>
<td>3</td>
<td>may occurred</td>
<td>tag describes a sound or an event which the participant is not sure if it really happened at this time (e.g. an announcement in a train).</td>
</tr>
<tr>
<td>4</td>
<td>occurred</td>
<td>tag describes a sound or an event which happened at this time.</td>
</tr>
</tbody>
</table>

Table 8.1: Rating scale used by the participants to rate the descriptiveness of a tag in the textual diary.

4Timeline Widget: http://www.simile-widgets.org/timeline/
8.5. Evaluation Approach

Figure 8.4: Screenshots of the Web-based diary review and feedback application used by participants to browse through the textual diary and annotations. (a) Main view with tags of the textual diary. The upper bars (blue) denoted the tags of the diary, whereas the lower bars (red) denoted participant’s annotations. (b) Rating options popup. (c) Annotations refinement popup.

Overall Rating of the Textual Diary

Participants were further asked to complete a questionnaire form after the study completed. Questions were asked about their opinion of the overall textual diary (e.g. overall tag descriptiveness), the descriptiveness of tags in specific locations (e.g. office, street) and which tags did they miss in the diary. Additionally, some person specific information were asked (e.g. participant’s sex and age). The results of the participant’s feedback analysis is presented in Section 8.6.1.

8.5.2 WordNet Analysis

As a second analysis method to evaluate the textual diary, we used WordNet. We analysed how the tag words included to the textual diary fit to the participant’s context annotations using WordNet’s word similarity measurements.
WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations [104]. WordNet is freely and publicly available for download and it is used as a tool for computational linguistics and natural language processing (e.g. for text mining [121]).

We used the similarity metric \( \text{lin} \) to measure the word similarity between a generated tag \( (tg) \) in the textual diary and the corresponding participants annotated context category \( (cat_i) \). The \( \text{lin} \) metric is based on the information content of the terms and their ancestors within a corpus (see [122]).

Since the context categories are very broad terms (e.g. “restaurant”) we defined three words to describe each category: additionally to the context category label itself, two other words were defined, e.g. for the category “restaurant” the words “restaurant”, “converse”, and “bar” were used. The two additional words are defined by selecting the two participants’ highest rated tags in each category. Finally, all three words are used to measure the relatedness between a tag and a concept category. The similarity between a tag \( (tg) \) and a context category \( (cat_i) \) was defined by:

\[
s_{\text{lin}}(tg, cat_i) = \max_{c \in cat_i} s_{\text{lin}}(tg, c)
\]

The metric \( s_{\text{lin}} \) has a range from 0 for lowest similarity to 1 for highest similarity. The results of the WordNet analysis is presented in Section 8.6.2.

8.5.3 Freesound Database Analysis

For our analysis, we took a snapshot of the Freesound database during May 2012. An overview on data, tag vocabulary, and final statistics of the modelling procedure is presented here.

Statistic of the Freesound Data

The database consisted of 136'615 audio files corresponding to \( \sim 75 \) days of continuous sound data or \( \sim 840 \) GB of stored data. Audio samples had in average a length of 48 s ranging from less than one second to 7 hours. More than 6200 individual users contributed to the Freesound database by uploading their annotated audio samples. At least one sample and maximum 2828 samples per contributor were uploaded. In average contributors uploaded 21 samples. The Freesound
tag vocabulary $V_f$ (see Figure 8.1) consisted of 37,423 individual tags. However, 17,573 tags have been used to describe only one audio sample. In average audio samples were annotated with $\sim 6$ tags whereas 90% of the samples had less than 10 tags. The three most used tags were *field-recording*, *drum*, and *multisample*, each used more than 9000 times to describe a sample.

**Tag Vocabulary Analysis**

For a more detailed analysis concerning the descriptiveness of the tag vocabulary we used WordNet. Around 30% of the tags in the vocabulary were not in WordNet. This set consisted of non-English words, misspelled words or abbreviations and thus not interesting for our diarization system. We analysed the remaining tags with WordNet using two methods: (1) by analysing which type of words (e.g. noun, verb, adjective or adverb) were used as tags and (2) by categorizing each tag in five general classes similar to [115]: *artefact or object*, *organism or being*, *action or event*, *location*, and *attribute or relation*. For this categorization we used the WordNet word similarity measure “path” [122] which represents the semantic relatedness of word senses. Each tag is mapped to the category having the highest similarity. Figure 8.5 shows the result of both analysis. Mostly nouns were in the vocabulary which were used to describe audio samples by sound sources (e.g. ”rain”, ”men”) or the recording location (e.g. ”office”, ”street”), followed by verbs to describe actions (e.g. ”accelerating”), and adjective to describe the sound itself (e.g. ”noise”, ”quietly”). The categorization showed that 33% of the words are objects/artefacts, whereas 17% are organisms/beings. 24% of the words were used to describe attributes/relations and 18% for actions/events. Only 8% were used to describe locations.

**Statistics of the Tag Modelling**

During tag modelling (see Sec. 8.3.1), 625 tags were selected for the tag vocabulary $V_d$. For this set of tags a retrieved audio sample set $S_{raw}$ with $\sim 121,000$ audio samples and a data size of $\sim 720$ GB of audio data was downloaded. This corresponded to 88% of Freesound’s total data size. Some audio samples were used for more than one tag in the tag vocabulary $V_d$. After MFCC feature extraction (see Section 8.3.1) of all audio samples in $S_{raw}$ the data size was decreased to 3%, however still more then 25 GB. Finally, the fingerprints $f^* \in S_{fin}$ of all audio
samples reduced data size to 920MB which is 0.3% of the corresponding MFCC feature data set.

8.6 Results

This section details the results of our daily life evaluation study. In Section 8.6.1 the results of the participant’s feedback evaluation are presented. Section 8.6.2 shows results of our WordNet analysis.

8.6.1 Results of the Participants’ Feedback Analysis

We received feedback from all 16 study participants, including detailed tag rating and questionnaire responses. In total, 16564 rated tag samples were received, with the rating scale presented in Section 8.5.1. Figure 8.6 shows the overall results of the tag rating as a histogram. According to the participants, the majority of tag samples (63%) described an occurred event in their daily life (occurred), whereas 5% were rated as may have occurred, 7% as tag describing an event with a similar sound, 22% as not occurred, and 3% of the samples were not understandable for the participants. Thus, for the participants 25%
of the tag samples (samples rated as *not occurred* and *not understandable*) did not describe their daily life in a meaningful way, whereas 75% (samples rated as *similar sound*, *may have occurred*, and *occurred*) were descriptive. The average rating over all tag samples was 3.03, indicating that a majority of tags described context that occurred or may have occurred. We interpret this result as a positive performance of the system in describing daily life context.

\[\text{Figure 8.6: Participant tag ratings. The dashed line denotes the mean rating value, indicating that a majority of tags described context that occurred or may have occurred.}\]

In the evaluation study, out of the 625 tags \(tg \in V_d\) only 185 tags were used by our diarization system. Figure 8.7 shows a word cloud of the used tags. The font size of each tag denotes how frequent the tag was used in the study (e.g. number of samples of the tag): tags with large font sizes are more frequently used by the system. The grey scale of the tag represents participants’ average rating. The most frequent used tags are "quiet" and "converse". Both tags have in average a high rating (3.91 and 3.94). Other frequently used tags were "restaurant" and "city ambiance" showing lower average ratings (2.2 and 2.1). These tags were sometimes used by the system in wrong daily life situations, e.g. the tag "restaurant" appeared in situations where a group of people were chatting, whereas "city ambiance" appeared in situations where the participant was in a noisy office with an open window. Some infrequently used tags like "low" or "restful" had low ratings since these tag descriptions were not understood by participants.
Figure 8.7: Word cloud of all used tags. The font size represents the tag’s frequency used by the recognition system. The grey scale represents the tag’s average participants’ rating. For readability hyphens were added to tags with more than one word.

We further evaluated how well the tags described different contexts in daily life. For this evaluation we used the twelve context categories which participants used for annotating their recordings (see Table in Fig. 8.3). Each tag sample was mapped to one category according to the participant’s annotations, e.g. tag samples used to describe situations where the participant was in the office are mapped to the office category. Figure 8.9 depicts the tag rating histograms for each context category. Depending on the context category, participants’ tag ratings varied. Tags in the context category market had the lowest ratings (in average 2.1). The word cloud of this context category (see Figure 8.8(a)) showed that in total only 24 different tags were used for context description. Tags like ”market”, ”converse”, ”public”, and ”speak” were rated as occurred tags, however the majority of the tags were not descriptive in this context. Other context categories showed average tag rating higher than 3.5: car, conversation, restaurant, and street. In each of these categories more than 60% of the tag samples were rated as occurred. As an example, the word cloud of category street is depicted in Figure 8.8(b). Compared to the category market a larger set of 92 different tags was used to describe the context. In this category more tags were rated as occurred describing the situations in the individual street contexts: describing the sound itself (e.g. ”street-noise”, ”rattling”), sound sources (e.g. ”cars”, ”tram”, ”passengers”), or locations (e.g. ”city centre”, ”downtown”). The categories describing public transportations (train, tram, and bus) showed average ratings of 2.7 to 3. Compared to others, these
categories had higher numbers of samples rated as similar sound or may have occurred. Tag samples rated as similar sound were often describing sounds of another public transportation context category, e.g. ”passenger train” in the category tram. Additionally, for these categories it was difficult for the participants to remember every detail (e.g. ”announcement” in the category train) and thus, rated these tags as may have occurred.

Figure 8.8: Word cloud comparison of two different context categories. (a) market, (b) street. Size and color tone of the tags were coded as in Figure 8.7. For readability, hyphens were added to tags with more than one word.

In the feedback questionnaire participants were asked to rate the descriptiveness of the overall textual diary and six individual context categories. A rating scale from 1 to 4 (not descriptive, marginal descriptive, descriptive, and very descriptive) were used. The overall impression of the participants given by the questionnaire matches the results of the detailed tag rating presented above. Overall rating (all categories) was 3.3 (see Figure 8.10). Concerning individual context categories, participants rated tags’ descriptiveness in the categories market (2.2) and home (2.8) the lowest. The category transportation which included all types of transportations resulted in the highest descriptiveness rating (3.5).
Figure 8.9: Histogram of the participants’ ratings grouped by the twelve context categories. Additionally, a comparison between the number of samples of the first two rating categories (not understandable and not occurred) to the other three categories (similar sound, may occurred, and occurred) is depicted. The dashed line denotes the average rating value.
8.6. Results

![Bar graph showing descriptiveness rating and tag rating for different context categories.]

**Figure 8.10:** Comparison of the descriptiveness rating for the whole diary and the average participants tag rating. Ratings are shown for six individual context categories, and for all categories. The category *transportation* is the average over all transportation categories.

8.6.2 Results of the WordNet Analysis

Figure 8.11 shows the result of the WordNet analysis in comparison with the participants’ feedback analysis. For each context category a box plot represents the distributions of the word similarities between the tags and the context category. On each box plot the central mark is the median and the edges of the box are the 25th and 75th percentiles of the distribution. Word similarity of 1 is the highest similarity, whereas 0 the lowest. Further, the *random guess* denotes the word similarity distribution of randomly chosen tags for the 12 context categories.

The WordNet analysis showed similar results of the system’s performance compared to the participants’ feedback analysis. Largest differences between the two measurements were in the categories *tram* and *bus*. This can be explained by tag rating category *similar sound* which was frequently used in these two categories (see Figure 8.9): such rated tags showed low word similarities. Smallest differences were in the categories *restaurant* and *conversation*. Overall, the correlation between the average participants’ tag rating and the average word similarities was 0.64. Thus, we can conclude that the WordNet analysis is a valuable way to measure the recognition performance of our context recognition system.
Figure 8.11: Comparison of average participant ratings (×) and average WordNet similarity (box plot) for the generated textual diaries. The box plot indicate median and the 25th and 75th percentiles of the similarity measured by WordNet. The random guess represents the word similarity distribution for randomly chosen words in the tag vocabulary $V_d$. Correlation between average participants’ rating and similarity value was 0.64.

8.7 Discussion

Our evaluation revealed that it is feasible to use crowd-sourced tags and audio data for context diarization in daily life. The used system architecture was scalable to model more than 600 tags based on audio data of over 700 GB. Participants of the user study rated more than 60% of the tags in the textual diaries as descriptive, hence reflecting that these tags described activities and situations in their recordings. Other studies focusing on environmental sound classification with more than 20 classes showed accuracies around 60%: Eronen et al. [13] reported an accuracy of 58% for 24 environmental sound classes, whereas in our previous work ([98]) 23 sound classes were recognized with 57% accuracy. Thus, compared to literature the performance of our context diarization system showed promising results.

Furthermore, the proposed WordNet analysis (see Sec. 8.5.2) offered a way to measure the performance of the context diarization system without time-consuming evaluation using participants feedback: the WordNet analysis indicated (similarly to the participant’s feedback
8.8 Conclusion

Although our evaluational showed good system performance, we see options for improvement. In the current setting of the context diarization system, audio data is segmented with a fixed length (4 min) for automatic tagging (see Sec. 8.3.2). While tags can be used to describe longer events by multiple consecutive segments, the segmentation method could be improved by using unsupervised techniques as introduced in [109, 110, 111].

WordNet was used for the tag vocabulary creation and for evaluation purposes (see Sec. 8.5.2). However, WordNet could improve the descriptiveness of the context diarization system by modelling the similarities of the tags. This could be used for e.g. preventing the usage of synonyms to describe one situation (e.g. describing a conversation with the synonym tags “discuss” and “converse”). In a similar way, this has been used in [123].

8.8 Conclusion

We presented an approach to daily life context diarization using crowdsourced tags and audio data sourced from the web audio community database “Freesound”. A subset of 625 crowd-sourced audio tags from the Freesound vocabulary was used by the system for diarization. Each tag was modelled in the audio domain with the set of associated audio samples as obtained from the database. In total, over 740 GB of audio data was used for recognition model training.

A daily life evaluation study with 16 participants was conducted and over 360 hours of continuous daily life sound was evaluated to test descriptiveness and intelligibility of the diarization system. Two analysis method were considered: (1) participant’s feedback after presenting participants with the system generated diary of their recordings, and (2) analysing word similarity to compare the participant annotations during recordings with the textual diary. Overall, more than 60% of the tags included in the diaries were rated by the participants as descriptions of occurred events. System’s descriptiveness varied by context: tags in the “market” context had the lowest ratings (score: 2.1), whereas high ratings were achieved for restaurant and street (score: > 3). Our results showed that with the WordNet analysis, we could measure the systems descriptiveness similarly to the participant’s feedback analysis. Our results confirm that context diarization based on audio data is a suitable approach to yield intelligible and descriptive systems.
Conclusion and Outlook

This chapter concludes this work and provides an outlook for potential further research.
9.1 Conclusion

Automatically recognizing user’s context enables applications to adapt their configuration and functionality to the user’s needs and support him in his daily life. In wearable context recognition motion modalities have been traditionally used to infer user’s body movements and activities. There are, however, limitations in recognizing user’s daily life context through motion. This calls for complementary modalities to improve and extend recognition of user’s context.

In this thesis we investigated in sound-based context recognition. We envisioned a personal wearable sound-based recognition system which provides continuous real-time context information of the user throughout his day. We designed, implemented real-time sound-based recognition systems for speaker sensing (e.g. speaker identification) and ambient sound sensing (e.g. recognition of user’s activities and locations, and indoor positioning). For both categories we focused on real-life challenges, proposed techniques to increase recognition performance and reduce the process of manually collecting training data, and analysed the performance of the prototypes in daily life environments. The following conclusions can be drawn:

- Our unsupervised speaker identification system showed to be a valuable way to unobtrusively recognize a person’s conversations during his daily life. With no manual data collection, the system automatically detects unknown speakers and subsequently uses the speech data to enrol new speaker models. Our results indicated a performance of up to 81\% recognition rate for 24 speakers. Additionally, our daily-life studies showed identification performances from 60\% to 84\%, depending on the background noise during conversation.

- Using ad-hoc collaborations between speaker identification systems, identification can be enhanced. Evaluation showed that with 4 collaborating systems identification rate increases for up to 9\% for 4 speakers and up to 21\% for 24 speakers.

- User’s activities and locations can be recognized on the smartphone. We showed that real-time recognition is possible either autonomously on the smartphone or in combination with a server. For 23 ambient sound classes, a recognition accuracy of 58.45\% was reached.
9.2. Outlook

Our presented sound localization approach has large application potential for indoor location-based services as it requires very short measurement times until a robust position estimate can be derived. The evaluation showed excellent recognition accuracy of > 98% for localize a position in a set of 20 rooms with a recognition time of less than 1 s.

Using web-collected audio data to construct a context recognition system showed to be a promising approach. It provides opportunities to reduce the process of manually collecting training data as it is available in large quantities from the web. While our test showed that the web-collected sound samples were highly diverse, the proposed outlier filtering methods yielded a recognition accuracy increase of up to 18%.

Our results confirmed that context diarization based on audio data is a suitable approach to yield intelligible and descriptive systems. More than 60% of the tags included in the diaries were rated by the participants as descriptions of occurred events, hence reflecting that these tags described activities and situations in their recordings. Furthermore, our results showed that system’s descriptiveness depended on the context, e.g. descriptiveness in ”market” situation were lower than in ”restaurant” situations. Finally, using word similarity between participant’s annotations and textual diary showed similar results to the participants’ ratings.

9.2 Outlook

Based on the insights gained in this thesis the following outlook for further research is formulated:

While our speaker identification system can operate robustly with the selected training time, a faster speaker enrolment may be desirable. For this purpose modelling techniques that permits incremental learning could enable speaker training with small speech segments.

In ad-hoc collaboration for speaker identification, further work should address speaker ID mapping approaches to optimize the performance of ad-hoc mappings when system owners enter into a conversation or meeting. Additionally, the benefit of collaboration in different noise environments should be investigated.
- AmbientSense showed that real-time recognition of activities and locations is possible also in combination with a server. This opens the possibility to investigate in crowd-sourced online learning allowing users to upload their own annotated ambient sound samples to improve the auditory scene models or to extend the model set.

- Our indoor localization method uses a hearable sound to measure the room impulse response. This is unsuitable for continuous use or applications where the user is unaware of the position estimation. However, we expect that our method and results open opportunities for indoor location estimation applications using smartphones, e.g. using an ultrasonic frequency range, which is not hearable for humans.
# Glossary

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acc</td>
<td>Normalized accuracy (see Section 1.6 for a definition)</td>
</tr>
<tr>
<td>ADL</td>
<td>Activities of Daily Living</td>
</tr>
<tr>
<td>AMI</td>
<td>Augmented Multiparty Interaction corpus</td>
</tr>
<tr>
<td>AUC</td>
<td>Area Under the Curve</td>
</tr>
<tr>
<td>C</td>
<td>codebook</td>
</tr>
<tr>
<td>CC</td>
<td>Collaborative-Closed set</td>
</tr>
<tr>
<td>CO</td>
<td>Collaborative-Open set</td>
</tr>
<tr>
<td>CT</td>
<td>Center Time</td>
</tr>
<tr>
<td>d(.)</td>
<td>distance</td>
</tr>
<tr>
<td>δ</td>
<td>decision threshold</td>
</tr>
<tr>
<td>EDT</td>
<td>Early Decay Time</td>
</tr>
<tr>
<td>f_d(.)</td>
<td>decision function</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform</td>
</tr>
<tr>
<td>GLA</td>
<td>Generalized Lloyd Algorithm</td>
</tr>
<tr>
<td>GMM</td>
<td>Gaussian Mixture Model</td>
</tr>
<tr>
<td>L_n</td>
<td>local speaker set</td>
</tr>
<tr>
<td>LI</td>
<td>Local-Identical sets</td>
</tr>
<tr>
<td>LINBANDS</td>
<td>Linear Bands</td>
</tr>
<tr>
<td>LN</td>
<td>Local-Nonidentical sets</td>
</tr>
<tr>
<td>LOGBANDS</td>
<td>Logarithmic Bands</td>
</tr>
<tr>
<td>LPCC</td>
<td>Linear Prediction Cepstral Coefficients</td>
</tr>
<tr>
<td>LTI</td>
<td>Linear Time-Invariant</td>
</tr>
<tr>
<td>MFCC</td>
<td>Mel-Frequency Cepstrum Coefficients</td>
</tr>
<tr>
<td>MFCCDD</td>
<td>MFCC with first and second derivatives</td>
</tr>
<tr>
<td>MLS</td>
<td>Maximum Length Sequence</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating Characteristic</td>
</tr>
<tr>
<td>S</td>
<td>speaker</td>
</tr>
<tr>
<td>S_{Relevant}</td>
<td>relevant speakers</td>
</tr>
<tr>
<td>S_{Collab}</td>
<td>collaborative speaker set</td>
</tr>
<tr>
<td>score(.)</td>
<td>Score function for new speaker detection</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal to Noise Ratio</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>UBM</td>
<td>Universal Background Model</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>VQ</td>
<td>Vector Quantization</td>
</tr>
<tr>
<td>x</td>
<td>feature vector</td>
</tr>
</tbody>
</table>
Bibliography


[37] M. Vacher, D. Istrate, L. Besacier, J.F.Serignat, and E. Castelli. Sound detection and classification for medical telesurvey. In Pro-


[116] Creative Commons Sampling Plus License. http://creativecommons.org/licenses/sampling+/1.0/, 2012.


Curriculum Vitae

Personal information

Mirco Rossi
Born October 1, 1982, Kilchberg ZH, Switzerland
Citizen of Brusino Arsizio TI, Switzerland

Education

2008–2013 PhD studies (Dr. sc. ETH) in Information Technology and Electrical Engineering, ETH Zurich, Switzerland.

2002–2008 Dipl. El.-Ing. in Information Technology and Electrical Engineering, ETH Zurich, Switzerland.

1996–2002 Gymnasium (Matura, Typus C) at Kantonsschule Zürcher Unterland, Bülach, Switzerland.

1989–1996 Primary and secondary school Winkel bei Bülach, Switzerland

Work experience

2008–2013 Research and teaching assistant at Electronics Laboratory, ETH Zurich, Switzerland

2006 Industrial internship at KDDI, Yokosuka, Japan.

Other

2010–2013 Civil service employments working with disabled people, Zurich, Switzerland