MyConverse: Recognising and Visualising Personal Conversations using Smartphones

Mirco Rossi
Wearable Computing Lab.,
ETH Zurich
mrossi@ife.ee.ethz.ch

Oliver Amft
ACTLab, Signal Processing Systems, TU Eindhoven
amft@tue.nl

Sebastian Feese
Wearable Computing Lab.,
ETH Zurich
feese@ife.ee.ethz.ch

Christian Käslin
Wearable Computing Lab.,
ETH Zurich
kaeslinc@ethz.ch

Gerhard Tröster
Wearable Computing Lab.,
ETH Zurich
troester@ife.ee.ethz.ch

UbiComp'13 Adjunct, September 8–12, 2013, Zurich, Switzerland.

http://dx.doi.org/10.1145/2494091.2497281

Abstract
MyConverse is a personal conversation recogniser and visualiser for smartphones. MyConverse uses the smartphone's microphone to continuously recognise the user's conversations during daily life. While it recognises pre-trained speakers, unknown speakers are detected and subsequently trained for future identification. Based on the recognition, MyConverse visualises user's social interactions on the smartphone. An extensive system parameter evaluation has been done based on a freely available dataset. Additionally, MyConverse was tested in different real-life environments and in a full-day evaluation study. The speaker recognition system reached an identification accuracy of 75 % for 24 speakers in meeting room conditions. In other daily life situations MyConverse reached accuracies from 60 % to 84 %.

Author Keywords
speaker identification, real-time smartphone sensing

ACM Classification Keywords
H.5.5 [Sound and Music Computing]; H.1.2 [User/Machine Systems]

Introduction
Identifying speakers in a person's daily conversations reveals interesting information of her/his social relations, behaviour and character. Moreover, it

enables many further applications such as recognizing speech or analysing social interactions. While stationary systems such as the smart meeting room at Berkeley[1] enable to analyse meetings in one room, mobile and wearable systems can enable speaker annotation without being constrained to particular locations and thus capturing a person's interactions during daily life. The wearable system "Sociometer" [2] revealed the potential of identifying speakers for social network analysis and organizational behaviour, including analysis of social behaviour in a research group [1], modelling of group discussion dynamics [10], and prediction of shopper's interest [6]. As the speaker identification with the Sociometer is achieved through IR communication, only individuals wearing this system can be recognised.

Sound-based speaker identification enables to identify speakers without any further infrastructure. Several speaker identification systems using one microphone have been proposed (e.g. [7, 17, 4]). However, these systems can only recognise speakers with a pre-trained model. Rossi et al. [15] presented an unsupervised speaker identification system which can detect and learn previously unknown speakers and unobtrusively capture a person's conversations during daily life. While a wearable DSP implementation was presented, the system was not tested in daily life situations.

In this work, we present MyConverse, a personal conversation recogniser and visualizer for Android smartphones. MyConverse uses the smartphone's microphone to continuously recognise the user's conversation during his 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 work makes the following contributions:

1. We present the system architecture of MyConverse including an extensive study of the recognition performance using a freely available dataset.
2. We discuss the implementation of MyConverse as an Android App. Additionally, we show how conversations can be visualised 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.

Related Work

Most related to our work are the following proposed systems also focusing on speaker identification on a smartphone: EmotionSense [11], Darwin [9], and SpeakerSense [7]. EmotionSense is a sensing platform for social psychology studies based on mobile

[1] ISCI Smart Meeting Room, Berkeley: http://www.icsi.berkeley.edu/Speech/mr/

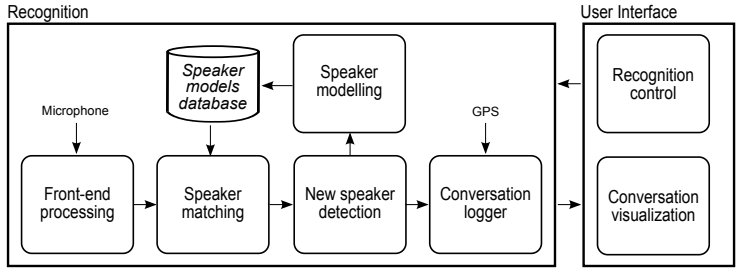


Figure 1: Architecture overview of MyConverse. The system recognises speakers and visualises the conversations in daily-life.

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. 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.

Architecture

The aim of MyConverse is to unobtrusively recognise a person's conversations 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 Fig. 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 visualise user's communication behaviours. Figure 1 depicts the components of MyConverse. The architecture was implemented as an Android app and completely runs locally on an 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 16 kHz at 16 bit 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. [12]. 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 [3]: $H(z) = 1 - \alpha z^{-1}$, 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. [7]), MFCCDD (MFCC with first and second derivatives, e.g. [17]), LPCC (linear prediction cepstral coefficients, e.g [15]), AM-FM (e.g [4]), and wavelets (e.g. [16]). For feature extraction a commonly used framing method [7, 15] 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 [15]. 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[2]. 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-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.

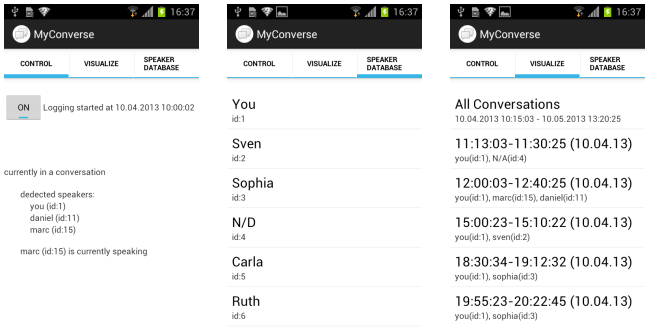
The **Speaker modelling** unit generates speaker models for unknown speakers and stores them in the *Speaker models database*. Speaker models were created based on training data consisting of a set of feature vectors $X_m = \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_M\}$ generated by the front-end processing 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. [5, 4]). 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 [8]. 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. [14]. 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[3] for GMM training and for GMM-UBM training we wrote our own code based on [8].

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

[2]CMU Sphinx Open Source Toolkit For Speech Recognition: http://cmusphinx.sourceforge.net/

[3]Mary TTS 5.0:https://github.com/downloads/marytts/marytts/marytts-5.0.zip

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: $\hat{S} = \underset{S_1 \leq k \leq S_n}{\arg\max}\, p(X|\lambda_k)$, where $p(X_r|\lambda_k)$ is the probability of the model λ_k given X_r (see [13]).
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 [14], this is verified by comparing the probability of model $\hat{S}$ and the UBM model: $\mathrm{LLR}_{\hat{S}} = \log p(X_r|\lambda_{\hat{S}}) - \log p(X_r|\lambda_{\mathrm{UBM}})$. $\mathrm{LLR}_{\hat{S}}$ is additionally normalized for better detection accuracy (see [14]): $\overline{LLR}_{\hat{S}} = \frac{\mathrm{LLR}_{\hat{S}}(X) - \mu_{\mathrm{LLR}}}{\sigma_{\mathrm{LLR}}}$, where μ_{LLR} is the mean and σ_{LLR} the variance of the set $\{\mathrm{LLR}_k(X)|k = S_1, \ldots, S_n\}$. A new speaker is detected if $\overline{LLR}_{\hat{S}}$ 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. 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 recognised 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 recognise 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. UI gives the user the possibility to start/stop the recognition, manually learn new models for a speaker, label existing models with a name, and see visualizations of the conversations. In the next section we present how MyConverse can visualise the information of the user's conversations.



(a) Control and real-time information

(b) Speaker models database

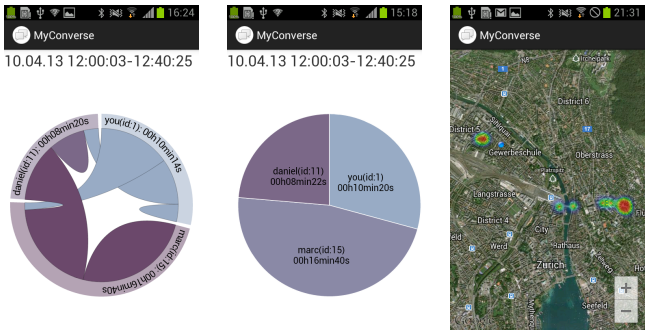
(c) Recognised conversations

Figure 2: User interface of MyConverse App enabling to control recognition, see real-time recognition information and a list of recognised conversations.

Visualization

Figure 2 shows the user interface of MyConverse. The app gives the user the possibility to start/stop the

recognition and see real time information, edit the names of enrolled speakers or manually train a model for a new speaker, and display a list of all recognised conversations including the involved speakers. Furthermore, MyConverse enables to visualise logged conversational information (see Fig. 3). Either a single conversation or all conversations together can be visualised by selecting an item in the tab "Visualize" (see Fig. 2(c)). Single conversations are visualised with two diagrams: The chord diagram depicts who has spoken to whom in a conversation assuming that a speaker A speaks with a speaker B if the speech segment of B is right after A. The pie diagram shows the total spoken time of each participant. For a visualization of all conversations together the pie diagram is also used. Additionally, all conversations are visualised by a heatmap, which shows the user's conversation on a geographical map. The colour of the heatmap represents the duration of the conversations at a specific location.



(a) Chord diagram (b) Pie diagram (c) Heatmap

Figure 3: Visualization possibilities in MyConverse. A single conversation or all conversations together can be visualised.

Evaluation

Several aspects of MyConverse were evaluated. We present the recognition performance of the system with different parameter sets, the system's recognition performance in different real-life environments, and a daily-life study targeting the evaluation of full-day usage of MyConverse focusing on conversation recognition and runtime performance of the app.

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[4] 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 $15\,\mathrm{s}$ and stored in the systems model database. The rest of the data was used to test the matching performance on a recognition length R_t of $3\,\mathrm{s}$. 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 shows the results. The highest accuracy was reached by MFCCDD feature set. GMM-UBM reaches higher recognition accuracy

[4] AMI Meeting Corpus: https://corpus.amiproject.org/

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 mixture 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.

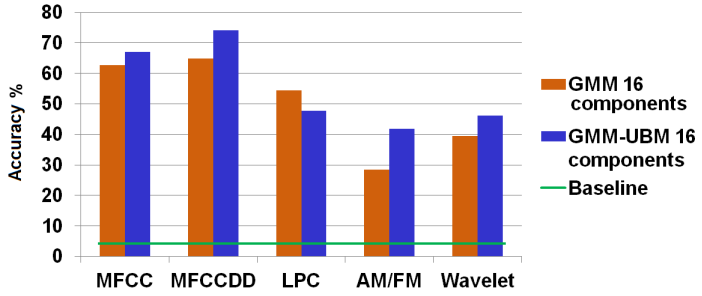


Figure 4: Recognition accuracy of the 24 speakers using different feature sets and modelling techniques (GMM and GMM-UBM). Training length was set to 15 s, recognition length to 3 s. For this evaluation new speaker detection was disabled.

The training length T_t and the recognition length R_t are crucial parameters of the recognition system. As it is desirable to recognise 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 recognise a speaker (recognition length R_t). For this evaluation we used the MFCCDD feature set and the GMM-UBM modelling approach. Figure 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.

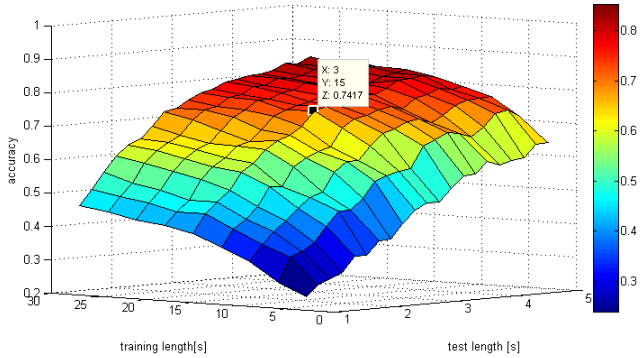


Figure 5: Recognition accuracy of the 24 speakers using MFCCDD feature set and GMM-UBM. The system performance trade-off with regard to training and recognition time is marked.

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 $\overline{\text{LLR}}_{S_{best}}$, where S_{best} is the best matching speaker model excluding the speaker model S. In the optimal case, all $\overline{\text{LLR}}_{S_{best}}$ should be smaller than the threshold T_S and be detected as a new speaker, whereas $\overline{\text{LLR}}_S$ should be above T_S and be detected as known speaker. For the selection of T_S we performed a parameter sweep maximizing the detection accuracy resulting in $T_S = 2.1$ with an accuracy of 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.

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.
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 recognised, 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 6. The accuracy of each location is an average over 5 conversations. For each location, the measured background noise level in dBA is annotated.

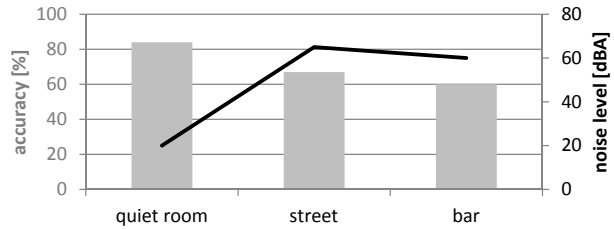


Figure 6: Comparison of speaker identification in speech segments during real-life conversations. For each location, the background noise level is annotated.

As expected, the conversation's speaker recognition accuracy for individual speech segments in the quiet room showed the best results (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.

Full-Day Evaluation Study

We investigated how well conversations were recognised in a person's daily life routines. In this evaluation study we analysed how accurate the system can recognise 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.

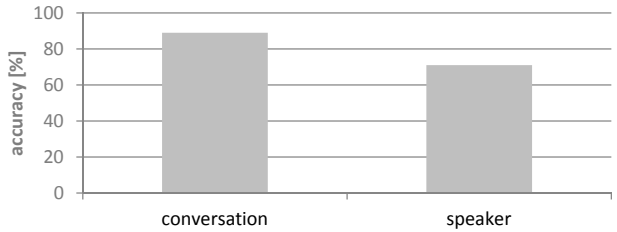


Figure 7: Recognition accuracy for conversations and speakers in the full-day evaluation study.

A conversation was counted as correctly recognised if a predicted conversation matches at least 80 % of a groundtruth 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 prediction was counted as correct. Figure 7 shows the results. In average over all three participants conversation were detected with an accuracy of 89%. Speaker recognition accuracy was 71 %.

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. 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.

Conclusion

We presented MyConverse, a personal conversation recogniser and visualiser 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 visualises user's social interactions on the smartphone. MyConverse was optimized such that new speakers are enrolled with a small amount of training data, and known speakers are recognised on small speech segments. Evaluation showed that MyConverse can recognise conversations in different real-life situations throughout the day.

References

[1] Choudhury, T. K. *Sensing and Modeling Human Networks*. PhD thesis, Massachusetts Institute of Technology, 2004.

[2] Choudhury, T. K., and Pentland, A. The sociometer: A wearable device for understanding human networks. In *Proc. of ACM Conf. on Computer Supported Cooperative Work (CSCW)* (2002).

[3] Deller, J. R., Proakis, J. G., and Hansen, J. H. *Discrete-Time Processing of Speech Signals*. Prentice Hall PTR, 2000.

[4] Deshpande, M. S., and Holambe, R. S. Speaker identification based on robust am-fm features. In *Proc. of Int. Conf. on Emerging Trends in Engineering & Technology* (2009).

[5] Deshpande, M. S., and Holambe, R. S. Speaker identification using admissible wavelet packet based decomposition. *Int. Journal of Signal Processing 6*, 1 (2010).

[6] Kim, T., Brdiczka, O., Chu, M., and Begole, J. Predicting shoppers' interest from social interactions using sociometric sensors. In *Proc. of Int. Conf. on Human Factors in Computing Systems (CHI)* (2009).

[7] Lu, H., Brush, A. B., Priyantha, B., Karlson, A. K., and Liu, J. Speakersense: Energy efficient unobtrusive speaker identification on mobile phones. Tech. rep., Microsoft Research, 2011.

[8] May, T., van de Par, S., and Kohlrausch, A. Noise-robust speaker recognition combining missing data techniques and universal background modeling. *IEEE Audio, Speech, and Language Processing 20*, 1 (2012), 108–121.

[9] Miluzzo, E., Cornelius, C., Ramaswamy, A., Choudhury, T., Liu, Z., and Campbell, A. Darwin phones: the evolution of sensing and inference on mobile phones. In *Proc. of Int. Conf. on Mobile systems, applications, and services (MobiSys)* (2010).

[10] Olguin, D., P.A.Goor, and Pentland, A. Capturing individual and group behavior withwearable sensors. In *Proc. of AAAI Spring Symposium on Human Behavior Modeling* (2009).

[11] Rachuri, K., Musolesi, M., Mascolo, C., Rentfrow, P., Longworth, C., and Aucinas, A. Emotionsense: A mobile phone based adaptive platform for experimental social psychology research. In *Proc of Int. Conf. on Ubiquitous computing (UbiComp)* (2010).

[12] Raj, B., Turicchia, L., Schmidt-Nielsen, B., and Sarpeshkar, R. An fft-based companding front end for noise-robust automatic speech recognition. *EURASIP J. Audio Speech Music Process*, 2 (2007).

[13] Reynolds, D. A. An Overview of Automatic Speaker Recognition Technology. In *ICASSP 2002* (2002), 4072–4075.

[14] Reynolds, D. A., Quatieri, T. F., and Dunn, R. B. Speaker verification using adapted gaussian mixture models. Tech. rep., M.I.T. Lincoln Laboratory, 2000.

[15] Rossi, M., Amft, O., Kusserow, M., and Tröster, G. Collaborative Real-Time Speaker Identification for Wearable Systems. In *Proc. of Int. Conf. on Pervasive Computing and Communications (PerCom)* (2010).

[16] Sarikaya, R., Pellom, B. L., and Hansen, J. H. L. Wavelet packet transform features with application to speaker identification. In *Proc. of IEEE Nordic Signal Processing Symp., Visgo* (1998).

[17] Sen, N., Patil, H., and Basu, T. A new transform for robust text-independent speaker identification. In *Proc. of IEEE India Conference (INDICON)* (2009).