TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Mensch-Maschine-Kommunikation

Intelligent Single-Channel Methods for Multi-Source Audio Analysis

Felix Johannes Weninger

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. sc. techn. Andreas Herkersdorf
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. habil. Björn W. Schuller
	2. Univ.-Prof. Dr. rer. nat. habil. Hans-Joachim Bungartz
	3. Univ.-Prof. Dr.-Ing. Eckehard Steinbach

Die Dissertation wurde am 13.11.2014 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 24.02.2015 angenommen.

Zusammenfassung

In dieser Arbeit wird das Potenzial aktueller Methoden des maschinellen Lernens zur einkanaligen Mehrquellen-Audioanalyse untersucht, d.h. Informationsgewinnung aus einkanaligen Audiosignalen, in denen Nutzsignale mit einem oder mehreren Störsignalen überlagert sind. Zunächst wird gezeigt, dass Quellentrennung durch kürzlich eingeführte Verfahren zur nichtnegativen Matrix-Faktorisierung die Erkennungsleistung erheblich erhöhen können, verglichen mit dem Standardverfahren, in dem ein Erkenner auf gemischten Signalen trainiert wird. Zweitens wird gezeigt, dass durch Formulierung der Quellentrennung als Erkennungsaufgabe aktuelle Verfahren zum überwachten Lernen z.B. tiefer neuronaler Netze genutzt werden können, um bisher unerreichte Performanz in der einkanaligen Quellentrennung zu erzielen. In diesem Kontext wird auch überwachtes Lernen für nichtnegative Modelle eingeführt. Die Aufgabe der Mehrquellen-Audioanalyse wird exemplarisch an realistischen Problemen dargestellt, wie z.B. automatischer Spracherkennung und Sprachsignalanhebung, wobei Sprachsignale mit nichtstationären Signalen wie Musik überlagert sind, und in diesem Zusammenhang werden führende Ergebnisse in internationalen Forschungswettbewerben werden vorgestellt. Durch Anwendung der vorgestellten Methoden werden weiterhin in ausgewählten Anwendungen der polyphonen Musikverarbeitung, z.B. Erkennung von Sängereigenschaften oder Musiktranskription, Ergebnisse nach dem neuesten Stand der Technik erzielt.

Abstract

This thesis investigates the potential of recent machine learning methods for the challenging task of single-channel, multi-source audio audio analysis, i.e., information extraction from single-channel audio where the sources of interest (e.g., speech) are mixed with multiple interfering sources. First, it is shown that source separation by recently proposed techniques for non-negative matrix factorization can significantly improve the recognition performance, compared to the state-of-the-art approach of training the recognition task with multi-source data. Second, it is shown that by formulating the source separation problem itself as a recognition task, state-of-the-art methods for supervised training of recognition systems such as deep neural network models can be used to achieve previously unseen performance in single-channel source separation. In this context, supervised training of non-negative models is introduced as well. The task of multi-source recognition as defined above is exemplified by challenging real-world speech separation and recognition problems where speech is mixed with non-stationary background noise such as music, and world-leading results in international evaluation campaigns are demonstrated for this task. Furthermore, state-of-the-art results are presented in selected music information retrieval applications involving polyphonic audio, such as characterizing the singer, or transcribing the music into a score.

Preface

This thesis is based on selected pre-publications made during my time as a PhD student in the Machine Intelligence & Signal Processing Group, Institute for Human-Machine Communication (MMK), Technische Universität München (TUM) in Munich, Germany. The selection is based on scientific relevance and recency, and aims at a coverage of various applications in both speech and music analysis. The first aim of this thesis is to make these pre-publications more accessible to the generally knowledgeable reader, who should be familiar with the basics of digital signal processing and machine learning. The second aim is to provide a broader view of the concept of multi-source analysis than is present in the pre-publications. To this end, the pre-publications have been restructured into methodology and results, cast into a general research framework, and augmented by an introductory chapter and concluding remarks. Hopefully, this results in a publication that is a joy to read from the beginning to the end. The interested reader can find references to the pre-publications and related literature throughout this thesis.

Munich,
Fall 2014

Felix Weninger

Acknowledgment

First of all, I would like to thank my supervisor Björn Schuller for his inspirational supervision. He was never short of ideas for exciting new research projects, comments to improve my scientific thinking and writing, as well as personal and professional advice. Furthermore, I would like to thank my colleagues at the Institute for Human-Machine Communication for the helpful discussions on the results presented in this thesis. In particular, I would like to thank Florian Eyben, Jürgen Geiger and Martin Wöllmer for contributing their expertise in audio signal processing, machine learning, and automatic speech recognition, and Johannes Bergmann, Jordi Feliu and Christian Kirst for their highly valuable contributions to the experiments on multi-core implementation, music separation and music transcription. Besides, I would like to thank John Hershey, Jonathan Le Roux and Shinji Watanabe at Mitsubishi Electric Research Laboratories, Cambridge, MA, for the great opportunity to do an internship in the area of robust speech analysis, and many fruitful follow-up discussions on source separation and feature enhancement. Particular thanks also go to the co-authors of the pre-publications being collected in this thesis – in addition to the people named above and in alphabetical order, these are Noam Amir, Ofer Amir, Hans-Joachim Bungartz, Jort Gemmeke, Antti Hurmalainen, Gerhard Rigoll, Irit Ronen, Yuuki Tachioka, and Tuomas Virtanen. My research at TUM was funded by the German Research Foundation (grant nos. SCHU 2508-2, 'Non-Negative Matrix Factorization for Noise-Robust Feature Extraction in Speech Processing' and SCHU 2508-4, 'Context-Sensitive Automatic Recognition of Spontaneous Speech with BLSTM Networks'), and by the HUAWEI Innovation Research Program (HIRP) in the Generic Live Audio Source Separation (GLASS) project. Last but not least, I thank my family for their moral and financial support.

Contents

Part I

Introduction

1

Introduction

The world's continual breathing is what we hear and call silence.
– Clarice Lispector

1.1 Motivation

Sound is everywhere – many environments we encounter in our daily lives contain multiple sound sources. As a consequence, audio signals of interest for technical systems, including speech, music, or acoustic events, rarely occur in isolated form. Thus, for intelligent systems of the future it will be crucial to understand audio signals that are blended with others – for example, recognizing the words in a speech signal mixed with background music and street noise while driving a car, a signal recorded on a mobile phone while walking down a busy street, or working in an office environment with interferences from background talkers and appliances.

In this thesis, *multi-source audio analysis* is formulated as the generic task of extracting information from an audio source of interest in the presence of potentially multiple interfering sources. Due to its practical importance, in the last years there has been considerable progress in this direction [41, 114], and attention has shifted away from considering audio analysis tasks in isolation inside the lab, in favor of realistic setups. This holds for tasks such as automatic speech and emotion recognition [108, 241], audio event classification [128], etc.

Applications of multi-source audio analysis are found in human-computer interaction – a very well known and highly important practical challenge is distant-talking Automatic Speech Recognition (ASR), where one or more microphones capture not only the speech source of interest, but also a considerable amount of interferences from the environment [108, 203]. Furthermore, multi-source audio analysis tasks are naturally given by Music Information Retrieval (MIR) applications, since music generally comprises multiple audio sources – e.g., singers, instrumentalists, and crowd noise in live recordings. Note that the task of transcribing the lyrics of a song, or

recognizing the identity of the singer, bears some similarity to speech and speaker recognition in non-stationary noise – with the 'non-stationary noise' comprising instrumental accompaniment [221].

Existing approaches to solve multi-source recognition can be roughly divided into two categories: noisy (multi-condition) training, where a task of interest (e.g., ASR) is learnt in a supervised way by associating audio mixtures with the labels (e.g., phonemes) of the source(s) of interest; and source separation, where the goal is to first estimate the source(s) of interest (in the time, time-frequency, or feature domains) before further processing. Of course, it seems natural to combine these approaches to achieve optimal performance, but this is more intricate than it might seem at first: Separated signals will generally mismatch both the 'clean' sources and the mixture signals on which typical recognizers are trained.

An important insight is that if we formulate the optimal recovery of the source(s) as the task of interest, source separation can be cast as an instance of noisy training. Hence, the general framework of this thesis is to solve prediction tasks in the presence of interferences. For this, machine intelligence will be leveraged, in order to tackle the particular challenge of performing multi-source recognition on single-channel audio. By that, source separation and noise-robust recognition will be unified in a single research framework, which allows for investigating optimal machine learning strategies for tasks including speech enhancement, ASR and MIR. The next section will present the above arguments in a more formal manner.

1.2 Problem statement

For a finite number of sources $S \geq 2$, let

$$y^* : (s_1, \ldots, s_S) \mapsto (y_1^*, \ldots, y_S^*) \tag{1.1}$$

be a labeling function, i.e., a mapping from source signals (finite sequences) $s_1, \ldots, s_S$ to target signals (finite sequences) $y_1^*, \ldots, y_S^*$. For example, when recognizing speech from concurrent speakers, the signals $s_1(\tau), \ldots, s_S(\tau)$ correspond to speech signals from S different speakers, and $y_1^*, \ldots, y_S^*$ correspond to textual transcriptions. Similarly, the automatic piano transcription problem (cf. Section 7.3) can be formalized in this framework by assuming that $s_1(\tau), \ldots, s_S(\tau)$ are sequences of acoustic realizations of piano notes belonging to a certain pitch ($S = 88$), and $y_1^*(\tau), \ldots, y_S^*(\tau)$ are the corresponding note onset indicator functions.

Applying the traditional view of pattern recognition, one would strive to find a set of single-source recognition functions $\hat{y}_l$ that approximate y_l^* for all $l = 1, \ldots, S$:

$$\hat{y}(s_1, \ldots, s_S) = (\hat{y}_1(s_1), \ldots, \hat{y}_S(s_S)) \approx y^*(s_1, \ldots, s_S). \tag{1.2}$$

However, in most practical scenarios, one does not have access to the separated source signals s. Instead, a mixing process $\mathcal{M}(s_1, \ldots, s_S) = (m_1, \ldots, m_C)$ yields C

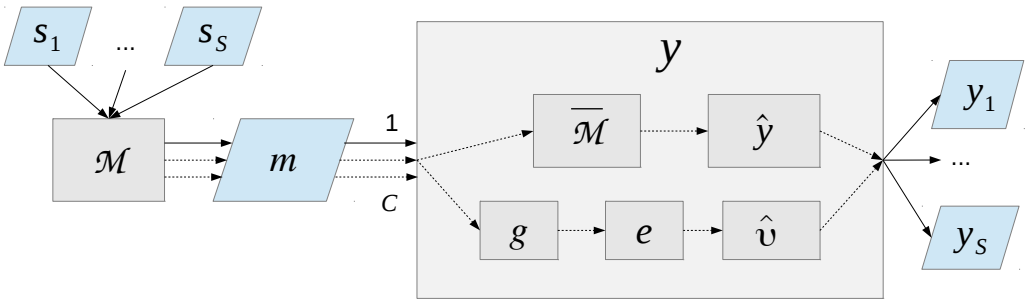


Figure 1.1: Schematic overview of multi-source audio recognition: The recognition function y is implemented as a mapping from mixture signal(s) $m_1, \ldots, m_C$ (obtained as the output of an unknown mixing process $\mathcal{M}$ on the sources $s_1, \ldots, s_S$) to source labels $y_1, \ldots, y_S$. The implementation may include source separation, $\overline{\mathcal{M}}$, or feature-domain separation / enhancement, e. $\hat{y}$ and $\hat{v}$ denote single-source recognition functions ($\hat{v}$: defined in the feature domain), and g denotes feature extraction.

observed mixture signals $m_1, \ldots, m_C$ on C receivers (microphones, channels). Based on this and (1.2), in this thesis, multi-source recognition is defined as finding a mapping y fulfilling

$$y(\mathcal{M}(s_1, \ldots, s_S)) = y(m_1, \ldots, m_C) \approx y^*(s_1, \ldots, s_S). \tag{1.3}$$

It is tempting to assume that the above problem is just an instance of the well-known *source separation* problem, i.e., that it can be reduced to inverting the mixing process $\mathcal{M}$. However, the connection between the two is more sophisticated. It is clear that if $\mathcal{M}$ is a bijective function, i.e., the mixing process is deterministic and reversible, there exists an 'unmixing' function $\mathcal{M}^{-1}$ such that for a suitable $\hat{y} \approx y^*$, $y = \hat{y} \circ \mathcal{M}^{-1} \approx y^*$. An example for a reversible mixing process is noiseless determined instantaneous mixing,

$$\mathcal{M}(s_1, \ldots, s_S) = (m_1, \ldots, m_S) = \mathbf{A}(s_1, \ldots, s_S), \tag{1.4}$$

with a full rank $S \times S$ matrix $\mathbf{A}$ and the mixture and source signals corresponding to matrix rows. In other words, assuming (i) one can find $\mathcal{M}^{-1}$ effectively, and (ii) recognition on separated sources fulfills $\hat{y} \approx y^*$, the ideal solution is to perform source separation, then apply the mapping $\hat{y}$ to the separated signals[1].

Yet these two assumptions are hardly met in any realistic application. First, the assumption that $\mathcal{M}$ is bijective is unrealistic, foremost due to underdetermination, $C < S$, i.e., having less channels than sources. In this thesis, the 'heavily underdetermined' case $C = 1$ (single-channel) is of particular interest, as systems capable of handling single-channel signals can be applied to any type of multi-source audio,

[1]This is not as trivial as it might seem since $\mathcal{M}$ is unknown in practical applications; however, Independent Component Analysis (ICA) has been shown to solve the case of noiseless determined instantaneous mixing to perfection more than a decade ago [92].

regardless of the number of available channels and the microphone placement. In this context, it must also be noted that in many application scenarios the mixing process is convolutive due to reverberant environments – i.e., each source signal is convolved with a Room Impulse Response (RIR) h_l:

$$\mathcal{M}(s_1, \ldots, s_S) = (m_1, \ldots, m_C) = \mathbf{A}(h_1 * s_1, \ldots, h_S * s_S), \tag{1.5}$$

with a matrix of mixing coefficients $\mathbf{A} \in \mathbb{R}_+^{C \times S}$. This effectively results in additional sources corresponding to reflected sounds (convolutive noise), $h_l * s_l \rightsquigarrow s_l + s_{l'}$. Finally, measurement noise, such as from recording equipment, adds random fluctuations to the mixing process $\mathcal{M}$, which are hard to impossible to revert.

Clearly, in case that $\mathcal{M}$ cannot be inverted, the question how to design y becomes very interesting. y will be called *multi-source recognition function* subsequently. For example, one could design y as a composition $y = \hat{y} \circ \overline{\mathcal{M}}$, where $\overline{\mathcal{M}}$ approximates the inverse of $\mathcal{M}$, i.e., $\overline{\mathcal{M}} \circ \mathcal{M} \approx \mathrm{id}$ – again reducing multi-source recognition to source separation followed by single-source recognition. However, unlike in the case of reversible mixing, there is no guarantee that this is optimal.

The second assumption, $y^* \approx y$, might not be met either, as y^* (the true labeling function) can be very hard to compute (think of the notoriously difficult ASR problem), inefficient to compute accurately, or even incomputable, so that $\hat{y} \not\approx y^*$. Given such $\hat{y}$ (say, a well-trained ASR system), the question arises: Should one design y as $\hat{y} \circ \overline{\mathcal{M}}$, i.e., imperfect source separation followed by imperfect recognition, or should one try to find a 'direct' mapping y such that $y \circ \mathcal{M}$ is close to y^*? As source separation seems to be a very hard problem, one might be tempted to consider the latter as being more straightforward – in other words, one could solve the multi-source recognition problem without performing explicit source separation. In fact, this approach (known as 'noisy training' or 'multi-condition training') can be very effective in practice, particularly if y is implemented by powerful machine learning models. For instance, in the case of recognizing speech mixed with noise sources, it is now known that Deep Neural Networks (DNNs) can exploit noisy speech data very well [183], and training with noisy speech even helps generalization compared to training with isolated speech [247]. Furthermore, if one has a generative model of $(m_1, \ldots, m_C) \mid (y_1^*, \ldots, y_S^*)$, i.e., of the mixed observations given certain labels of the sources, one can attempt to solve the multi-source recognition problem (1.3) by inference [160, 206][2].

There are a couple of arguments to the defense of source separation, though. First, it is now well known that some processes in the human auditory processing resemble source separation [129]. Thus, if the goal is to design intelligent methods in the sense of *artificial intelligence* resembling human cognitive processes, one should not neglect source separation. Furthermore, in some applications, such as speech separation for human-human communication (e.g., in telephony or hearing aids), the

[2]For larger numbers of possible labels and multiple sources this can quickly become intractable.

task of interest is to recover source signal(s) – to phrase this in the above framework, one can simply define the labeling y^* to be the identity function, and it follows that one needs to find y such that $y \circ \mathcal{M} \approx \mathrm{id}$, which is exactly the source separation problem, i.e., $y \approx \mathcal{M}^{-1}$. Finally, speech has many facets that can be recognized – the spoken words, the attributes (age, gender, personality, likability, etc.) of the speaker, speaking style (casual, whispered, emotional), etc. [174]. If one wants to recognize all these facets robustly, it might be ineffective to perform multi-condition training over and over again for all these tasks, as opposed to having only recognizers trained on separated sources ('clean training') and a source separation module that can be connected[3].

Another relevant aspect for the design of the multi-source recognition function is that in practice, recognition is hardly ever performed directly on time domain signals, but features are extracted in an intermediate step (cf. Section 3.1). Let us now explicitly model the feature extractor as a function g; then, the multi-source audio analysis problem in the feature domain becomes

$$v(g(\mathcal{M}(s_1, \ldots, s_S))) = v(g(m_1), \ldots, g(m_C)) \approx y^*(s_1, \ldots, s_S) \tag{1.6}$$

where v is used instead of y to distinguish the problem from the time-domain equivalent (1.3). Then, instead of attempting to approximate (1.6) by finding a source separation function $\overline{\mathcal{M}}$, we can find a *feature enhancement* function e such that $e \circ g \circ \mathcal{M} \approx g$, giving rise to the multi-source recognition function $y = \hat{v} \circ e \circ g$ with $\hat{v}$, $\hat{v} \circ g \approx y^*$, being a single-source recognition function in the feature domain. This will be applied to a practical ASR task in Section 5.2.

Finally, it is noteworthy that in some applications, such as speech enhancement (cf. Chapter 4), or speech recognition in noise (cf. Chapter 5), only one source $l \in \{1, \ldots, S\}$ is of interest. The above framework is flexible enough to support this case – one can simply suppose $y^*(s_{l'}) = \epsilon$ (empty string) for all $l' \neq l$ and adjust the recognition objective accordingly. A practical example for this is given in Section 3.6.

Figure 1.1 summarizes the multi-source recognition process via multi-condition training, source separation, and feature enhancement.

1.3 Objectives

Based on the above considerations, the research agenda for this thesis consists of two main objectives:

1. To verify the hypothesis that including an explicit source separation or feature enhancement step in the multi-source recognition process yields higher performance in ASR and MIR tasks than noisy training.

[3]For this reason, a 'plug-and-play' scenario, where a clean trained recognizer is used on separated signals, will be considered in this thesis whenever possible.

2. To demonstrate that it is particularly effective to formulate source separation as a recognition task and use supervised training of models which are typically used for recognition, instead of unsupervised or weakly supervised methods.

Arguably, *intelligence* is both necessary and sufficient to solve the single-channel multi-source audio analysis problem: necessary, since rule-based methods generally do not perform well on challenging multi-source tasks, such as speech enhancement in non-stationary noise, where they are outperformed by machine learning [111, 229]; and sufficient, considering human performance on this task, e.g., recognizing the melody and lyrics of a song played on the radio.

To quantify if the above objectives are met, standard evaluation measures and datasets will be used. These will be the subject of Chapter 2. In particular, this thesis will demonstrate the successful application of the proposed techniques to the 2011 and 2013 Computational Hearing in Multisource Environments (CHiME) Challenges, as well as the 2014 Reverberant Voice Enhancement and Recognition Benchmark (REVERB). Next, Chapter 3 will describe some of the chosen machine learning methods in detail, including their efficient implementation on modern parallel computing architectures. Then, Chapter 4 will describe the potential of machine learning based methods to find separation functions in the light of the second objective, by the example of speech enhancement. Chapter 5 will investigate the first hypothesis in the context of automatic speech recognition in additive noise, and Chapter 6 will do the same for convolutive noise (reverberation). Finally, in Chapter 7 the above hypotheses will be tested in the domain of MIR. Conclusions are drawn in Chapter 8.

Part II

Methodology

2

Measures of success

Goals are dreams with deadlines. – Diana Scharf Hunt

It can be argued that in the field of engineering, the true measure of success is the acceptance of the resulting system by users. In the case of source separation, this can correspond to the perceived separation quality, such as the intelligibility of a separated speech signal. In the case of ASR, it corresponds to whether the errors of the system lead to critical misunderstandings between the human and the machine. However, such *subjective* measurements are costly to obtain, and notoriously difficult to predict automatically [105]. As a consequence, the research community has picked up on *objective measures* that are deterministic functions of the recognizer output and the desired output, such as the rate of words recognized correctly in a speech signal, or the similarity of the separated and the original sources. In this thesis, established objective measures for multi-source recognition are used, starting with energy-based measures for source separation quality and then moving to evaluation in terms of recognition accuracy. These will be presented in the first part of this chapter.

This chapter then concludes with an overview of data sets that are used to compute these objective measures. The crucial question is how objective measures obtained on a given data set relate to the expected performance in some application scenario – again, in general terms, this question is unsolved. However, there are still some commonly used criteria for classifying data sets according to their suitability for evaluation. These are mostly based on heuristics and practical experience – a simple example is the size of the data set, resulting from the belief that 'there is no data like more data'. More of these criteria will be discussed below.

2.1 Evaluation measures

In the light of the objectives of this thesis, mainly two kinds of objective criteria come to mind: source separation quality and recognition accuracy.

2.1.1 Source separation evaluation

This section deals with *energy-based* objective measures for source separation evaluation. Let us start by reviewing simple measures of correlation between true source signals and estimated sources.

2.1.1.1 Correlation-based source separation measures

Smaragdis [185] introduced the Source Ratio (SR) for a source s_1, interference s_2 and estimated source $\hat{s}_1$ as follows[1]:

$$\mathrm{SR} = 20 \log_{10} \frac{\varrho(\hat{s}_1, s_1)}{\varrho(\hat{s}_1, s_2)}, \tag{2.1}$$

where ϱ denotes the Pearson correlation coefficient. In case that the source and interference signals have zero expectation, this equals

$$\begin{aligned} \mathrm{SR} &= 20 \log_{10} \frac{\sum_\tau \hat{s}_1(\tau) s_1(\tau) \sqrt{\sum_\tau \hat{s}_1(\tau)^2} \sqrt{\sum_t s_2(\tau)^2}}{\sum_\tau \hat{s}_1(\tau) s_2(\tau) \sqrt{\sum_\tau \hat{s}_1(\tau)^2} \sqrt{\sum_t s_1(\tau)^2}} && (2.2) \\ &= 20 \log_{10} \frac{\langle \hat{s}_1, s_1 \rangle |s_2|}{\langle \hat{s}_1, s_2 \rangle |s_1|} && (2.3) \end{aligned}$$

with $\langle \cdot, \cdot \rangle$ denoting the scalar product and $|\cdot|$ the L2-norm.

An advantage of this measure is that it is very efficient to compute (in linear time). A drawback is that if the source and the interference are similar, i.e., $s_1 \approx s_2$, the above is likely to yield $\mathrm{SR} \approx 0\,\mathrm{dB}$ regardless of the goodness of the approximation $\hat{s}_1$. However, in many applications sources and interferences can be assumed to be uncorrelated, e.g., in speech / noise separation, or in speaker separation, which is why the above measure is certainly justified.

2.1.1.2 Projection-based source separation measures

A more generic approach for source separation evaluation, which is nowadays widely used, also as competition measure in the Signal Separation Evaluation Campaign (SiSEC) [141], was introduced by Vincent et al. [200]. Reasons for the popularity of this approach include that the measures are still rather straightforward to compute, easy to extend to an arbitrary number of sources, and that they do not make any assumption about the mixing or unmixing process, or about the independence / orthogonality of the sources. Furthermore, distortions by source separation are taken into account explicitly. This is achieved by decomposing an estimated source

[1] Since Smaragdis [185] addresses the specific case of separating two concurrent speakers, the acronym SR originally stands for *Speaker* Ratio.

$\hat{s}_l(\tau)$ into a component $s_l^{\text{target}}(\tau)$ that represents the actual source, plus error terms $e_l^{\text{interf}}(\tau)$ and $e_l^{\text{artif}}(\tau)$ representing the presence of interferences and artifacts:

$$\hat{s}_l(\tau) = s_l^{\text{target}}(\tau) + e_l^{\text{interf}}(\tau) + e_l^{\text{artif}}(\tau). \tag{2.4}$$

This decomposition is effectively computed by

$$\begin{aligned}
s_l^{\text{target}} &= \langle \hat{s}_l, s_l \rangle s_l / |s_l|^2, && (2.5)\\
(\mathbf{R_{ss}})_{l,l'} &= \langle s_l, s_{l'} \rangle, && (2.6)\\
\mathbf{C} &= \mathbf{R_{ss}}^{-1} \left(\langle \hat{s}_l, s_1 \rangle, \ldots, \langle \hat{s}_l, s_S \rangle \right)^\intercal, && (2.7)\\
e_l^{\text{interf}} &= \mathbf{C}^\intercal \mathbf{s} - s_l^{\text{target}}, && (2.8)\\
e_l^{\text{artif}} &= \hat{s}_l - \mathbf{C}^\intercal \mathbf{s}, && (2.9)
\end{aligned}$$

where $s_l(\tau)$ is the separated source l and $\mathbf{s}$ is a matrix with rows corresponding to source signals. $\mathbf{R_{ss}}$ is the *Gram matrix* whose inverse is used to define an orthogonal projector in (2.7). Then, one can define the Source-to-Distortion Ratio (SDR), Source-to-Interference Ratio (SIR), and Source-to-Artifacts Ratio (SAR) [200] by

$$\begin{aligned}
\text{SDR}_l &= 10 \log_{10} \frac{|s_l^{\text{target}}|^2}{|e_l^{\text{interf}} + e_l^{\text{artif}}|^2}, && (2.10)\\
\text{SIR}_l &= 10 \log_{10} \frac{|s_l^{\text{target}}|^2}{|e_l^{\text{interf}}|^2}, && (2.11)\\
\text{SAR}_l &= 10 \log_{10} \frac{|s_l^{\text{target}} + e^{\text{interf}}|^2}{|e_l^{\text{interf}}|^2}. && (2.12)
\end{aligned}$$

It is easy to see that for $S = 2$ orthogonal sources, i.e., $\langle s_1, s_2 \rangle = 0$, and letting $l = 1$ and $l = 2$ represent the desired source and interference,

$$\begin{aligned}
s_1^{\text{target}} &= \langle \hat{s}_1, s_1 \rangle s_1 / |s_1|^2, && (2.13)\\
e_1^{\text{interf}} &= \langle \hat{s}_1, s_2 \rangle s_2 / |s_2|^2, && (2.14)\\
\text{SIR}_1 &= 10 \log_{10} \frac{\langle \hat{s}_1, s_1 \rangle^2 |s_2|^2}{\langle \hat{s}_1, s_2 \rangle^2 |s_1|^2}, && (2.15)
\end{aligned}$$

the latter being equivalent to SR (2.3). Thus, SIR can be seen as a generalization of SR to multiple non-orthogonal sources.

Although for non-orthogonal sources, the computation of e_l^{interf}, and hence SDR, SIR and SAR, requires solving a linear system of equations, the dimensionality of this system is only determined by the number of sources S. Thus, for a constant number of sources, SDR, SIR and SAR can be computed in linear time.

2.1.2 Recognition evaluation

In the following, the most common objective measures to evaluate the performance of a recognition system with symbolic output, such as class labels or text strings, will be presented. Apart from these, there are other measures which are relevant to specific applications, but for clarity these will be introduced 'on-the-fly' later on in Part III.

2.1.2.1 Classification accuracy

Probably the most straightforward objective measure of a system's performance in a classification task is to estimate the probability of correct classification. Given a sequence of discrete labels $y_1^*, \ldots, y_N^*$ and a sequence of predicted labels $y_1, \ldots, y_N$ for the N test instances, accuracy (Acc) is defined as

$$\mathrm{Acc} = \frac{\sum_{i=1}^{N} \delta(y_i, y_i^*)}{N}, \tag{2.16}$$

where δ is the Kronecker delta. Alternatively, if the distribution of class labels is non-uniform, it can be more meaningful to consider (average) recall (Rec):

$$\mathrm{Rec} = \frac{1}{L} \sum_{c=1}^{L} \frac{\sum_{i:y_i^*=c} \delta(y_i, y_i^*)}{N_c} \tag{2.17}$$

where L is the number of classes and $N_c = |\{i : y_i^* = c\}|$ is the number of test instances with class label c. The difference to Acc is that always picking the class with the highest prior N_c/N can result in high accuracy, whereas it leads to chance level recall.

2.1.2.2 Transcription accuracy

In computing transcription accuracy, a reference sequence of symbols $y_1^*, \ldots, y_N^*$ is compared to a recognized sequence of symbols $y_1, \ldots, y_{N'}$. For example, in automatic speech recognition, the symbols correspond to words. Since the recognition algorithm can erroneously insert or omit symbols, there is generally no one-to-one correspondence between reference and and recognized symbols; hence, the measures defined above cannot be used directly. Instead, an alignment of the reference and recognized sequences has to be determined first, before calculating accuracy. This can be done by a dynamic programming algorithm which is similar to the one proposed by Wagner and Fischer [207] for the calculation of the Levenshtein (edit) distance, but counts insertions, deletions, and substitutions separately. After execution of this algorithm, the Word Accuracy (WA) is given by

$$\mathrm{WA} = \frac{N - I - D - S}{N} \times 100\,\%, \tag{2.18}$$

where N is the number of symbols, and I, D, and S are the numbers of insertions, deletions, and substitutions. Note that in contrast to Acc and Rec, WA can be negative, for example in case of many insertions. As the 'opposite' of WA, Word Error Rate (WER) is defined as

$$\mathrm{WER} = 100\,\% - \mathrm{WA} = \frac{I + D + S}{N} \times 100\,\%. \tag{2.19}$$

2.1.2.3 Relation to source separation quality

When evaluating multi-source recognition systems, it is tempting to conjecture that there is a generic relation between recognition accuracy on separated sources and source separation quality. However, this hypothesis has still to be verified on a larger scale. This thesis presents some evidence in Section 5.1.1 and Section 5.1.2. Furthermore, it is promising that automatic speech recognition accuracy has been demonstrated to be a predictor of speech intelligibility in the cases of speech coding and pathological speech [73, 195].

2.1.3 Significance testing

An issue of high practical interest is to compare the performance of multiple systems on a given recognition task. It has to be determined whether observed differences in performance are caused by random fluctuations. This is usually achieved by statistical significance testing. The following section briefly introduces important types of significance tests for classification and general recognition tasks.

2.1.3.1 Comparing two accuracies

The z-test as described by Dietterich [30] can be used to test if the accuracies of two systems A and B are significantly different. Let p_A and p_B denote the probabilities of correct classification, which are measured as the accuracies on a given data set. Without loss of generality, we assume that $p_B > p_A$, i.e., B seems to perform better than A. One then supposes (formulates the *null hypothesis* H0) that the observed performances are caused by random deviations from the probability of correct classification by either system, p_{AB}. Since p_A and p_B are measured on the same data set, it holds that $p_{AB} = (p_A + p_B)/2$. Then, the null hypothesis is disproven at a chosen level of significance. Denoting the number of correct classifications by N_c, N_c follows a binomial distribution with success probability p_{AB}:

$$N_c \sim \mathrm{Bin}(N, p_{AB}). \tag{2.20}$$

Accordingly, one observes the improved accuracy of B with probability

$$P(N_c > p_B \cdot N) = 1 - P(N_c \leq p_B \cdot N). \tag{2.21}$$

Approximating the binomial distribution by a normal distribution with mean $N \cdot p_{AB}$ and variance $N \cdot p_{AB}(1 - p_{AB})$, the standardized (z-normalized) test quantity $z^*_{c,B}$ becomes

$$z^*_{c,B} = \frac{p_B - p_{AB}}{\sqrt{p_{AB}(1 - p_{AB})}}\sqrt{N}. \tag{2.22}$$

The criterion for rejecting H0 is

$$p = 1 - \Phi(z^*_{c,B}) < \alpha, \tag{2.23}$$

for the significance level α and the cumulative distribution function of the standard normal distribution Φ. One typically speaks of significant differences if $\alpha < 0.05$ (this is the significance level used in this thesis unless stated otherwise). The left hand side of the inequality (2.23) is called *p-value* and corresponds to the probability of wrongly rejecting H0, i.e., assuming a significant difference when none is given. As noted by Dietterich [30], the above test tends to overestimate significances, i.e., underestimate p-values. However, its calculation is very simple and is still worthwhile if significance testing is not understood in the strict sense of calculating an error probability but rather as an objective measure for identifying observations deserving further investigation [137].

2.1.3.2 Comparing multiple performance measurements

Due to the assumption of a binomial distribution, the above significance test can only be applied when test instances are classified independently from each other. This is not the case, e.g., in ASR. In this case, comparing WAs according to the above, with N corresponding to the number of words, leads to false positives [60]. Gillick and Cox [60] suggest to obtain the WAs per sentence, since these can be assumed to be statistically independent measurements – the state of a typical speech recognizer is reset after processing an utterance, cf. Section 3.7. An alternative is to use the WA per speaker, as proposed by the National Institute of Standards and Technology (NIST) [143].

In all these cases, the assumption of a Bernoulli experiment for each measurement (transcription of a sentence or the utterances of a speaker) does not make sense. A similar argument holds for tasks such as source separation, where the measurement consists of computing real-valued quantities such as the Signal-to-Noise Ratio (SNR) or SDR for each test signal. Thus, in the following a more generic significance test for comparing two sets of performance measurements is presented. Denoting the performance measurements of system A by a random variable A and those of B by B, the performance difference is also a random variable, denoted by D.

As null hypothesis H0, we suppose that the difference D is normally distributed with zero mean and standard deviation σ_D. Under H0, the standard normal test

quantity becomes

$$z_D^* = \frac{\overline{D}}{\sigma_D}\sqrt{N}, \tag{2.24}$$

where $\overline{D}$ is the mean difference on the test set. If the goal is to test whether B is better than A (or A is better than B), one speaks of a one-sided test (as in the z-test described above), and H0 can be rejected if $1 - \Phi(z_D^*) < \alpha$.

The crux is that $\overline{D}$ and σ_D are not known and have to be estimated from the observations of A and B. For small N, the estimation error in $\overline{D}$ and σ_D can not be neglected, and these quantities have to be modeled as random variables themselves. This is addressed by Student's t-test, which is preferred over a z-test in the case of small N. A disadvantage of Student's t-test is that the random variables A and B need to follow a normal distribution, which might not always be given.

A non-parametric alternative, i.e., one that does not make particular assumptions about the distributions of the measurements, is Wilcoxon's signed rank test [86, 234]. As in other non-parametric statistics such as Spearman's rank correlation coefficient, the core idea is to define a normally distributed random variable based on a ranking of the observations. In this case, the ranking R_i of the absolute measurement differences $|a_i - b_i|$, $i = 1, \ldots, N$ is used. Wilcoxon's test statistic is defined as

$$W = \left| \sum_{i=1}^{N} (\mathrm{sgn}(a_i - b_i) \cdot R_i) \right|. \tag{2.25}$$

The null hypothesis H0 is that the median difference of A and B is zero. It can be shown that under H0 and for sufficiently large N, W follows a normal distribution with mean $\mu_W = 0.5$ and standard deviation

$$\sigma_W = \sqrt{\frac{N(N+1)(2N+1)}{6}}. \tag{2.26}$$

Then, the standard normal test quantity $z_W^* = (W - 0.5)/\sigma_W$ is tested for significance, and H0 can be rejected if $1 - \Phi(z_W^*) < \alpha$. Comparing speaker WERs with Wilcoxon's signed rank test is one of the tests used by the NIST for their official evaluations [143].

2.2 Data sets for multi-source audio analysis

Having discussed the evaluation measures, the data sets used in this thesis for evaluation of multi-source recognition systems will be introduced. For comparability of results, a clear focus is on standard data sets that are publicly available, and that have been used for research challenges. A few criteria will be defined that might help in assessing the strengths and weaknesses of data sets in providing a performance

measure for a multi-source recognition algorithm. This adds another facet to the important topic of result validation, along with statistical significance testing as introduced in the previous section.

An important criterion is whether data sets are constrained to a single domain, a small (closed) set of sources (e.g., speakers, singers, instruments, noise types), etc. This is because if a system outperforms another on such a constrained data set, it is hard to estimate if this would also be the case in a more general scenario. Furthermore, data sets pose a different level of challenge at the signal level – for example, present noise sources can be stationary or non-stationary. If multi-source data is generated artificially, this can be done by convolutive or additive mixing of separated sources – the former arguably posing a greater challenge to recognition systems.

Ideally, data sets should also feature real recordings of multi-source data instead of artificially generated recordings, because it is not clear if results obtained on artificially generated data can be replicated in real applications. Conversely, simulated test data can be useful to some extent to gain insight into the behavior of algorithms in very specific conditions (e.g., SNR levels, reverberation times), and simulated training data might be useful to help generalization, due to the extreme ease of simulating the most varying conditions.

2.2.1 Speech and noise

Let us begin this discussion with the important special case of speech recognition in noisy environments. For this scenario, a plethora of data sets are available – a comprehensive overview is given by Le Roux and Vincent [110]. These data sets can be characterized by a few criteria – the following lists are to be understood as non-exhaustive. First, the ASR task itself can be characterized by, e.g.,

- vocabulary (small: few hundred words or less / medium: few thousand words / large: $> 10\,\mathrm{k}$ words),
- speaking style (prompted or conversational / casual speech), and
- speaker independence, i.e., disjoint sets of speakers, in the training / test split commonly used for evaluation.

Then, the interferences can be characterized by their types, e.g., by

- additive stationary vs. non-stationary noise,
- convolutive noise: short-term (channel distortions) and long-term (room reverberation),
- real vs. artificially added noise and reverberation, and

- strict training / test split of noise (only relevant in case of artificially added noise).

It can be argued that an ASR system that comes close(r) to the goal of interacting with humans in daily life needs to cope with speech that is actually recorded in a variety of acoustic environments (and thus might or might not be similar to artificially corrupted speech), and spontaneous speaking style, in a speaker-independent fashion with more or less open vocabulary. It is notable that despite the relative ease of collecting large amounts of realistic speech data, so far there is no publicly available data set that reflects these conditions. However, in the opinion of the author, a few publicly available data sets are interesting enough to be considered for evaluation, because they cover at least a few of the desirable aspects.

2.2.1.1 Computational Hearing in Multisource Environments (CHiME)

The database referred to as *CHiME-2011* was the subject of the 2011 CHiME Challenge [7, 21]. By that, it enables comparison of the results obtained in this thesis with other studies. It considers the scenario of separating speech from 'highly non-stationary' noise. In particular, noises from a home environment including household appliances and small children are considered; furthermore, speech is heavily reverberated. In contrast, the ASR task itself is very easy, featuring command utterances with a fixed grammar (from the Grid corpus [24] that had already been used for the preceding CHiME-2006 benchmark). Nevertheless, in the noise and reverberation the speech recognition task is challenging even for humans – a trained human was found to be at around 94 % accuracy [7].

The corpus contains corrupted versions of 24 200 utterances of 34 speakers from the Grid corpus [24], subdivided into a training (17 000 utterances), development (3 600), and test set (3 600). Noise and reverberation were added artificially. The dry utterances in each set were convolved with a different binaural RIR, corresponding to varying room configurations (e.g., doors open/closed, curtains drawn/undrawn). Noise was added to the development and test sets by selecting segments from a recorded noise corpus matching specific SNRs from -6 to 9 dB, in steps of 3 dB. Only limited scaling of speech and noise is used. A corpus of isolated training noise signals is provided, and the background noise in the development and test set is disjoint from this training noise. Yet, no official multi-condition training set is provided.

The database referred to as *CHiME-2013* was introduced for the second track[2] of the second installment of the CHiME Challenge in 2013 [202]. The noise corpus is the same as in the CHiME-2011 data, but the ASR task is considerably more complex, featuring medium vocabulary (5 k) speech from the WSJ-0 corpus [147]. While large

[2]The data used for the *first* track of the second CHiME Challenge was virtually identical to the CHiME-2011 setup – despite small head movements which have next to no influence on monaural systems – and is thus of minor relevance for this thesis. It will be referred to as *CHiME-2013-SV*.

vocabulary speech (20 k) is also available, no comparative evaluations were done on these data in the Challenge, which is why results in this thesis are reported only on the 5 k task. The noise-free training set as well as the multi-condition training set consist of 7 138 utterances each (in analogy to the WSJ-0 SI-84 set). For the multi-condition training set, six SNRs from -6 to 9 dB are used, in steps of 3 dB. The development and test sets (similar to the WSJ-0 si_dt_05 and si_et_05 sets) consist of 410 and 330 utterances at each of these SNRs, for a total of 2 460 / 1 980 utterances. By construction of the WSJ-0 corpus, the CHiME-2013 evaluation is speaker-independent. As in CHiME-2011, there is a training / test split of noise and RIRs.

A drawback of the CHiME family of databases, besides the absence of real recordings of noisy speech, is that the simulated acoustic conditions largely overlap between training and test data. The noise recordings and RIRs, while split into training and test sets, are all taken from the same home environment, and hence vary only slightly. The resulting absence of mismatched conditions in training and test might introduce a bias towards training-based approaches to ASR (such as deep neural network acoustic models) when evaluating on these data sets.

2.2.1.2 Data sets introduced by the author

2.2.1.2.1 CHiME-2011-Buckeye A variation of the above database was created by the author and his colleagues [223]. It will be referred to as *CHiME-2011-Buckeye*. The noise is exactly the same as in CHiME-2011, but the ASR task is among the most challenging ones so far considered in the domain of noise-robust ASR, featuring conversational speech from the Buckeye corpus [148]. At the same time, the mixing setup from the CHiME-2011 Challenge (SNR levels and selection of speech / noise pairs matching a specific SNR, instead of scaling) was preserved as closely as possible. The high variability of conversational speech presents an additional challenge to training-based approaches for source separation and robust ASR, besides the variability of the noise.

2.2.1.2.2 Noisy TUM AVIC With a similar goal, yet taking into account also multiple (simulated) mouth-to-microphone distances and recording rooms, as well as aiming at realistically sounding recordings instead of random speech and noise combinations, the Noisy TUM AVIC corpus (NAVIC) was introduced by the author of this thesis and his colleagues [43]. It is an artificially corrupted version of Technische Universität München (TUM)'s Audio-Visual Interest Corpus (AVIC), which consists of spontaneous dialogues where 21 subjects show various levels of arousal depending on their interest in the conversation. In the following, the data set and partitioning as used by Weninger et al. [229] is presented. As test partition, the one defined for the INTERSPEECH 2010 Paralinguistic Challenge [176] is used, which is balanced and stratified by gender. A random 30 % split of the Challenge

training and development set is used for early stopping of the training algorithm (cf. below). By this partitioning, strict speaker independence is given.

Realistic noise samples of three types as used by Eyben et al. [44] serve as additive noise: Babble noise, city street noise, and music. Babble noise recordings are samples from the `freesound.org` website in the categories pub-noise, restaurant chatter, and crowd noise. Music recordings are instrumental and classical music from the `last.fm` website. The city recordings were recorded in Munich, Germany [178]. The length of the noise pool is 30 minutes per noise type in the test set and roughly 6.5 hours in total in the training set.

Furthermore, RIRs from the Aachen Impulse Response Database [95] were used to add convolutive noise. A few meaningful combinations of noise types and RIRs were selected to realize the above mentioned goal of realistically sounding recordings: babble noise and lecture room, babble noise and stairway, city noise and meeting room, and music noise and chapel (Aula Carolina), thus representing a wide range of stationary and non-stationary additive noises and favorable to heavily reverberated room acoustics. For each condition, three different virtual microphone distances are employed. Degraded speech utterances were created by first padding with silence – this allows algorithms to perform blind background noise estimation –, then convolving with a RIR, normalizing to -6 dB peak amplitude, and mixing with a randomly selected additive noise sample (respecting the train/test split), which is convolved with the RIR ('far' distance) and scaled in order to achieve a given SNR. The test set of each corpus is convolved with the 'near', 'mid', and 'far' impulse responses and noise is added at SNRs from 0 to 20 dB in steps of 5 dB, resulting in 15 test sets for each acoustic condition, thus 60 test sets with 73 k utterances in total. The training set has twelve times the size of the original AVIC training set (32 k utterances in total), because each utterance is included once for the 3 RIR distances and four acoustic conditions. In the training set, noise at random SNRs (uniformly distributed on the range 0–25 dB and with 10% probability of SNR = ∞) is added. SNRs are calculated after first order high pass filtering of speech and noise, approximating A-weighting to better match human perception.

2.2.1.3 REVERB Challenge

The corpus of the 2014 REVERB Challenge[3] [104] serves to evaluate mainly the removal of convolutive noise (room reverberation effects). Here, eight microphone channels are available. The corpus contains both a simulated data set based on the WSJCAM0 corpus [162], which is convolved using six different real room impulse responses (three rooms, near and far microphone distances) and corrupted by various types of pre-recorded stationary noise, and a 'real world' data set from the MC-WSJ-AV corpus [116], recorded in a reverberant meeting room with real ambient

[3] `http://reverb2014.dereverberation.com/`

noise, at near and far distances. These sets will be referred to as SimData and RealData in the ongoing. In accordance with the 2014 REVERB Challenge guidelines, the experiments in this thesis use either the first (reference) channel or all eight channels. Overall, the SimData set has 1 484 utterances from 20 speakers (10 female) and the RealData set has 179 utterances from five speakers. For Multi-Condition Training (MCT), the Challenge multi-condition training set is used, which is generated in analogy to the SimData set. It is of the same size as the clean WSJCAM0 training set, containing 7 861 utterances from 92 speakers (39 female); room impulse responses and noise types are chosen randomly, with equal probability. Note that both MC-WSJ-AV and WSJCAM0 are based on the prompts from the Wall Street Journal (WSJ) corpus, which allows for using the standard WSJ language models for both corpora. According to the 2014 REVERB evaluation protocol, the RealData set is used only for evaluation, not for training. Hence, the REVERB corpus allows assessing the generalization capability of a system trained on simulated data.

2.2.1.4 Other databases

- *Aurora-2*: This was one of the first attempts at a standard set to evaluate noise-robust ASR [84]. It was motivated by the application of continuous digit recognition on mobile devices. Thus, the ASR task is characterized by little phonetic confusions between the keywords, but the possibility of insertions and deletions (as opposed to fixed grammar tasks, cf. below). While channel effects (G.712 and MIRS filtering according to International Telecommunication union (ITU) standards) were addressed, there is no long-term reverberation. While the database has a mismatched training/test setup (test noise and channel effect not seen in training), the amount of noise recordings is very little compared to the amount of speech, and there is no strict training/test split of noise. The WER of a simple HMM recognizer using the ETSI front-end and noisy training is already below 10 % [57]. Overall, the database seems a bit outdated today and is thus not considered further in this thesis.

- *Aurora-4*: This database is similar to Aurora-2, but featuring the WSJ-0 5 k ASR task [147] instead of spoken digits. Compared to Aurora-2, the amount of interference is lower, as the lowest SNR is 5 dB – this might be more realistic, because additive mixing neglects the Lombard effect which is not appropriate at low SNR. As in Aurora-2, since the same noise corpus is used, there is little variety in noise. The database is not considered in this thesis because there is a similar, yet more challenging setup presented by the CHiME-2013 corpus.

- *CHiME-2006*: This data set is intended to study the task of speaker separation, i.e., reversing the cocktail party effect, in a single channel [23]. In principle, this appears to be a very challenging task due to non-stationarity. However,

the setup of the task was rather artificial: (i) only mixings of exactly two speakers were considered; (ii) it was assumed that for every test recording, it is known which two speakers are present; (iii) the effect of reverberation was not considered; (iv) the individual tracks follow a fixed grammar, where each keyword can only occur at a defined point in the utterance. Due to the peculiarities of this setup, very specific recognizer architectures [160, 206, 210] are the most successful on this task. Hence, the practical significance of these results might be rather limited, and the database is thus not considered further in this thesis.

- *COSINE*: This database provides recordings of *conversational speech in noisy environments* [190]. It certainly has a few appealing facts to it: Recordings are taken from real multi-party conversations on a campus, thus featuring multiple sources (speakers, environmental noise) in real instead of artificial mixing; there are time-aligned recordings with different microphone arrays as well as a close-talk microphone, offering potential for real stereo training; and the speech is spontaneous and conversational. In the opinion of the author, the main drawback is that most of the interferences are masked by wind noise; consequently, the performance is heavily influenced by a system's capability of removing wind noise, instead of by its capability to separate the other sources. It might be one of the reasons that the database has not seen widespread usage in the noise-robust ASR community – to the knowledge of the author, few studies so far report any results on this database (e.g., [237]).

2.2.1.5 Summary

A summary of the databases mentioned in this section, including the total recording lengths, is given in Table 2.1. Note that the Aurora-4, CHiME-2013, and REVERB databases are all based on the WSJ-0 corpus, and the different lengths result from the variants of artificial corruption used.

2.2.2 Polyphonic music

Let us now proceed to databases for MIR in polyphonic music, applying similar criteria as above. It is clear that the criterion of speaker-independence can be easily 'translated' to singer- or instrument-independence for recognition tasks in vocal and instrumental music, and training/test splits or training/test mismatches can be defined in the same way. A different criterion is musical genre, (e.g., classical, opera, jazz, or chart music), which strongly influences the audio sources which are present, and also the singing or interpretation style (for example, figurations or vibrato in singing yield much more complex melodic patterns, arguably increasing the difficulty of source separation).

Table 2.1: Databases featuring speech and noise sources; Abbreviations: ST: Speech type (conv: conversational; FG: fixed grammar; SV/MV/LV: small/medium/large vocabulary); NS: non-stationary (additive) noise; CM: Convolutive mixing / presence of long-term reverberation; SI: speaker independence; TT: train/test split of interference; MM: mismatched train/test setup; RR: (partly) real recordings. [1]: LV available, but not used here.

Name	Length	ST	NS	CM	SI	TT	MM	RR
Aurora-2	42:46 h	read/SV	√	–	√	–	√	–
Aurora-4	84:54 h	read/MV	√	–	√	–	√	–
CHiME-2006	8:48 h	read/FG	√	–	–	–	–	–
COSINE	12:47 h	conv/MV	√	–	√	√	–	√
CHiME-2011	12:15 h	read/FG	√	√	–	√	–	–
CHiME-2011-Buckeye	50:30 h	conv/MV	√	–	√	√	–	–
CHiME-2013	40:23 h	read/MV/LV[1]	√	√	√	√	–	–
REVERB	26:28 h	read/MV	–	√	√	√	√	√
TUM-NAVIC	76:18 h	conv/MV	√	√	√	√	–	–

Table 2.2: Databases for polyphonic music information retrieval. [1]: Not 'instrument-independent'. See Table 2.1 for an explanation of the abbreviations.

Name	Length	Style	CM	SI	TT	MM	RR
Singer Trait Recognition							
TUM-Ultrastar [175]	37:03 h	Pop	√	√	√	–	√
Singing Style Recognition							
TAU-Vibrato [219]	0:13 h	Opera, Jazz, Pop	√	√	√	–	√
Piano Transcription							
MIDI [149]	20:03 h	Classical	–	–[1]	–	–	–
MAPS-MIDI [38]	13:45 h	Classical	–	–[1]	–	–	–
MAPS-Disklavier [38]	4:22 h	Classical	√	–[1]	–	–	√

Compared to ASR, the distinction between artificial and realistic data cannot be easily tied to the mixing process. A truly 'in-the-wild' recording would correspond to a live recording of an artist with accompaniment, probably in the presence of interfering sources (audience, environmental noise etc.) Yet, many of the audio recordings found in people's music archives are produced by professional mixing in studios, and in this case they are usually augmented by artificial reverberation, rather than recordings in real acoustic environments. Thus, a big difference between robust ASR and polyphonic music databases is that studio recordings of music are clearly meaningful from an application point of view, whereas it is hard to find an application where artificial mixings of speech and other sources are of interest. What will be referred to as 'artificial data' in the context of MIR is polyphonic music generated by synthesis, e.g., from MIDI files.

Table 2.3: Number of instances per class (no vibrato / vibrato), genre (jazz / pop / opera) and fold number in the TAU Vibrato Database.

Fold #	Genre	No vib.	Vibrato	Sum
1	Jazz	12	41	53
	Pop	24	28	52
	Opera	–	65	65
	All	36	134	170
2	Jazz	8	34	42
	Pop	8	33	41
	Opera	–	47	47
	All	16	114	130
3	Jazz	14	21	35
	Pop	18	29	47
	Opera	–	48	48
	All	32	98	130
All	Jazz	34	96	130
	Pop	50	90	140
	Opera	–	160	160
	All	84	346	430

A summary of the databases discussed in this section, according to the aforementioned criteria, is given in Table 2.2. The MIR tasks associated with these databases will be explained in detail below.

2.2.2.1 TAU Vibrato database

The Tel Aviv University (TAU) Vibrato database consists of 430 segments, each corresponding to one sung note in professionally recorded music. 30 different artists are found in the database, all of which are accomplished female singers. Genres cover jazz, pop and opera with 10 singers each, and 130, 140 and 160 instances, respectively. The segments were extracted manually and labeled by experts as containing vibrato or not. All opera segments are sung with vibrato while the percentage of vibrato segments is approximately 2/3 for pop and 3/4 for jazz. The average note duration is 1.86 s with a standard deviation of 1.10 s and considerably differs among genres, with jazz exhibiting at the same time the longest average duration (2.12 s) and the greatest standard deviation (1.54 s). In some instances, vibrato is delayed in the note, which is particularly often the case in jazz: a delay occurs for 52 of 96 instances (54 %) of jazz notes, reaching up to 81 % of the total note duration.

For vibrato singing, singer-independence seems particularly desirable, as vibrato styles differ considerably among singers [139]. Thus, a singer-dependent system would be prone to over-fitting to the specificities of the singers in the database – which could be called a 'singer effect' in analogy to the well-known 'speaker effect'. To

ensure singer-independence for the experiments presented in this thesis, the database is subdivided into three folds for singer-independent cross-validation stratified by genre, i. e., the class and genre distributions in each fold are approximately equal. For the sake of reproducibility, these folds were obtained as follows: For each genre, the ten singers were sorted alphabetically and assigned to fold 1 (singers 1–4), fold 2 (singers 5–7) or fold 3 (singers 8–10). Consequently, of the 430 instances in the database, 170 / 130 / 130 are assigned to each of the three folds. The resulting numbers of instances per fold are shown in Table 2.3.

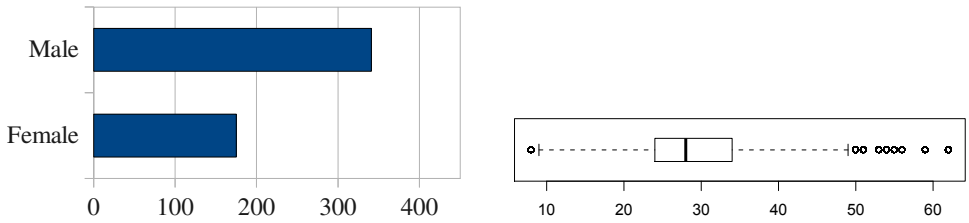
2.2.2.2 UltraStar singer traits database

For experiments on singer trait recognition (classification of singers by age and gender), the UltraStar database was introduced by the author of this thesis and his colleagues, first with gender annotation [175, 218], and later with additional attributes: age, height, and (biological) race [213]. The database contains 581 songs commonly used for the 'UltraStar' karaoke game, corresponding to over 37 h total play time. The ground truth tempo is provided and lyrics are aligned to (quarter) beats.

To ensure a singer-independent partitioning, the database is split according to the first letter of the name of the performer into training (A, D, G, ...), development (B, E, H, ...) and test partitions (0-9, C, F, ...). The identity of the singer(s) was determined at beat level wherever possible. This is particularly challenging in case of formations such as 'boy-' or 'girl-groups', in which case the 'singer diarization' (alignment of the singer identity to the music) was determined from publicly available music videos. Then, information on gender, height, birth year and race of the 516 distinct singers present in the database was collected and multiply verified from on-line textual and audiovisual knowledge sources, including IMDB, Wikipedia and YouTube. All annotation was performed by two male experts for popular music (24 and 28 years old). Of the annotated traits, only age and gender will be considered for automatic recognition in this thesis, due to the strong correlation of height with gender and the data scarcity for non-white singers in the chosen set of popular songs.

In a multitude of cases, two or more singers are singing simultaneously. This was handled by the following scheme: For gender, the beats were marked as 'unknown' unless all simultaneously present artists share the same attribute value. For age, the average value was calculated, since in formations the individual artists' traits are usually similar. This procedure was also followed to treat performances of formations where an exact singer diarization could not be retrieved, by assuming presence of an 'average singer' throughout. In case that the desired attribute is missing for at least one of the performing vocalists, the corresponding beats were marked as 'unknown' – these are excluded from analysis later on.

The distribution of gender among the 516 singers is shown in Figure 2.1a. Age (Figure 2.1b) is shown as box-and-whisker plot where boxes range from the first to



(a) gender (b) age [years]

Figure 2.1: Evaluation database for singer trait recognition: Distribution of gender, race, and height among 516 singers in the UltraStar Singer Trait Database. Distribution of age is shown on beat level, since it is dependent on recording date.

Table 2.4: Number of beats per trait, class and set (train / devel /test) in the UltraStar singer trait database. 'Unknown' (?) includes simultaneous performance of artists of different gender, as well as those with unknown age.

# beats	**train**	**devel**	**test**	Σ
no voice (0)	90 076	75 741	48 948	214 765
		gender		
female (f)	32 308	23 071	9 739	65 118
male (m)	55 505	49 497	37 686	142 688
?	86	253	771	1 110
		age		
young (y)	48 510	42 056	25 682	116 248
old (o)	34 074	24 596	18 712	77 382
?	5 315	6 169	3 802	15 286
Σ	**177 975**	**148 562**	**97 144**	**423 681**

the third quartile and all values that exceed that range by more than 1.5 times the width of the box are considered outliers, depicted by circles. Unlike gender, the age distribution can only be given on beat level since age is not well defined per artist (due to different recording dates) nor per song (due to potentially multiple singers per song). The continuous-valued age was discretized to 'young' (y, < 30 years) and 'old'; the threshold is close to the median (28 years) to avoid scarcity of either class. From the manual singer diarization and collection of singer meta data, the beat level annotation is generated automatically, resulting in the number of beats and according classification tasks shown in Table 2.4. The annotation (singer meta-data, voice alignments, song list with recording dates and partitioning) is made publicly available for research purposes at `http://www.openaudio.eu`.

Table 2.5: Evaluation databases for automatic music transcription: MIDI, MAPS MIDI and MAPS Disklavier. Total recording lengths of the partitions given in hours:minutes:seconds.

Dataset	Partition	# pieces	# notes	Length
MIDI	training	200	519 477	14:18:18
	validation	26	59 835	1:59:33
	test 1	35	71 242	2:20:03
	test 2	23	58 223	1:25:05
MAPS MIDI	training	155	334 974	9:41:18
	validation	21	48 921	1:45:23
	test 1	23	36 075	1:23:47
	test 2	11	41 018	0:54:04
MAPS Disklavier	training	36	86 010	2:23:56
	validation	6	16 487	0:43:21
	test 1	3	5 675	0:11:08
	test 2	15	46 180	1:03:17

2.2.2.3 Automatic music transcription

To evaluate the performance of automatic transcription of piano music, the MIDI database introduced in [149] and the MAPS (MIDI Aligned Piano Sounds) database [38] are used. The MIDI database consists of MIDI files collected from the Classical Piano MIDI Page[4]. The MIDI files are converted to waveforms with a sampling rate of 44.1 kHz using the freely available Maestro Concert Grand v2 sound font[5]. The MAPS database consists of synthesized music as well as real piano recordings. The first part (MAPS MIDI) is created with different software synthesizers, configurations and virtual recording conditions. The second part (MAPS D) dataset contains music played by a Yamaha Disklavier Mark III in realistic recording conditions ('ambient' and 'close'). For a detailed description of the database the reader is referred to [38]. The Disklavier part of the MAPS database is treated as an individual corpus, since these are the only recordings of a real piano in the data sets considered.

Statistics of the individual data sets are shown in Table 2.5. The partitioning into training, validation and test sets as used by Poliner and Ellis [149] and Böck and Schedl [15] is followed. However, note that Böck and Schedl restricted their evaluation on the test set to a subset for which the alignments were manually verified (testing 1); in this thesis, additional evaluations are done on the full test set (testing $1 \cup 2$), which may contain alignment errors.

[4] `http://www.piano-midi.de`

[5] `http://www.linuxsampler.org/instruments.html`

3

Learning multi-source recognition

There are some things you learn best in calm, and some in storm.
– Willa Cather

In the following, the implementation of multi-source audio recognition functions will be discussed, fleshing out the details of the coarse framework laid out in Section 1.2. In particular, methods for supervised learning of recognition functions on mixed and separated sources, as well as source separation and feature-domain enhancement, will be presented. For the sake of readability, and in the light of this thesis' objectives, single-channel mixtures will be assumed throughout ($C = 1$). It is straightforward to extend the feature extraction and learning schemes to multiple channels, for example, by concatenating single-channel feature vectors [91, 117].

3.1 Feature extraction

The traditional way of pattern recognition looks at a recognition function y as a composition of a 'hard-coded' pre-processing and feature extraction process with few, if any, free parameters – which are usually 'tuned' manually – and a model h for the prediction of the labels with trainable parameters, which are typically gained from minimizing the prediction error of y. Denoting the pre-processing and feature extraction steps by $g_1, g_2, \ldots, g_K$, one can write this as

$$y = h \circ g_K \circ \cdots \circ g_1(x). \tag{3.1}$$

For example, let us assume that it is desired to build a predictor for the gender of a speaker. Then, g_1 could represent the Short-Term Fourier Transformation (STFT) of the time-domain signal, g_2 the detection of the fundamental frequency from the spectrum, g_3 the process of pitch tracking, and g_4 the calculation of the mean over time, etc. For this case of 'hand-crafted' feature extraction, h could be very simple

in turn: for example, a decision stump. However, if the type of model chosen for h has enough power to subsume $g_k, \ldots, g_K$, one can reformulate (3.1) as

$$y = v \circ g_{k-1} \circ \cdots \circ g_1(x) \tag{3.2}$$

with $v = h \circ g_K \circ \cdots g_k$ being the new recognition model.

In fact, many features considered relevant in speech and audio processing, such as loudness, fundamental frequency, voice quality, chroma, etc., can be derived from the STFT. Thus, it can be conjectured that powerful recognition models can learn to exploit these relationships automatically for prediction of the task of interest, and thus the manual feature extraction process can be constrained to very simple operations. Moreover, if z is formulated as a function with trainable parameters, the feature extraction steps g_{k-1} to g_K can be optimized according to the recognition objective (e.g., the classification accuracy). As a matter of fact, it is believed that the power of DNN models (cf. Section 3.3) is partly due to this kind of 'blurring' between feature extraction and recognition [131]. Furthermore, as will be shown in Section 3.5.2, more advanced models such as Recurrent Neural Networks (RNNs) even possess the power to perform temporal integration of information – theoretically[1], they can subsume g_1 to g_4 as well as h in the above example. For these reasons, in this thesis, the feature extraction process will, for the most part, be constrained to simple spectral and cepstral features[2].

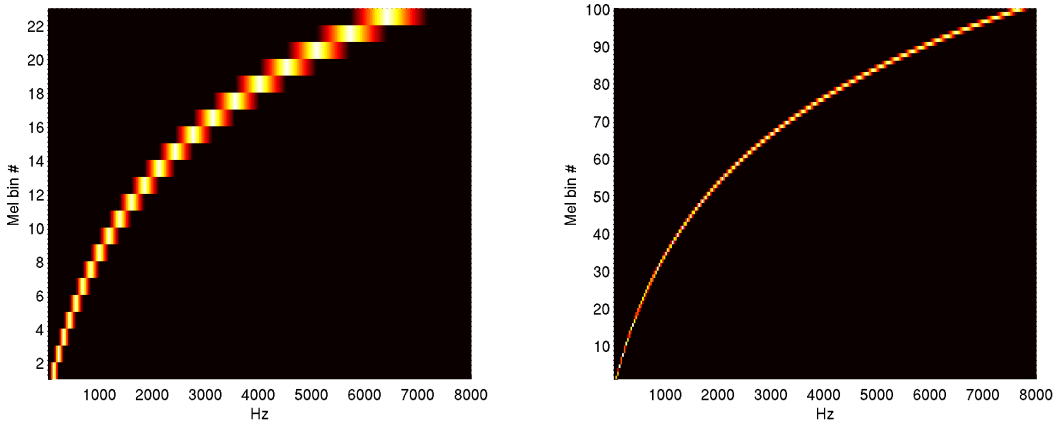
The spectral feature extraction process is based on the complex STFT $\mathbf{z} = \mathcal{F}(x, w, \Delta\tau)$, where $x(\tau)$ is a discrete-time single-channel signal, $w(\tau)$ is a discrete-time window function for a window size W, and $\Delta\tau$ is the frame shift, $\Delta\tau \in]0, W/F_s]$, where F_s is the sampling frequency. In most audio recognition applications, the phase of the complex spectrum $\mathbf{z} \in \mathbb{C}^W$ is neglected and only the magnitude spectrum $\mathbf{x} = |\mathbf{z}|$ is kept, yielding $\mathbf{x} \in \mathbb{R}_+^F$ with $F = \lfloor W/2 + 1 \rfloor$ frequencies due to symmetry.

Further, non-linear warping is often applied to the magnitudes and frequencies. The magnitudes are warped by a mapping $\mathbf{x} \mapsto \mathbf{x}^\alpha$ – e.g., $\alpha = 2$ to obtain the power spectrum, and $\alpha = 2/3$ to obtain the 'auditory' spectrum as in calculation of Perceptual Linear Prediction (PLP) coefficients [78]. $\alpha < 1$ is used to compress the dynamic range, which can alternatively be achieved by logarithmic warping, $\mathbf{x} \mapsto \log \mathbf{x}$, where log is applied element-wise.

To implement the warping of the frequency axis, a linear transformation with matrix $\mathbf{B}$ can be used, $\mathbf{x} \mapsto \mathbf{B}\mathbf{x}$. The well-known 'Mel filter bank' with B filters is a suitable choice for that purpose, which can be represented as a matrix $\mathbf{B} = (b_{i,f}) \in [0, 1]^{B \times F}$, such that $\sum_{f=1}^{F} b_{i,f} x_f^\alpha$ is the output of filter i applied to the F-dimensional

[1] In practice, this ability is restricted by sub-optimal training schemes and the quantity of available training data – for instance, Sections 4.2 and 5.2 will show that indeed the parameterization of the feature extraction can be important even with RNNs.

[2] This fact is also mirrored in the notation: the index f will be used for both feature and frequency indices, and the symbol F for numbers of features/frequencies.



(a) $B = 23$ (b) $B = 100$

Figure 3.1: Heatmap visualization of the matrix $\mathbf{B}$ representing the Mel filterbank used for non-linear warping of the frequency axis, for $B = 23$ and $B = 100$ Mel filters.

spectrum. The coefficients are chosen such that centers of the filters are equidistantly spaced on the Mel frequency scale. Furthermore, they fulfill $\sum_f b_{i,f} = 1$, and each filter has a triangular shape, with the rising flank of filter i being equivalent to 1 minus the falling flank of filter $i-1$. Figure 3.1 shows the matrix $\mathbf{B}$ for $W = 512$, i.e., $F = 257$, and $B = 100$ (which seems a good choice for source separation applications, cf. Section 4.2) and $B = 23$ (which is the default in the open-source ASR toolkit Kaldi [152]).

In addition, the filterbank outputs are often processed by logarithmic warping. Combining this with warping of the filterbank *inputs* yields filterbank features $\log \mathbf{B}\mathbf{x}^\alpha$, where $\mathbf{x}$ is a magnitude spectrum. This kind of features for $\alpha = 1$ will be referred to as 'Log-FB' subsequently.

The *Mel cepstrum* is (conceptually) obtained as $\tilde{\mathbf{x}} = \mathbf{D} \log \mathbf{B}\mathbf{x}^\alpha$, where $\mathbf{D}$ is the matrix representing the Discrete Cosine Transformation (DCT) coefficients. Typically $\alpha = 1$ or $\alpha = 2$ are chosen in the computation of the Mel cepstrum. The components of the vector $\tilde{\mathbf{x}}$ are known as Mel-Frequency Cepstral Coefficients (MFCCs)[3]. The DCT serves as an approximation of a de-correlation operation [107], which is useful for some recognition models such as diagonal covariance Gaussian Mixture Models (GMMs).

Subsequently, $\alpha = 1$ (use of magnitude spectra) will be assumed unless stated otherwise.

[3]The MFCC features as used in typical ASR systems, and in this thesis, include further operations such as high-pass filtering, and scaling ('liftering') of the MFCCs – for details, refer to [245].

3.2 Non-Negative Matrix Factorization

As a first example of a multi-source recognition algorithm, which operates on simple spectral features (usually magnitude or power spectra), Non-Negative Matrix Factorization (NMF) will be introduced, which is popular and effective for single-channel audio source separation, cf., e.g., [5, 157, 185, 220]. In the most general form, the goal of NMF is to represent a data matrix $\mathbf{V}$ with T non-negative observations in columns, $\mathbf{V} = [\mathbf{v}_1 \cdots \mathbf{v} \cdots \mathbf{v}_T]$, $\mathbf{v}_t = (v_{1,t}, \ldots, v_{F,t})^\intercal \in \mathbb{R}_+^F$, where F is the dimension of the feature space, by

$$v_{f,t} \approx \sum_{r=1}^{R} w_{f,r} h_{r,t} \tag{3.3}$$

subject to $w_{f,r}, h_{r,t} \geq 0$ for a given number of *components* R. The matrix $\mathbf{W} = (w_{f,r})$ is called *dictionary*, with dictionary atoms in columns.

Note that imposing other constraints apart from non-negativity on the representation (3.3) leads to other popular information reduction or extraction methods. For example, Principal Component Analysis [99] constrains the dictionary vectors to the eigenvectors of the covariance matrix $\mathbf{C_{VV}}$ (and hence to be orthogonal), while no constraints are put on the coefficients $h_{r,t}$. If only $w_{f,r}$ are required to be non-negative, this leads to *semi-NMF*, which has some applications in image processing [196]. However, as will be explained in more detail below, non-negativity constraints for $w_{f,r}$ and $h_{r,t}$ are motivated from the audio source separation problem.

The factors $\mathbf{W}$ and $\mathbf{H} = (h_{r,t})$ are obtained by minimizing a distance (cost) function $D(\mathbf{V}|\mathbf{WH})$, such as the Euclidean distance

$$D_{\mathrm{ED}}(\mathbf{V}, \mathbf{WH}) = \sum_{f,t} (v_{f,t} - (\mathbf{WH})_{f,t})^2 = \sum_{f,t} \left(v_{f,t} - \sum_r w_{f,r} h_{r,t} \right)^2 \tag{3.4}$$

or the Kullback-Leibler-(KL-)-divergence-like function

$$D_{\mathrm{KL}}(\mathbf{V}|\mathbf{WH}) = \sum_{f,t} v_{f,t} \log \frac{v_{f,t}}{(\mathbf{WH})_{f,t}} - v_{f,t} + (\mathbf{WH})_{f,t} \tag{3.5}$$

$$= \sum_{f,t} \left(v_{f,t} \log \frac{v_{f,t}}{\sum_r w_{f,r} h_{r,t}} - v_{f,t} + \sum_r w_{f,r} h_{r,t} \right), \tag{3.6}$$

which is called *generalized KL divergence*[4]. Both the Euclidean distance and the generalized KL divergence are instances of the family of β-divergences D_β, for which a generic NMF algorithm can also be derived [46]. In this framework, the Euclidean

[4]It differs from the regular KL divergence by the term $\sum_{f,t} -v_{f,t} + (\mathbf{WH})_{f,t}$, which ensures that the properties $D_{\mathrm{KL}} \geq 0$ and $D_{\mathrm{KL}} = 0 \iff \mathbf{V} = \mathbf{WH}$ do not only hold for probability distributions (for which the regular KL divergence is defined), but also for arbitrary non-negative data.

distance corresponds to the 2-divergence and the generalized KL divergence to the 1-divergence – thus, they will be denoted as D_2 and D_1 below.

The minimization of (3.4) or (3.6) is often performed by gradient-descent-like algorithms. Updates are usually performed alternatingly for $\mathbf{W}$ and $\mathbf{H}$,

$$\mathbf{W}^{(k+1)} = \mathbf{W}^{(k)} - \eta \nabla_{\mathbf{W}} D(\mathbf{W}^{(k)}), \tag{3.7}$$

$$\mathbf{H}^{(k+1)} = \mathbf{H}^{(k)} - \eta \nabla_{\mathbf{H}} D(\mathbf{H}^{(k)}), \tag{3.8}$$

where the 'matrix gradients' are defined as $\nabla_{\mathbf{W}} D = (\frac{\partial D}{\partial w_{f,r}})$ and $\nabla_{\mathbf{H}} D = (\frac{\partial D}{\partial h_{r,t}})$ and the free parameter η determines the step size. Obviously, after performing the gradient descent update, the non-negativity constraints can be violated. An easy solution is to perform *projected gradient descent* [115], which simply truncates the updates such that the parameters stay non-negative:

$$w_{f,r}^{(k+1)} = \max\left(w_{f,r}^{(k)} - \eta \frac{\partial D}{\partial w_{f,r}}(w_{f,r}^{(k)}), 0\right), \tag{3.9}$$

$$h_{r,t}^{(k+1)} = \max\left(h_{r,t}^{(k)} - \eta \frac{\partial D}{\partial h_{r,t}}(h_{r,t}^{(k)}), 0\right), \tag{3.10}$$

Note that this effectively results in gradient descent with a non-constant step size which depends on the current parameter values.

A more elegant solution that both preserves non-negativity and dispenses with the step size parameter η (which has to be tuned empirically) is to perform *multiplicative updates* of the form

$$\mathbf{W}^{(k+1)} = \mathbf{W}^{(k)} \otimes \frac{\nabla_{\mathbf{W}}^{-} D(\mathbf{W}^{(k)})}{\nabla_{\mathbf{W}}^{+} D(\mathbf{W}^{(k)})}, \tag{3.11}$$

$$\mathbf{H}^{(k+1)} = \mathbf{H}^{(k)} \otimes \frac{\nabla_{\mathbf{H}}^{-} D(\mathbf{H}^{(k)})}{\nabla_{\mathbf{H}}^{+} D(\mathbf{H}^{(k)})}, \tag{3.12}$$

where $\nabla_{\mathbf{W}}^{-}$, $\nabla_{\mathbf{W}}^{+}$, $\nabla_{\mathbf{H}}^{-}$ and $\nabla_{\mathbf{H}}^{+}$ denote a split of the matrix gradient into positive and negative terms, i.e., $\nabla_{\mathbf{W}} = \nabla_{\mathbf{W}}^{-} + \nabla_{\mathbf{W}}^{+}$, $\nabla_{\mathbf{H}} = \nabla_{\mathbf{H}}^{-} + \nabla_{\mathbf{H}}^{+}$, $\otimes$ denotes element-wise multiplication and the quotient line element-wise division.

To set this in relation to gradient descent, let us assume that (3.11) and (3.12) are equivalent to (3.7) and (3.7) for some (non-constant) step size $\eta_{f,r}^{(k)}$:

$$\mathbf{W}^{(k)} \otimes \frac{\nabla_{\mathbf{W}}^{-} D(\mathbf{W}^{(k)})}{\nabla_{\mathbf{W}}^{+} D(\mathbf{W}^{(k)})} = \mathbf{W}^{(k)} - (\eta_{f,r}^{(k)}) \otimes \nabla_{\mathbf{W}} D(\mathbf{W}^{(k)}). \tag{3.13}$$

Solving for $\eta_{f,r}^{(k)}$ yields

$$(\eta_{f,r}^{(k)}) = -\frac{\mathbf{W}^{(k)} \otimes \frac{\nabla_{\mathbf{W}}^{-} D(\mathbf{W}^{(k)})}{\nabla_{\mathbf{W}}^{+} D(\mathbf{W}^{(k)})} - \mathbf{W}^{(k)}}{\nabla_{\mathbf{W}}^{+} D(\mathbf{W}^{(k)}) - \nabla_{\mathbf{W}}^{-} D(\mathbf{W}^{(k)})} \tag{3.14}$$

$$= \frac{\mathbf{W}^{(k)}}{\nabla^{+}_{\mathbf{W}} D(\mathbf{W}^{(k)})}. \tag{3.15}$$

The solution for $\mathbf{H}^{(k)}$ is derived in analogy. Due to the non-negativity of the parameters, it is clear that $\eta^{(k)}_{f,r} > 0$, and thus every multiplicative update performs a gradient descent like step into the 'right direction', albeit with a step size that varies for each parameter. Still, from the above no convergence guarantee can be derived – whereas gradient descent is guaranteed to converge for small enough step size, $\eta^{(k)}_{f,r}$ in the above need not be small in general. Furthermore, depending on the choice of D, the choice of an appropriate 'split' into positive and negative parts might be ambiguous.

Let us now derive multiplicative update rules for minimizing either of the cost functions above. For the Euclidean distance, the partial derivatives w.r.t. the dictionary elements read

$$\frac{\partial D_2}{\partial w_{f',r'}} = \sum_t \left((\mathbf{WH})_{f',t} - v_{f',t} \right) h_{r',t}, \tag{3.16}$$

which can be rewritten in matrix gradient form as

$$\nabla_{\mathbf{W}} D_2 = (\mathbf{WH} - \mathbf{V})\mathbf{H}^{\intercal}. \tag{3.17}$$

The only obvious split leads to the well-known update rule,

$$\mathbf{W}^{(k+1)} = \mathbf{W}^{(k)} \otimes \frac{\mathbf{VH}^{\intercal}}{\mathbf{WHH}^{\intercal}}. \tag{3.18}$$

It is easy to derive a corresponding update rule for $\mathbf{H}$, considering that

$$\mathbf{V} = \mathbf{WH} \iff \mathbf{V}^{\intercal} = \mathbf{H}^{\intercal}\mathbf{W}^{\intercal}. \tag{3.19}$$

Thus, an update rule for $\mathbf{H}^{\intercal}$ can be directly derived from the update rule for $\mathbf{W}$. From the properties of the matrix transpose it follows that the following is an update rule for $\mathbf{H}$:

$$\mathbf{H}^{(k+1)} = \mathbf{H}^{(k)} \otimes \frac{\mathbf{W}^{\intercal}\mathbf{V}}{\mathbf{W}^{\intercal}\mathbf{WH}}. \tag{3.20}$$

Similar to the above, updates for minimizing the generalized KL divergence (3.6) can be derived. Splitting the gradient as follows:

$$\frac{\partial D_1}{\partial w_{f',r'}} = \sum_t -\frac{v_{f',t}}{(\mathbf{WH})_{f',t}} h_{r',t} + h_{r',t} \tag{3.21}$$

$$= \sum_t \left(1 - \frac{v_{f',t}}{(\mathbf{WH})_{f',t}} \right) h_{r',t} \tag{3.22}$$

leads to the update rules

$$\mathbf{W}^{(k+1)} = \mathbf{W}^{(k)} \otimes \frac{(\mathbf{V}/\mathbf{WH})\mathbf{H}^\intercal}{\mathbf{1}\mathbf{H}^\intercal}, \tag{3.23}$$

$$\mathbf{H}^{(k+1)} = \mathbf{H}^{(k)} \otimes \frac{\mathbf{W}^\intercal(\mathbf{V}/\mathbf{WH})}{\mathbf{W}^\intercal\mathbf{1}}. \tag{3.24}$$

where $\mathbf{1}$ denotes all-one matrices of the right dimensions. For both the Euclidean distance and generalized Kullback-Leibler divergence updates, convergence can be proven [112].

The above update rules are usually executed until convergence of the cost function has been reached. To limit the run-time for practical applications, typically a maximum number K of iterations is specified. The factors are often initialized from Gaussian random numbers (taking the absolute value).

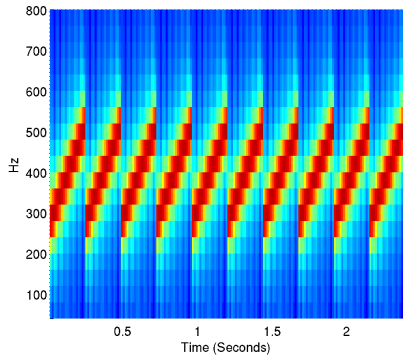
Figure 3.2 shows NMF and PCA decompositions of the magnitude spectrogram $\mathbf{X}$ of a periodic rising chirp signal[5]. In this example, NMF finds dictionary atoms corresponding to a single fundamental frequency, and activations model the temporal evolution of the fundamental frequency. Conversely, the representation found by PCA is hardly interpretable, apart from the fact that the activation matrix seems to capture different temporal resolutions. Despite the apparent advantage of the NMF representation, it has to be stated here that an NMF solution, in contrast to PCA or related methods, has no guaranteed properties apart from non-negativity – in particular, there is no unique solution, as dictionary atoms can be permuted or otherwise transformed by an invertible $R \times R$ matrix leading to alternative solutions fulfilling $\mathbf{V} \approx \mathbf{WH}$.

3.2.1 NMF for audio source separation

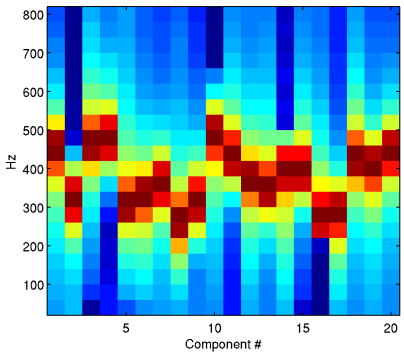
When applied to audio processing tasks, NMF is used to decompose a matrix of F-dimensional non-negative spectral features. In case of audio source separation, this is the spectrogram $\mathbf{M}$ of a mixture signal, $\mathbf{M} = [\mathbf{m}_1^\alpha \cdots \mathbf{m}_T^\alpha]$, where T is the number of frames and $\mathbf{m}_t \in \mathbb{R}_+^F$, $t = 1, \ldots, T$ are STFT magnitudes obtained from the time-domain mixture signal $m(\tau)$. The NMF constraints $w_{f,r} \geq 0$ and $h_{r,t} \geq 0$ incorporate the model assumptions that the dictionary atoms $\mathbf{w}_r = (w_{1,r}, \ldots, w_{F,r})^\intercal$ represent spectra of acoustic events, and that these are combined additively into an observed spectrogram, i.e., that cancellations in the time-frequency domain can be neglected.

To apply NMF for audio source separation, a surjective mapping $a : \{1, \ldots, R\} \to \{1, \ldots, S\}$ is assumed, i.e., each dictionary atom can be assigned to exactly one of S sources, and a source comprises one or multiple NMF components – e.g., an

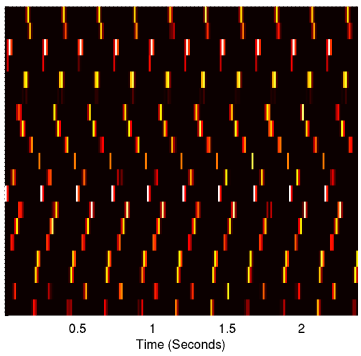
[5]For visualization purposes, the warping exponent $\alpha = 1/5$ is used for $\mathbf{X}$ and the NMF factor $\mathbf{W}$.



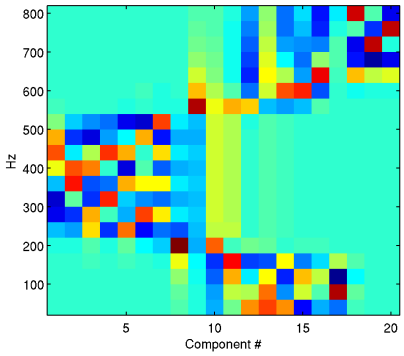
(a) **V**



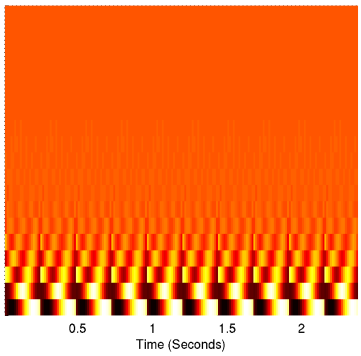
(b) **W** (NMF)



(c) **H** (NMF)



(d) **W** (PCA)



(e) **H** (PCA)

Figure 3.2: Visualization of the factors **W** and **H** computed by Non-Negative Matrix Factorization (NMF) and Principal Component Analysis (PCA) from the magnitude spectrogram with magnitude spectrogram (**V**) of a periodically rising chirp signal.

instrument playing multiple notes, or a speaker speaking a sequence of phonemes, which are modeled as dictionary atoms. This mapping can be found automatically,

e.g., by a classifier $y(\mathbf{w}_r, \mathbf{h}_r)$, or by manual inspection, resulting in a user-guided separation scheme. A well-known scenario where automatic assignment of components to sources can be applied effectively is the separation of percussive and harmonic instruments, as these have distinctive properties both in their spectral and temporal distributions [77, 170, 197].

Given a mapping a, without loss of generality one can assume a set of R_l non-negative dictionary atoms $\mathbf{w}_1^l, \cdots, \mathbf{w}_{R_l}^l$ for each source $l \in \{1, \ldots, S\}$, which constitute the dictionaries $\mathbf{W}^l = [\mathbf{w}_1^l \cdots \mathbf{w}_{R_l}^l]$. The non-negative factorization can then be written as[6]

$$\mathbf{M} \approx \mathbf{WH} = [\mathbf{W}^1 \cdots \mathbf{W}^S][\mathbf{H}^1; \cdots; \mathbf{H}^S]. \tag{3.25}$$

An approach related to Wiener filtering is typically used to reconstruct each source while ensuring that the source estimates sum to the mixture:

$$\hat{\mathbf{S}}^l = \frac{\mathbf{W}^l\mathbf{H}^l}{\sum_l \mathbf{W}^l\mathbf{H}^l} \otimes \mathbf{M}. \tag{3.26}$$

$\hat{\mathbf{S}}^l$ is then transformed back to the time-domain by inverse STFT and overlap-add.

3.2.2 Sparse regularization for NMF

Sparse regularization is useful for many machine learning problems in order to prevent overfitting. Here, Sparse NMF (SNMF) with L1 regularization is considered, which minimizes the cost function

$$D_\beta(\mathbf{V} \mid \widetilde{\mathbf{W}}\mathbf{H}) + \mu|\mathbf{H}|_1, \tag{3.27}$$

where $\widetilde{\mathbf{W}} = \left[\frac{\mathbf{w}_1}{\|\mathbf{w}_1\|} \cdots \frac{\mathbf{w}_R}{\|\mathbf{w}_R\|}\right]$ is the column-wise normalized version of $\mathbf{W}$, $|\mathbf{H}|_1 = \sum_{r,t} |h_{r,t}| = \sum_{r,t} h_{r,t}$ is the L1-norm of $\mathbf{H}$, and $\mu > 0$ is the weight of the sparsity penalty. Sparse regularization for NMF incorporates the model constraint that only a few spectra from the dictionary $\mathbf{W}$ should be present at each time instant, which makes sense if $\mathbf{W}$ contains spectra of audio events such as notes or phonemes.

Since the L1 sparsity constraint on $\mathbf{H}$ is not scale-invariant, it can be trivially minimized by scaling of the factors; by including the normalization in the cost function, the scale indeterminacy can be avoided. Note that for the reasons pointed out by Eggert and Körner [37], this is not the same as performing standard NMF optimization and scaling one of the factors to unit norm after each iteration – the latter can indeed lead to inferior performance [216].

A multiplicative update algorithm to optimize (3.27) for $\beta = 2$ was introduced by Eggert and Körner [37] and was later generalized to arbitrary $\beta \geq 0$ by O'Grady and Pearlmutter [140]. For $\beta = 1$, as mainly used in this thesis, each iteration $k = 1, \ldots, K$ of the algorithm performs the following steps:

[6] For simplicity, the notation $[\mathbf{a}; \mathbf{b}]$ for $[\mathbf{a}^\intercal \mathbf{b}^\intercal]^\intercal$ is used.

1. Calculate the L2-normalized dictionary atoms:

$$\mathbf{W}^{(k)} = \widetilde{\mathbf{W}}^{(k)} = \left[\frac{\mathbf{w}_1^{(k)}}{\| \mathbf{w}_1^{(k)} \|} \cdots \frac{\mathbf{w}_R^{(k)}}{\| \mathbf{w}_R^{(k)} \|} \right] \tag{3.28}$$

2. Update the activation matrix:

$$\mathbf{H}^{(k+1)} = \mathbf{H}^{(k)} \otimes \frac{\mathbf{W}^{(k)\mathsf{T}}(\mathbf{V}/\mathbf{W}^{(k)}\mathbf{H}^{(k)})}{\mathbf{W}^{(k)\mathsf{T}}\mathbf{1} + \mu} \tag{3.29}$$

3. Update the dictionary atoms (not preserving normalization):

$$\mathbf{W}^{(k+1)} = \mathbf{W}^{(k)} \otimes \frac{\left(\mathbf{V}/\mathbf{W}^{(k)}\mathbf{H}^{(k+1)}\right)\mathbf{H}^{(k+1)\mathsf{T}} + \mathbf{1}\left(\mathbf{1}\mathbf{H}^{(k+1)\mathsf{T}} \otimes \mathbf{W}^{(k)}\right) \otimes \mathbf{W}^{(k)}}{\mathbf{1}\mathbf{H}^{(k+1)\mathsf{T}} + \mathbf{1}\left(\left(\mathbf{V}/\mathbf{W}^{(k)}\mathbf{H}^{(k+1)}\right)\mathbf{H}^{(k+1)\mathsf{T}} \otimes \mathbf{W}^{(k)}\right) \otimes \mathbf{W}^{(k)}}. \tag{3.30}$$

Note that the multiplications with all-one matrices do not have to be carried out explicitly – in practice, they can be replaced by more efficient operations[7].

3.2.3 Supervised NMF

NMF as introduced above is an *unsupervised* algorithm, since no prior knowledge about the constituents of the signal is assumed. Yet, the perspectives of unsupervised NMF for speech source separation tasks, such as removal of non-stationary noise, are rather limited at the time of this writing. Compared to the case of percussive-harmonic separation, where simple spectral features such as harmonicity can be used to distinguish between speech and noise components [77, 197], this seems much harder in the case of speech and noise sources that are often similar in their spectral characteristics. As a result, virtually all NMF-based speech separation approaches nowadays use supervised initialization of (part of) the dictionary atoms based on training data, cf., e.g., [5, 96, 134, 185, 220], with only a few exceptions relying on only model-based constraints that might work in specific scenarios [113].

In *supervised NMF* [187], all $\mathbf{W}^l$ are learnt in advance from training data, and at run time only the activation matrices $\mathbf{H}^l = [\mathbf{h}_1^l \cdots \mathbf{h}_T^l]$, where $\mathbf{h}_t^l \in \mathbb{R}_+^{R_l}$, are estimated. In the supervised case, the activations for each frame are independent from the other frames ($\mathbf{m}_t \approx \sum_l \mathbf{W}^l \mathbf{h}_t^l$). Thus, source separation can be performed on-line and with latency corresponding to the window length plus the computation time to obtain the activations for one frame [96]. At test time, supervised NMF finds the optimal activations $\hat{\mathbf{H}}$ such that

$$\hat{\mathbf{H}} = [\hat{\mathbf{H}}^1; \cdots; \hat{\mathbf{H}}^S] = \arg\min_{\mathbf{H}} D(\mathbf{M} \mid \mathbf{W}\mathbf{H}) + \mu|\mathbf{H}|_1, \tag{3.31}$$

[7] In Matlab, the update (3.30) can be implemented efficiently using `bsxfun`.

where D is a cost function that is minimized when $\mathbf{M} = \mathbf{WH}$ (e.g., D_1 or D_2). The minimization (of D_1) is performed using the multiplicative update (3.29). For supervised NMF, $\mathbf{H}^{(0)}$ can be initialized deterministically instead of randomly, e.g., as $h^{(0)}_{r,t} = 1/R$ – in that initial solution, every observation is the average of the dictionary atoms.

3.2.4 Obtaining NMF dictionaries

A common approach [140, 166, 185] to obtaining dictionaries $\mathbf{W}^l$ is to fit an NMF model $\mathbf{W}^l\mathbf{H}^l$ to the spectrograms of source signals, $\mathbf{S}^l$, performing the optimization separately for each source l. In the case of SNMF, the following criterion is used:

$$\hat{\mathbf{W}}^l, \hat{\mathbf{H}}^l = \underset{\mathbf{W}^l,\mathbf{H}^l}{\arg\min}\, D_\beta(\mathbf{S}^l \mid \widetilde{\mathbf{W}}^l\mathbf{H}^l) + \mu|\mathbf{H}^l|_1, \tag{3.32}$$

where $\widetilde{\mathbf{W}}^l = \left[\frac{\mathbf{w}^l_1}{\|\mathbf{w}^l_1\|} \cdots \frac{\mathbf{w}^l_{R_l}}{\|\mathbf{w}^l_{R_l}\|}\right]$ is the column-wise normalized version of $\mathbf{W}^l$.

To obtain large dictionaries – e.g., if it is desired to cover multiple speakers and pronunciation variants in a speech dictionary – *exemplar-based* approaches have become popular [5, 52, 58, 157], where every atom corresponds to an observation of the source l in the training data. These avoid the peculiarities and complexity of SNMF training. There, given a short-time spectral representation $\mathbf{S}^l = \left[\mathbf{s}^l_1 \cdots \mathbf{s}^l_{T^l}\right]$ of training signals for source l, a subset $\mathcal{E} = \{e_1, \ldots, e_{R^l}\} \subset \{1, \ldots, T^l\}$ is chosen, and the NMF dictionary is simply defined as

$$\mathbf{W}^l = \left[\mathbf{s}^l_{e_1} \cdots \mathbf{s}^l_{e_{R^l}}\right]. \tag{3.33}$$

3.2.5 Semi-supervised NMF

In *semi-supervised NMF* [187], at test time the dictionary atoms for one source l, $\mathbf{W}^l$, are estimated alongside with the activations $\mathbf{H}$ of all dictionary atoms:

$$\hat{\mathbf{H}}, \hat{\mathbf{W}}^l = \underset{\mathbf{H},\mathbf{W}^l}{\arg\min}\, D(\mathbf{M} \mid \mathbf{WH}) + \mu|\mathbf{H}|_1. \tag{3.34}$$

Examples for the successful application of semi-supervised NMF include speech and noise separation, where either the speech dictionary is pre-defined and the noise dictionary is estimated [96, 142, 220, 224], or vice versa [34]. The latter variant can be transferred to the separation of the vocal artist from polyphonic music, assuming that an accompaniment dictionary can be built on the non-vocal parts before estimating the leading voice dictionary at test time [75].

It is also possible to extend semi-supervised NMF with model-based constraints for a specific application scenario. For example, the separation of the most prominent vocal source from polyphonic music (leading voice separation) has been formulated

as an optimization problem resembling semi-supervised NMF by Durrieu et al. [36]. In their approach, the mixture is represented as the additive combination of the vocal and accompaniment sources v and a in the STFT domain:

$$\mathbf{M} = \mathbf{S}^v + \mathbf{S}^a. \tag{3.35}$$

While the accompaniment is modeled directly by NMF, $\mathbf{S}^a = \mathbf{W}^a\mathbf{H}^a$, the vocals are assumed to follow an excitation-filter model in the frequency domain:

$$\mathbf{S}^v = \mathbf{S}^{v,E} \otimes \mathbf{S}^{v,F}. \tag{3.36}$$

Thus, each observed spectrum of the leading voice, $\mathbf{s}_t^v$ is the multiplication (corresponding to a convolution in the time domain) of a time-varying excitation $\mathbf{s}_t^{v,E}$ with a time-varying filter $\mathbf{s}_t^{v,F}$. Without further constraints, this model is too generic and could fit any spectral observation. Hence, a supervised NMF model is used for both $\mathbf{S}^{v,E}$ and $\mathbf{S}^{v,F}$ to restrict the set of possible excitations and filters,

$$\mathbf{S}^v = (\mathbf{W}^E\mathbf{H}^E) \otimes (\mathbf{W}^F\mathbf{H}^F), \tag{3.37}$$

where both $\mathbf{W}^E$ and $\mathbf{W}^F$ are pre-defined. Since the goal is to separate vocals, the excitation is assumed to be periodic. Consequently, the matrix $\mathbf{W}^E$ is initialized with dictionary atoms resembling harmonic combs with various fundamental frequencies (F0) in a given range, e.g., 100 to 800 Hz, equally spaced on the semitone scale. The matrix $\mathbf{W}^F$ contains overlapping Hann windows, such that $\mathbf{S}^{v,F}$ becomes a smooth filter. Fixing $\mathbf{W}^E$ and $\mathbf{W}^F$, the first step of the algorithm is to estimate the parameters $\mathbf{H}^E$, $\mathbf{H}^F$, $\mathbf{W}^a$ and $\mathbf{H}^a$ by multiplicative updates similar to the ones used for NMF. In a second step, F0 tracking is applied to $\mathbf{H}^E$ to eliminate jumps due to octave errors, and elements outside an interval around the tracked F0 are set to zero. Then, the above named parameters are re-estimated in a second step. Since multiplicative updates are used, the zeroed elements in $\mathbf{H}^E$ will remain zero, imposing the constraint of a smooth F0 curve.

After parameter estimation, the estimated spectrogram of the vocals $\hat{\mathbf{S}}^v$ is obtained by Wiener-like filtering, $\hat{\mathbf{S}}^v = \mathbf{S}^v/(\mathbf{S}^v + \mathbf{W}^a\mathbf{H}^a)$ with $\mathbf{S}^v$ being computed according to (3.37), and a time-domain signal is resynthesized by inverse STFT.

3.2.6 Efficient parallel implementation of NMF

The discussion of NMF in this thesis is concluded by describing efficient implementation of NMF using multi-core architectures, as in a previous study by the author and his advisor [217].

Two observations can be made regarding parallelization of NMF. First, since NMF is an iterative algorithm, it is highly sequential in nature: Parallelization can only performed within each iteration. Second, the lion's share of the computational

effort in each NMF iteration lies in the multiplicative update of the parameters. In turn, the complexity of the update rules is dominated by dense matrix-matrix multiplication: The asymptotic complexity of each iteration is $O(FRT)$.

Since there is virtually no restriction on the matrix dimensions (such as $F = R$, which would lead to a square $\mathbf{W}$ matrix), it is hard to apply specialized matrix-matrix multiplication algorithms with lower asymptotic complexity. However, parallel implementation of matrix-matrix multiplication on multi-core architectures, in particular Graphics Processing Units (GPUs), can be used to lower the actual computation time drastically. While this parallelization is a non-trivial problem – every entry of the result matrix consists of a scalar product, whose computation requires a parallel reduction –, there are well-developed 'off-the-shelf' libraries that lend themselves to this task, especially those provided by GPU vendors, such as NVIDIA's Compute Unified Basic Linear Algebra Subroutines (CUBLAS). Other operations required by the multiplicative update NMF algorithm include element-wise matrix operations (addition, multiplication, and division), which do require some 'hand-crafted' routines, but these are less critical for performance [8].

As an example, the update of the $\mathbf{H}$ matrix performed by the KL-NMF algorithm features an element-wise division of the $\mathbf{H}$ matrix by a vector of column sums –equivalent to the multiplication with the $\mathbf{1}$ matrix in (3.24). In the author's and his colleagues' GPU implementation of NMF in the openBliSSART library [212, 217], these operations are realized by means of Compute Unified Device Architecture (CUDA) *kernels*, i.e., small programs resembling instructions in a single instruction, multiple data (SIMD) architecture. Every kernel handles a data subset, and the data subsets as well as the kernel threads are instantiated and distributed automatically across the GPU cores by the CUDA library. If every kernel modifies a set of matrix entries that is disjoint from the others – in particular, if every kernel writes to exactly one entry of the result matrix – no synchronization is needed across kernels, making parallel implementation trivial and dispensing with expensive inter-thread communication. Using these 'multiplicative update kernels' enables running the NMF algorithm entirely on the GPU without time intensive memory transfers to the Central Processing Unit (CPU)'s memory. More details on the implementation of CUDA kernels can be found in [8].

In the following, experimental results obtained in [217] with GPU and CPU implementations of NMF will be presented. The speedup by GPU parallelization over CPU implementation, as well as the impact of single- or double-precision calculation on CPU and GPU on Real-Time Factor (RTF) and source separation performance, will be evaluated. The code used to achieve these results is publicly available in the openBliSSART toolkit[8]. As a strong baseline for efficient single-core CPU implementation of linear algebra operations, the open-source Basic Linear Algebra Subroutines (BLAS) implementation provided by the Automatically Tuned

[8] `http://openblissart.github.com/openBliSSART`

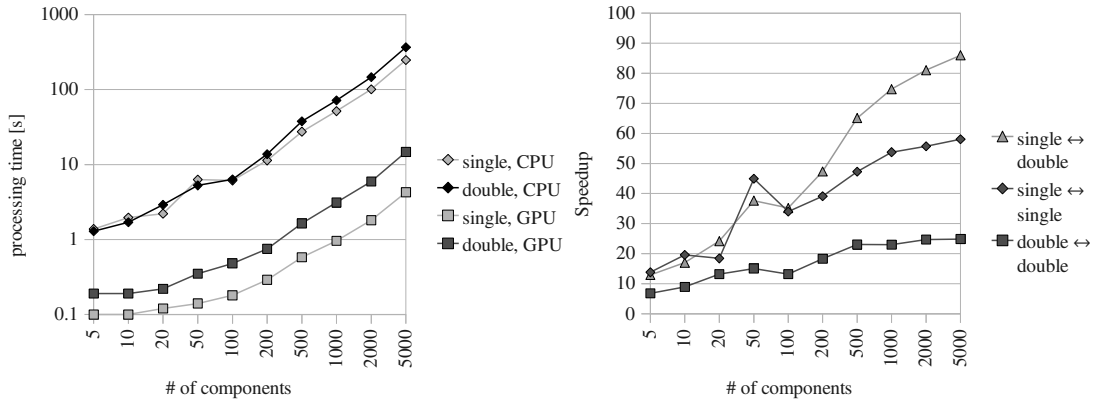


Figure 3.3: Processing time of the NMF algorithm (KL divergence) on a $500 \times 1\,000$ matrix by the number of NMF components; single / double floating point precision and CPU /GPU computation. Speedup for double or single precision on both CPU and GPU, and single precision GPU vs. double precision CPU.

Linear Algebra Software (ATLAS) project [233] is chosen. The ATLAS libraries can be considered industry standard and are at the core of Matlab's linear algebra capabilities. From the experience of the author, the ATLAS routines decrease the RTF by an order of magnitude for typical NMF applications, compared to a straightforward implementation of schoolbook matrix multiplication in C++.

All computation was performed on a desktop PC running Ubuntu Linux 10.04. The PC had an Intel Core2Quad CPU with 2.4 GHz clock frequency and 4 GB of RAM, and an NVIDIA GeForce GTX560 GPU with 336 CUDA cores – each with 810 MHz core and 160 MHz shader frequency – and 2 GB of RAM. CPU computation was performed using a single computation thread (i.e., using only one of the four cores), in order to minimize singular effects due to interferences with concurrent system processes, to reflect an end-user application where only part of the CPU computation power can be dedicated to audio processing, and to compare to a baseline with no parallelization across computation units. However, intra-core vectorized processing on the CPU is performed by the employed ATLAS library, exploiting the Intel Multimedia Extensions (MMX) and their follow-up technologies.

First, the influence of the computation architecture (CPU vs. GPU) on the RTFs is evaluated, using matrix dimensions encountered in typical applications in speech and music processing. It is of high interest to assess whether there is a 'break-even point' for GPU calculation: It is expected that due to the overhead introduced by CPU-to-GPU memory transfers and thread synchronization of parallel computations on the GPU, GPU can outperform CPU calculation only for 'sufficiently high' matrix dimensions. Furthermore, the impact of floating point precision is evaluated for both CPU and GPU, as current generation GPUs increasingly support double precision calculations – they are increasingly popular in scientific computing applications,

where the required precision depends on the application [244].

In Figure 3.3, the processing times of the NMF algorithm (KL divergence) on a $500 \times 1\,000$ matrix are shown. This corresponds to a $W = 1\,001$ point DFT (≈ 63 ms window size at 16 kHz sampling rate), i.e., $F = 500$, and 10 s signal length at 10 ms frame shift, i.e., $T = 1000$ signal frames. The number of NMF components R was varied from 5 to 5 000 to reflect NMF applications ranging from the extraction from a few number of simple sources (cf., e.g., [76]) to high-dimensional decomposition (cf., e.g., [57]). Comparing single vs. double precision, a consistent speedup from 1.9 up to 3.5 is observed for GPU computation, depending on R. In contrast, for CPU computation, the results are more mixed when using lower numbers of components; particularly, for $R = 50$, double precision is about 1.2 times faster than single precision. Furthermore, in case of single precision, the measured RTF is not linear in R. These phenomena can be attributed to peculiarities of the employed ATLAS library.

Measuring the speedup by using parallel GPU instead of CPU computation, a speedup of at least 6.8 is observed even for 'small' matrices ($R = 5$), indicating that the overhead by parallel computation is quite low in a typical audio source separation scenario. Speedups of up to 24.9 are reached for double precision and up to 58.1 for single precision. In other words, the maximum speedup obtained by using single-precision GPU computation instead of double-precision CPU computation is 86.0. As expected, speedups increase with the dimensionality R. In the result, by using single-precision GPU computation, real-time processing (i. e., a processing time below 10 s) is achievable even for 5 000 NMF components. Note that the above benchmark results have been obtained with the openBliSSART benchmark tool delivered with the source code to ensure best reproducibility.

To put these results in perspective, let us note that in [56], a speedup factor of 28 has been reported for SNMF classification (cf. Section 5.1.1) of a spectral matrix with $R = 8\,000$, $F = 690$ and $T = 182$, comparing double precision NMF computations on a CPU with a single precision implementation on a GPU. In that study, a slightly less performant GPU with only 675 MHz core frequency (instead of 810) was used; however, it can be assumed that the timing of the CPU computation refers to using both cores of the employed Intel Core2Duo CPU with 2.4 GHz (as is the default setting in Matlab), whereas here only one CPU core is used. Generally, the RTFs reported above are higher than in [56], where supervised NMF (Section 3.2.3, i.e., excluding the $\mathbf{W}$ updates) is applied, while the above performance measurements pertain to unsupervised NMF, including the update for $\mathbf{W}$. Furthermore, in an earlier study on parallelization of music source separation by NMF [8], an 18-fold speedup has been reported for $R = 30$, $F = 512$ and $T = 3\,445$ comparing single precision calculations in a single CPU thread (Intel Core i7 920) vs. a single precision CUDA implementation (NVIDIA GTX 280); this speedup is lower than the speedup gained in the above experiments for similar dimensions, but this can be attributed to the slightly different processing hardware used.

Table 3.1: Double and single precision in supervised speech separation by NMF: Separation performance in terms of SDR, SIR and SAR, as well as corresponding real-time factors (RTF) for CPU and GPU computation.

	Separation performance [dB]			RTF	
Precision	SDR	SIR	SAR	CPU	GPU
double	5.16	10.15	7.92	.522	.068
single	5.16	10.15	7.92	.937	.033

Second, experiments are performed in the framework of a somewhat realistic application: supervised speaker separation with NMF (cf. Section 3.2.3). The experimental protocol is defined in accordance with Smaragdis' study [185]. 12 random pairs of male and female speakers were selected from the TIMIT database [47]. For each pair, two randomly selected sentences of roughly equal length were mixed at an SNR of 0 dB. From the spectra in the other sentences spoken by each speaker, an NMF dictionary $\mathbf{W}$ was computed using $K = 250$ multiplicative update iterations. Through supervised NMF with $\mathbf{W}$, separated signals for both speakers were obtained, cf. (3.26). As quality measures, SDR, SIR and SAR were employed [200].

In this context, it is investigated whether using single instead of double floating point precision has a negative impact on separation quality. From Table 3.1, it is evident that this is not the case: In fact, the SDR, SIR and SAR values are identical for double and single precision up to the third decimal. This is in accordance with the findings of [56] for non-negative sparse classification. Interestingly, the matrix dimension in the speaker separation case ($R = 50$) coindices with a configuration where double precision is faster than single precision in CPU computation, confirming the singularity evident from Figure 3.3; this surprising result has been corroborated in multiple repetitions to cope with random fluctuations due to operating system CPU usage etc. Conversely, in GPU computation, the RTF can be halved by using single precision without decreasing the separation quality.

Overall, it can be concluded that GPU parallelization is highly rewarding in efficient NMF implementation, as first promising results from [56] and [8] could be corroborated in a larger scale study. RTFs smaller than one can be achieved for several thousand NMF components, motivating fully real-time processing on GPUs in the future – the crux is that in this case, no parallelization across time steps can be performed, as was done in the above experiments. In this context, it will also be of interest to derive efficient parallel implementations of *semi*-supervised real-time NMF algorithms (cf., e.g., [96]).

The exact speedups induced by parallel computation of NMF depend on a variety of external factors such as the number of double and single precision floating point units which can operate in parallel, the implementation of the employed BLAS

libraries and the instruction set architecture of the CPU and GPU. While separation quality is seemingly unaffected by floating point precision, using single precision is not necessarily faster for all configurations. Performance measurements in a real-life, end-user oriented setup featuring a state-of-the-art desktop PC – as in this article – usually imply that the CPU has a complex instruction set architecture (ISA) where performance of floating point operations strongly depends on the used machine instructions. This is in contrast to GPUs whose ISA is usually similar to the concept of reduced instruction set computer (RISC) architectures. Overall, it is believed that by providing the source code of the algorithms as well as the benchmarks, it will be straightforward for the research community to gain additional insights into the performance of NMF in different hardware setups.

3.3 Deep Neural Networks

In the following, DNNs will be introduced briefly. In contrast to NMF, which is a model-based algorithm that incorporates problem constraints such as non-negativity and additivity, DNNs are very generic models that are principally able to handle arbitrary features and recognition tasks. Still, they have been proven very successful in multi-source recognition, cf., e.g, [87, 183].

A DNN is functionally equivalent to the well-known Multi-Layer Perceptron (MLP). A K-layer DNN computes a non-linear function $\mathbf{y}(\mathbf{x})$

$$\mathbf{y} = \mathcal{H}^{(K)}\left(\mathbf{W}^{(K)}\mathcal{H}^{(K-1)}\left(\mathbf{W}^{(K-1)}\cdots\mathcal{H}^{(1)}\left(\mathbf{W}^{(1)}\mathbf{x}\right)\right)\right), \tag{3.38}$$

where $\mathbf{x}_t$ are the input features and $\mathcal{H}^{(k)}, k = 1, \ldots, K$ are activation functions applied to $\mathbf{a}^{(k)} = \mathbf{W}^{(k)}\mathbf{h}^{(k-1)} \in \mathbb{R}^{U(k)}$ with $U(k)$ being the number of *units* in layer k. The layers with indices $k = 1, \ldots, K-1$ are called *hidden layers*, so that the DNN from (3.38) is said to have $K-1$ hidden layers. In the above, we omit bias inputs for simplicity. The following element-wise activation functions are commonly used (where the layer index k is omitted for readability):

- *Identity*: $\mathcal{H}(\mathbf{a}) = \mathbf{a}$;
- *Half-wave*: $\mathcal{H}\left((a_1, \ldots, a_U)^\intercal\right) = (\max(a_1, 0), \ldots, \max(a_U, 0))^\intercal$;
- *Logistic*: $\mathcal{H}\left((a_1, \ldots, a_U)^\intercal\right) = (\sigma(a_1), \ldots, \sigma(a_U))^\intercal$ with $\sigma(x) = 1/(1+\exp(-x))$;
- *Hyperbolic tangent*: $\mathcal{H}\left((a_1, \ldots, a_U)^\intercal\right) = (\tanh(a_1), \ldots, \tanh(a_U))^\intercal$ with $\tanh(x) = 2\sigma(2x) - 1$.

The *hidden layer activations* in layer k are defined as $\mathbf{h}^k = \mathcal{H}(\mathbf{a}^k)$. Consequently, in case of identity and half-wave activation functions, the corresponding hidden layer units are often referred to as *linear* and *rectified linear*. An activation function

commonly used in the output layer is the *Softmax* function, which is the only common activation function that is not applied element-wise:

$$\begin{aligned} \mathcal{H}\left((a_1,\ldots,a_U)^\intercal\right) &= z^{-1}\cdot(\exp(a_1),\ldots,\exp(a_U))^\intercal, \\ z &= \sum_u \exp(a_u). \end{aligned} \tag{3.39}$$

It is easy to see that the above fulfills $h_u \in [0,1]$, $\sum_u h_u = 1$, $u = 1,\ldots,U$ and hence, h_u can be interpreted as pseudo-probabilities, which is useful, e.g., to have a DNN output class posteriors.

3.3.1 Gradient descent based training

The weights $\mathbf{W}$ of a DNN are trained by minimizing the following cost function:

$$E_{\mathcal{T}}(\mathbf{W}) = \sum_{t\in\mathcal{T}} E_t = \sum_{t\in\mathcal{T}} D(\mathbf{y}_t^* \mid \mathbf{y}(\mathbf{x}_t)), \tag{3.40}$$

where $\mathcal{T}$ denotes the set of indices belonging to training vectors $\mathbf{x}_t$ along with their ground truth labels $\mathbf{y}_t^*$. The choice of D depends on the application but also on the network structure, in particular the activation function $\mathcal{H}^{(K)}$ of the output layer. The most common criterion, which can be used for any activation function, is the squared deviation of the output from the training target,

$$E_{\mathcal{T}}(\mathbf{W}) = \sum_{t\in\mathcal{T}}\sum_f (y_{f,t}^* - y_{f,t})^2. \tag{3.41}$$

Since in case of a softmax activation function at the output layer, the outputs $\mathbf{y}_t$ can be interpreted as pseudo-probabilities, the cross-entropy of a target Probability Density Function (PDF) given the output PDF can be used:

$$E_{\mathcal{T}}(\mathbf{W}) = \sum_{t\in\mathcal{T}} H(\mathbf{y}_t^* \mid \mathbf{y}_t) = -\sum_{t\in\mathcal{T}}\sum_f y_{f,t}^* \log y_{f,t}. \tag{3.42}$$

Besides continuous-valued targets $\mathbf{y}_t^*$, one can also have discrete labels $c_t^* \in \{1,\ldots,L\}$ for L-way classification. In this case, the discrete label is usually 'coded' in a 1-of-L scheme, i.e., $y_{f,t}^* = 1$ for $f = c_t^*$ and 0 otherwise. Then, the cross-entropy function simplifies to

$$E_{\mathcal{T}}(\mathbf{W}) = -\sum_{t\in\mathcal{T}} \log y_{c_t^*,t}. \tag{3.43}$$

The error function $E_{\mathcal{T}}$ (3.40) is usually mininimized via gradient descent,

$$\mathbf{W}^{(q+1),(\cdot)} = \mathbf{W}^{(q),(\cdot)} - \eta\nabla E_{\mathcal{T}}\left(\mathbf{W}^{(q),(\cdot)}\right). \tag{3.44}$$

with training epoch q, learning rate $\eta > 0$ and $(\cdot)$ indicating that the update is performed for all layers simultaneously. In practice, since the scale of $E_{\mathcal{T}}$ depends on the cardinality of $\mathcal{T}$, and hence the choice of η would depend on the training set size, the cost function is normalized by the factor $1/|\mathcal{T}|$. Furthermore, often a momentum term [150] is added, which serves to minimize 'oscillations' of the cost function. This is done by treating the weight update $\Delta\mathbf{W}$ as the 'velocity' of the gradient descent process, which is updated in each iteration [191]:

$$\Delta\mathbf{W}^{(q+1),(\cdot)} = \mu\Delta\mathbf{W}^{(q),(\cdot)} - \eta\nabla E_{\mathcal{T}}\left(\mathbf{W}^{(q),(\cdot)}\right), \tag{3.45}$$

$$\mathbf{W}^{(q+1),(\cdot)} = \mathbf{W}^{(q),(\cdot)} + \Delta\mathbf{W}^{(q+1),(\cdot)}, \tag{3.46}$$

with $\Delta\mathbf{W}^{(0),(\cdot)} = \mathbf{0}$ and $0 < \mu \leq 1$ being the momentum coefficient.

In the computation of the gradient of the error function with respect to the weights, one can exploit the fact that because of the chain rule, the function composition in the computation of the DNN output (3.38) translates to the multiplication of the weight gradients per layer. It can be shown that for the gradient with respect to weights in layer k, it holds that

$$\nabla_{\mathbf{W}^{(k)}} E_{\mathcal{T}} = \sum_t \nabla_{\mathbf{W}^k} E_t = \sum_t \left(\partial E_t / \partial w_{i,j}^{(k)}\right) = \sum_t \left(h_{i,t}^{(k-1)} \delta_{j,t}^{(k)}\right), \tag{3.47}$$

with E_t being the error in timestep t, and the matrix of *deltas* being defined as

$$\boldsymbol{\Delta}_t^{(k)} = (\delta_{j,t}^{(k)}) = \mathbf{W}^{(k)\intercal}\left(\nabla_{\mathbf{a}}\mathcal{H}^{(k)}(\mathbf{a}_t^{(k)}) \otimes \left(\ldots\left(\mathbf{W}^{(K)\intercal}\left(\nabla_{\mathbf{a}}\mathcal{H}^{(K)}(\mathbf{a}_t^{(K)}) \otimes \nabla_{\mathbf{y}} E_t\right)\right)\right)\right). \tag{3.48}$$

The latter can be formulated as an iterative algorithm (*backpropagation*) that is initialized with the output layer deltas, i.e., the gradient of the error function for time step t with respect to the network outputs, $\nabla_{\mathbf{y}} E_t$, and computes the deltas for layer $k-1$ based on the deltas for layer k:

$$\boldsymbol{\Delta}_t^{(K+1)} := \nabla_{\mathbf{y}} E_t, \tag{3.49}$$

$$\boldsymbol{\Delta}_t^{(k)} := \mathbf{W}^{(k)\intercal}\left(\nabla_{\mathbf{a}}\mathcal{H}^{(k)}(\mathbf{a}_t^{(k)}) \otimes \boldsymbol{\Delta}_t^{(k+1)}\right). \tag{3.50}$$

For some activation functions, the gradient computation can be further simplified. For example, for the element-wise sigmoid function σ, it can be reduced to a simple transformation of the hidden layer activations:

$$\nabla_{\mathbf{a}}\sigma(\mathbf{a}^{(k)}) = \sigma(\mathbf{a}^{(k)}) \otimes (\mathbf{1} - \sigma(\mathbf{a}^{(k)})) = \mathbf{h}^{(k)}(\mathbf{1} - \mathbf{h}^{(k)}). \tag{3.51}$$

As these activations have to be computed anyway during the *forward pass*, i.e., the computation of (3.38), this significantly reduces the time and space complexity of the algorithm with respect to an arbitrary activation function.

3.3.2 Deep Neural Networks vs. Multi-Layer Perceptrons

All the above considerations about gradient descent based training hold equally for the MLP. Yet, the term DNN is commonly associated with a few recent advances in training schemes that make gradient descent based training of large models on large amounts of data practicable – a recent overview is given by Deng et al. [29].

For example, layer-wise training avoids training many weight parameters at once from a random initialization, easing the high-dimensional optimization problem. A simple, yet effective iterative implementation of layer-wise training of a K-layer DNN uses the weights of a $k-1$-layer DNN as initialization for the weights of a k-layer DNN ($k = 2, \ldots, K$), with only the weights in the k-th layer being randomly initialized [246]. Generative pre-training [83] is an example for a model-inspired method that particularly lends itself to training with scarce data. Overall, layer-wise training seems of little importance for very large training sets [29].

Another recent insight is that popular audio features such as MFCCs, which were partly motivated by the constraints of Gaussian mixture modeling in ASR (cf. Section 3.7.1), turn out to be less useful than simpler features such as 'raw' Mel filterbank outputs [131]. Some researchers believe that given 'low-level' input features, the network can – to some degree – extract its own higher-level features, with each hidden layer computing a higher-level representation [82]. However, in practice this ability is constrained by the chosen network topology (connections and activation functions), which can sometimes lead to counterintuitive results, such as rotations of the input features having a large impact on performance [131].

Yet another crucial point for the success of DNNs is that exploiting large amounts of training data has become practicable, mainly due to the increased prevalence of multi-core CPU and GPU architectures (cf. Section 3.5.5). In fact, for ASR it has been shown that DNNs provide significant advantages over previous models particularly with large-scale training data [181]. Large parts of DNN training can be easily parallelized on multi-core architectures: In the delta computation (3.48), all timesteps can be processed in parallel in a matrix-matrix multiplication. These multiplications are interleaved with element-wise operations, which can also be vectorized. The gradient computation (3.47) is easy to re-formulate as a matrix product, too. Thus, within the gradient computation sequential computation is only required between layers.

A precondition for effective large-scale training is to achieve a training time (for a given reduction of the cost function) that is sub-linear in the amount of training data. In practice, this can be achieved by stochastic gradient descent (SGD), where weight updates are performed on *mini-batches* $\mathcal{B}$ of size $|\mathcal{B}|$ with $\mathcal{B} \subset \mathcal{T}$, based on the assumption that each mini-batch is a representative sample of the whole training set. This leads to a set of cost functions for mini-batches $\mathcal{B}_i$,

$$E_{\mathcal{B}_i}(\mathbf{W}) = \sum_{t \in \mathcal{B}_i} D(\mathbf{y}(\mathbf{x}_t), \mathbf{y}_t^*), \tag{3.52}$$

such that $\sum_i E_{\mathcal{B}_i} = E_{\mathcal{T}}$. Hence, the mini-batch update becomes

$$\mathbf{W}^{(q+1),(\cdot)} = \mathbf{W}^{(q),(\cdot)} - \frac{\eta}{|\mathcal{B}_i|} \nabla E_{\mathcal{B}_i} \left(\mathbf{W}^{(q),(\cdot)} \right). \tag{3.53}$$

The above still involves significant sequential computation, as no two sets of mini-batches can be computed in parallel. To remedy this problem, the update can be slightly reformulated to allow a time delay $\Delta q \in \mathbb{N}$ between the current estimate of the weights and the estimate of the weights that the weight update calculation is based on:

$$\mathbf{W}^{(q+1),(\cdot)} = \mathbf{W}^{(q),(\cdot)} - \frac{\eta}{|\mathcal{B}_i|} \nabla E_{\mathcal{B}_i} \left(\mathbf{W}^{(q-\Delta q),(\cdot)} \right). \tag{3.54}$$

This concept leads to *asynchronous* gradient descent, which is suitable for large-scale distributed computations [27]. To improve the convergence of asynchronous gradient descent, an initial estimate of the parameters $\mathbf{W}$ can be obtained by running a few iterations of SGD ('warm start') [27].

Improving generalization As DNNs are very powerful models, they are likely to over-adapt to spurious patterns in the training data (over-fitting). Three common heuristics are frequently used in this thesis to alleviate over-fitting and improve generalization: (i) input noise, where white noise is added to the input features at the start of each training epoch (incorporating the model constraint that the network's output should invariant to small variations of the input features, and hence regularizing the regression / decision function); (ii) training set shuffling, where the order of training instances is determined randomly – this is important for SGD, effectively randomizing the order in which gradient steps corresponding to mini-batches are taken, and supposedly reducing the susceptibility to local minima in the error function; and (iii) early stopping, where the convergence criterion for the error function is not only evaluated on a training set, but also on a disjoint 'held-out' validation set. Optimal DNN training is still an active area of research, and more 'recipes' to improve generalization are presented in [133].

3.4 Regression-based source separation

Since a main application of machine learning in this thesis is audio source separation, and the methods presented in the previous section are trained in a supervised fashion (using labeled training data), it will now be examined how source separation can be formulated as a regression problem using supervised training.

To this end, let us first review the process of obtaining the separated source l by NMF as introduced in Section 3.2, cf. (3.26). For each time-frequency bin, this can be re-written as

$$\hat{s}^l_{f,t} = \frac{\sum_{r \in I_l} w_{f,r} h_{r,t}}{\sum_r w_{f,r} h_{r,t}} m_{f,t} \tag{3.55}$$

$$= \Pr(l \mid f,t) m_{f,t}, \tag{3.56}$$

where $\Pr(l \mid f,t)$ indicates the probability that source l is active in the time-frequency bin with frequency f and time t, and I_l is the set of indices of dictionary atoms belonging to source l. The second equality comes from a probabilistic interpretation of the filter term, which is guaranteed to be in $[0,1]$ due to the non-negativity constraints. The above formulation is known as *time-frequency masking*.

From this perspective, there is no evident reason why $\Pr(l \mid f,t)$ needs to be computed by NMF, apart from the conceptual advantage of being a model-based approach. Instead, any method for estimating class posteriors could be used to obtain $\Pr(l \mid f,t)$. In particular, it has been shown that binary classification by decision trees [61] and Support Vector Machines (SVMs) [111] can be effectively used for this task. However, as in many other areas of audio processing, there is currently an increasing trend towards DNN based source separation [87, 136, 229].

3.4.1 Supervised training for source separation

It is straightforward to derive a supervised training scheme for time-frequency masking, by training a system to predict an ideal mask for a wanted signal from features of a mixed signal. Specifically, to train single-channel source separation, a training corpus of source signals $s(\tau)$ and a parallel training corpus of mixtures $m(\tau)$ is assumed to be available[9]. From this, the residual signals $\overline{s}_l(\tau) = m(\tau) - s_l(\tau)$ are computed. Then, the STFTs $(\mathbf{s}_t^l)_t$ and $(\overline{\mathbf{s}}_t^l)_t \in \mathbb{R}_+^F$ of $s(\tau)$ and $\overline{s}(\tau)$ with F frequency bins are computed. From this, an *ideal ratio mask* (IRM) for source l in each training frame t is obtained as follows:

$$\mathbf{y}_t^l = \frac{\mathbf{s}_t^l}{\mathbf{s}_t^l + \overline{\mathbf{s}}_t^l}. \tag{3.57}$$

Now, supervised training can be applied to obtain a mapping $\mathbf{y}^l(\mathbf{m}_t), \mathbf{m}_t \mapsto \hat{\mathbf{y}}_t^l$, where $\hat{\mathbf{y}}_t^l$ is an estimated ratio mask for source l. The objective function to be minimized in supervised training is

$$E^{\mathsf{MA},l} = \sum_{f,t} D(\hat{y}_{f,t}^l, y_{f,t}^l), \tag{3.58}$$

where D is a distance measure and 'MA' stands for 'mask approximation'. In this thesis, the squared Euclidean distance, $D = D_2$, is used, which ensures that E^{MA} is closely related to the source separation objective (cf. Section 3.6).

At test time, using a full-resolution ratio mask $\mathbf{y}_t^l \in [0,1]^F$, the source spectrum $\hat{\mathbf{s}}_t^l$ is estimated as

$$\hat{\mathbf{s}}_t^l = \mathbf{y}_t^l \otimes \mathbf{m}_t, \tag{3.59}$$

[9] Since mixtures can be generated artificially from isolated training signals, this is not a stronger assumption than assuming the availability of training signals for all sources, such as as in supervised NMF.

where $\mathbf{m}_t$ is the spectrum of the mixture. From this, a time-domain signal $\hat{s}_l(\tau)$ is reconstructed using inverse DFT and overlap-add.

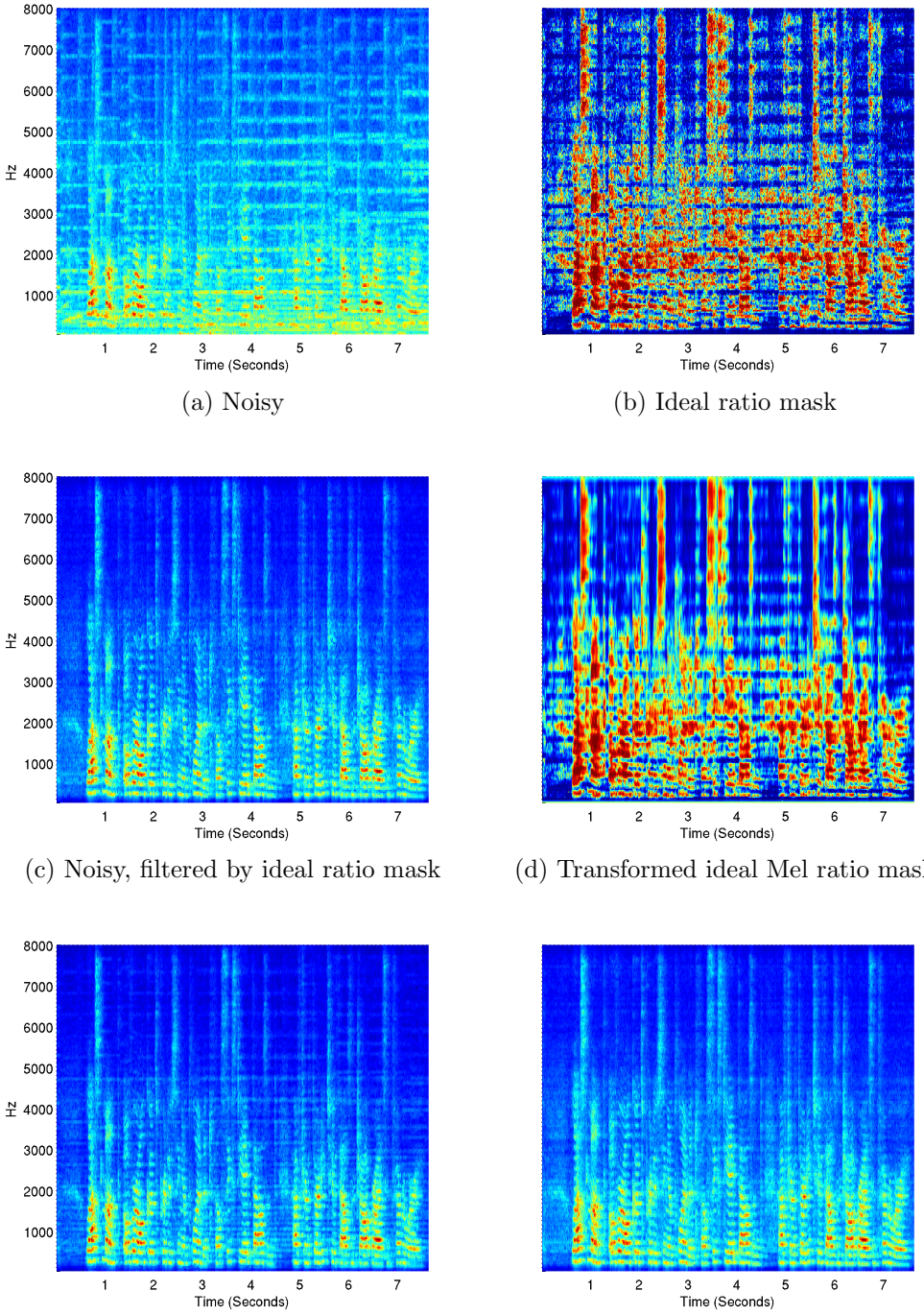
Considering the above, DNNs have a few convenient properties that can be exploited for training the task of source separation: First, the masking function for all frequency bins can be represented in a single (*multi-task*) model, which allows for efficient computation of $\mathbf{y}^l$ at test time. Besides, non-linearities in the feature representation can be handled effectively, thus allowing for (e.g., logarithmic) compression of the spectral magnitudes, which is considered useful in speech processing. Once trained, source separation only needs to evaluate (3.38) and (3.59), which is very efficient compared to iterative methods such as NMF. Finally, the backpropagation algorithm allows for switching the training objective easily, since only the gradient computation of the objective function with respect to the network output $\hat{\mathbf{y}}^l$ needs to be changed. This property can be exploited for discriminative training (cf. Section 3.6).

A fundamental limitation of the above approach is that even if the classifier / regressor is perfect, the resulting speech estimate might not equal the clean speech magnitude, $\Pr(l \mid f,t)m_{f,t} \neq s^l_{f,t}$. This is because due to cancellations in the time-frequency domain it might be the case that $m_{f,t} < s^l_{f,t}$ and hence the 'true mask' $s^l_{f,t}/m_{f,t}$ is not a probability. However, this is not a specific disadvantage of regression-based speech separation, but rather of time-frequency masking in general. Another problem is that only the magnitudes are modified. To get back to the time domain, a phase estimate is needed, which typically is – lacking better alternatives – the phase of the mixture. Consequently, even if the magnitudes of the source are estimated perfectly, the resulting time-domain signal $\hat{s}_l(\tau)$ will still be different from the source signal $s_l(\tau)$. Again, this is not a specific disadvantage of the algorithms presented in this thesis, but rather of any single-channel source separation algorithm working in the time-frequency domain, which also comprises popular speech and audio de-noising schemes such as unsupervised spectral subtraction [16], minimum statistics [124], etc.

Despite these theoretical limitations, it can often be observed that time-frequency masking using the 'ideal' ratio mask (3.57) delivers good separation quality. Figure 3.4 shows the result of applying such masks to a noisy speech signal from the CHiME-2013 corpus (cf. Section 2.2.1.1).

3.4.2 Mel-domain separation

The source separation algorithms considered so far operated in the full-resolution Discrete Fourier Transformation (DFT) domain. Consequently, for a window size W, $F = \lfloor W/2 + 1 \rfloor$ probabilities $\Pr(l \mid f,t)$ have to be estimated per frame. Especially in case of training data scarcity, it might be hard for a machine learning based model to provide accurate estimates for all the DFT bins. Thus, it might be useful to



(a) Noisy

(b) Ideal ratio mask

(c) Noisy, filtered by ideal ratio mask

(d) Transformed ideal Mel ratio mask

(e) Noisy, filtered by transformed Mel ideal ratio mask

(f) Clean

Figure 3.4: Time-frequency masking in the full-resolution STFT and the Mel domain applied to the utterance 050c0101 at 0 dB input SNR from the CHiME-2013 data.

reduce the dimensionality of the mask. Furthermore, coarser frequency masks often generalize better to unseen speakers and noise [5] – yet there is a trade-off with the achievable separation quality [5].

The most obvious choice of feature reduction, which is motivated from acoustic modeling in automatic speech recognition, is to employ Mel filter-banks. As explained in Section 3.1, this effectively results in a linear transformation of the features $\mathbf{m}_t$ by the matrix $\mathbf{B} = (b_{i,f}) \in \mathbb{R}^{B \times F}$, where $B < F$ is the number of Mel bins and $b_{i,f}$ is the weight of the DFT bin f in the i-th Mel bin. In this case, the mask $\mathbf{y}_t^l$ is also in the Mel domain:

$$y_{i,t}^{\mathsf{Mel},l} = \Pr(l \mid i,t) = \frac{(\mathbf{B}\mathbf{s}^l)_{i,t}}{(\mathbf{B}\mathbf{s}^l)_{i,t} + (\mathbf{B}\bar{\mathbf{s}}^l)_{i,t}}. \tag{3.60}$$

Directly applying this mask to the Mel features $\mathbf{B}\mathbf{m}_t$ is not advisable for speech reconstruction, as the matrix $\mathbf{B}$ is rectangular ($B < F$) and hence the corresponding linear transform is not invertible. Instead, the speech estimate can be computed by filtering the full-resolution spectrum,

$$\hat{\mathbf{s}}_t^l = (\mathbf{B}^\intercal \mathbf{y}_t^{\mathsf{Mel},l}) \otimes \mathbf{m}_t. \tag{3.61}$$

This is motivated by the special structure of $\mathbf{B}$, where $\sum_i b_{i,f} \leq 1$. In particular, for $i < B$ the falling slope of filter i overlaps with the rising slope of filter $i+1$, and $b_{i,f} = 1 - b_{i+1,f}$ in this region. With $y_{t,f}^{\mathsf{Mel},l} \in [0,1]$, it holds that $\mathbf{B}^\intercal \mathbf{y}_t^{\mathsf{Mel},l} \in [0,1]^F$, which can be interpreted as a full-resolution mask. Mathematically, the probability mass for each Mel-frequency bin, $\Pr(l \mid i,t)$ is redistributed to the DFT bins, $\Pr(l \mid f,t)$, by the weights $b_{i,f}$.

A more principled approach could be using a Wiener-like filter, where the Mel-domain speech and noise estimates are both transformed with the pseudo-inverse $\mathbf{B}^+$ of $\mathbf{B}$. However, the author's experiments with source separation on the CHiME-2013 data (cf. Section 4.2) showed that this approach did not perform better in terms of SDR than the above ad-hoc approach.

Figure 3.4 shows the ideal ratio mask, $[\mathbf{y}_t^l]_t$, the transformed Mel ideal ratio mask, $[\mathbf{B}^\intercal \mathbf{y}_t^{\mathsf{Mel},l}]_t$ ($B = 100$), as well as the corresponding reconstructions of the magnitude spectrogram of the speech, for an utterance from the CHiME-2013 data (development set, 050c0101) where speech is mixed with a music source at 0 dB SNR. It can be seen that the Mel transformation induces a smoothing of the mask in the higher frequencies. In particular, comparing the clean spectrogram with the Mel-filtered spectrogram and the full-resolution filtered spectrogram, it is evident that both filters preserve the speech information very well, while the Mel filter leaves some audible musical noise, as can be seen from the harmonics in the spectrogram. This is expected, as the low-resolution Mel filter cannot separate harmonics of speech and noise that lie close together. Still, in terms of objective measures, the Mel transformation results in only a slight reduction of the separation strength (15.9 dB vs. 15.3 dB SDR), while reducing the amount of filter parameters by 50 % ($B/F = 100/201$). The latter

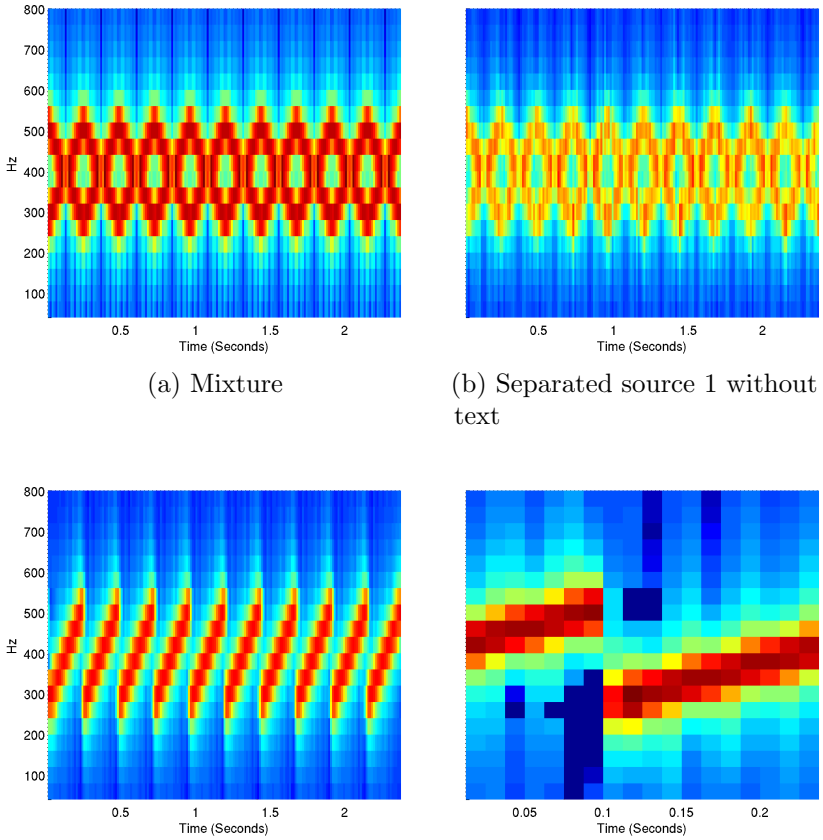
could lead to more reliable parameter estimation by machine learning, a hypothesis that will be verified in Section 4.2, along with a comparative evaluation of various regressors to estimate the masks.

3.5 Context-sensitive methods

So far, STFT-based features have been considered as input for the source separation algorithms. As a consequence, the algorithms are forced to discriminate sources based on a short-term observation, with length corresponding to W/F_s. This context size, however, is not always sufficient, as can be easily seen from a simple example, which is inspired by the work of Mohammadiha et al. [132]. Figure 3.5 shows the superposition $m(\tau) = s_1(\tau) + s_2(\tau)$ of rising and falling chirp signals $s_1(\tau)$ and $s_2(\tau)$. Intuitively, a source separation algorithm that has source models based on the original signals $s_1(\tau)$ and $s_2(\tau)$ should be able to robustly separate the mixture. Yet, supervised NMF, where dictionaries $\mathbf{W}^1$ and $\mathbf{W}^2$ are trained on the STFT spectrograms $\mathbf{S}^1$ and $\mathbf{S}^2$ of the original chirp signals ($R_1 = R_2 = 20$), fails to separate the mixture (Figure 3.5b), effectively resulting in two virtually identical source estimates, $\hat{\mathbf{S}}^1 \approx \hat{\mathbf{S}}^2$. This surprising result is easy to understand when examining the estimated dictionary $\mathbf{W}^1$: It will contain only small-band spectra, which can represent the short-term observations in both $\mathbf{S}^1$ and $\mathbf{S}^2$ perfectly. In the following, a simple solution to remedy this problem based on *frame stacking* is introduced, before more advanced methods are outlined.

In a frame stacking approach, the observation super-vector $\mathbf{m}'_t$ at time t corresponds to the observations $[\mathbf{m}_{t-T_\mathrm{L}}; \cdots ; \mathbf{m}_t; \cdots ; \mathbf{m}_{t+T_\mathrm{R}}]$ where T_L and T_R are the left and right context sizes. Missing observations at the beginning and end of the data can be replaced by copying the first and last observations, i.e., $\mathbf{m}_{t:t<0} := \mathbf{m}_1$ and $\mathbf{m}_{t:t>T} := \mathbf{m}_T$. The above is a generic approach that can be applied to any recognition algorithm. In particular, DNNs are usually trained using similar temporal context windows [82]. In case of NMF frame stacking [57], each dictionary atom will also correspond to a spectral super-vector. Consequently, dictionaries are trained on matrices $\mathbf{S}^{l\prime}$ containing spectral super-vectors of the separated sources in their columns, and the Wiener-like filter (3.26) also yields a sequence of super-vectors, each of the form $[\hat{\mathbf{s}}^l_{t-T_L}; \cdots ; \hat{\mathbf{s}}^l_{t-T_R}]$. From these, one can either just output the 'center frame' $\hat{\mathbf{s}}^l_t$, or average the source estimates within the window [57].

In the example, when dictionaries $\mathbf{W}^{1\prime}$ and $\mathbf{W}^{2\prime}$ are trained on $\mathbf{S}^{1\prime}$ and $\mathbf{S}^{2\prime}$, and supervised NMF is applied to $\mathbf{M}'$, using context sizes of $T_L = T_R = 9$, the source signals can be reconstructed as expected (Figure 3.5c). This is because every dictionary atom (such as the one displayed in Figure 3.5d, 'reshaped' into a matrix with $T_L + T_R + 1$ columns) now describes the temporal evolution of its corresponding source signal. For readability, subsequently the $'$ is dropped, and it is assumed that it is clear from context whether features correspond to stacked column vectors.



(a) Mixture

(b) Separated source 1 without context

(c) Separated source 1 with context

(d) Basis atom with context

Figure 3.5: Importance of context in source separation: Mixture of rising and falling chirp signals, separated by NMF without and with context.

3.5.1 Matrix deconvolution

A disadvantage of super-vector approaches in general is that the model is 'blind' to the information loss which occurs by 'flattening' structured data. In the case of frame stacking in NMF, the temporal information within the super-vectors is lost. In practice, this can result in an unnecessary blow-up of the model size. To understand why, let us again consider the case of a periodically rising chirp signal, where the period is significantly longer than the frame shift $\Delta\tau$. Intuitively, a context-sensitive basis representation should be able to model this signal with a single time-dependent dictionary atom, which captures the rising period. However, it is easy to see that this is not possible with frame stacking - since every observation corresponds to a sliding window, which in turn corresponds to a cyclic shift of the rising period of the chirp, every possible cyclic shift needs to be modeled by an additional dictionary atom, such as the one shown in Figure 3.5d.

A much more compact signal model can be obtained if time shifts are taken into account directly by the mixing model, an argument already put forth for the case of speech recognition by Hurmalainen et al. [88]. This consideration leads to *non-negative matrix deconvolution* (NMD) or *convolutive NMF* [184]. There, the observed spectrogram $\mathbf{M}$ is modeled as a convolution of 'basis' spectrograms $\mathbf{w}_r^l = (w_{r,f,p}^l)_{f,p} \in \mathbb{R}_+^{F \times P}$, each spreading over P frames of context, with the activation matrix $\mathbf{H}$:

$$m_{f,t} \approx \sum_{p=0}^{P-1} \sum_{l=1}^{S} \sum_{r} w_{r,f,p}^l h_{r,t-p+1}^l \tag{3.62}$$

In matrix form, this can be written as

$$\mathbf{M} \approx \sum_{p=0}^{P-1} \sum_{l=1}^{S} \mathbf{W}^l(p) \overset{p\rightarrow}{\mathbf{H}} = \sum_{p=0}^{P-1} \mathbf{W}(p) \overset{p\rightarrow}{\mathbf{H}} \tag{3.63}$$

with

$$\mathbf{W}^l(p) = \begin{bmatrix} w_{1,1,p}^l & \cdots & w_{R_l,1,p}^l \\ \vdots & \ddots & \vdots \\ w_{1,F,p}^l & \cdots & w_{R_l,F,p}^l \end{bmatrix}, \tag{3.64}$$

the concatenated dictionary $\mathbf{W}(p) = \left[\mathbf{W}^1(p) \cdots \mathbf{W}^S(p)\right]$ and the shift operator $\overset{p\rightarrow}{\cdot}$ being defined as

$$\overset{p\rightarrow}{\mathbf{A}} := \begin{cases} \left[\mathbf{0}^{M \times p}\,,\, \mathbf{A}_{:,1:T-p}\right] & p > 0 \\ \mathbf{A} & p = 0 \end{cases}. \tag{3.65}$$

In analogy to NMF, supervised and semi-supervised schemes with NMD dictionary learning, sparse NMD, and exemplar-based NMD can be derived [88, 91, 140, 185, 211]. In turn, these are based on multiplicative update rules for obtaining a dictionary $\mathbf{W}(p)$ and activations $\mathbf{H}$ from a spectrogram $\mathbf{X}$ which represents a mixture or a source. These rules are given in matrix form as follows:

$$\mathbf{W}(p)^{(k+1)} = \mathbf{W}(p)^{(k)} \otimes \frac{(\mathbf{X} \otimes (\hat{\mathbf{X}})^{\beta-2})(\overset{p\rightarrow}{\mathbf{H}})^\intercal}{(\hat{\mathbf{X}})^{\beta-1}(\overset{p\rightarrow}{\mathbf{H}})^\intercal}, \tag{3.66}$$

$$h_{j,t}^{(k+1)} = h_{j,t}^{(k)} \frac{1}{P(t)} \sum_{p=0}^{P(t)-1} \left[\frac{\mathbf{W}(p)^\intercal (\mathbf{X} \otimes (\hat{\mathbf{X}})^{\beta-2})}{\mathbf{W}(p)^\intercal (\hat{\mathbf{X}})^{\beta-1}} \right]_{j,t}^{\overset{\leftarrow p}{}}. \tag{3.67}$$

In the above, $\hat{\mathbf{X}}$ denotes the reconstructed spectrogram which is computed in analogy to (3.63), and the 'left shift' operator $\overset{\leftarrow p}{\cdot}$ is defined as

$$\overset{p\rightarrow}{\mathbf{A}} := \begin{cases} \left[\mathbf{A}_{:,p+1:T}\,,\, \mathbf{0}^{M \times p}\right] & p > 0 \\ \mathbf{A} & p = 0 \end{cases}, \tag{3.68}$$

$j = 1, \ldots, R$, $t = 1, \ldots, T$ and $P(t) := \min\{P, T - t + 1\}$. β is the parameter of the β-divergence cost function (cf. Section 3.2) defined in analogy to NMF. The above formulation [217] avoids the divisions by zero in the rightmost columns of the right hand side operands, which occur in the original matrix formulation of the algorithms [185, 208]. A Wiener-like filter equation for estimating the source l in the NMD model can be derived in analogy to NMF (3.26):

$$\hat{\mathbf{S}}^l = \frac{\sum_{p=0}^{P-1} \mathbf{W}^l(p) \overset{p\rightarrow}{\mathbf{H}}}{\sum_{p=0}^{P-1} \sum_l \mathbf{W}^l(p) \overset{p\rightarrow}{\mathbf{H}}} \otimes \mathbf{M}. \tag{3.69}$$

Efficient implementation of NMD It is beneficial to formulate NMD update rules in 'matrix form' in order to exploit linear algebra routines employing vectorization (cf. Section 3.2.6). However, if implemented naïvely, the shift operators introduce additional operations and increase memory usage. Hence, it will be shown how to eliminate them completely by reducing multiplications with shifted matrices to multiplication of submatrices, as proposed by the author and his advisor in [217]. This is in contrast to [22] where it was suggested to use special Matlab functions to eliminate the shifts.

Observe that the shift operators are always used within a matrix-matrix multiplication, introducing zeros into one of the factors: Thus, the shifting can be 'simulated' by adjusting the summation ranges in the scalar products used in matrix multiplication. This allows an easy and very efficient implementation without having to compute (and store) shifted versions of the matrices, or submatrices: The BLAS standard supports submatrix multiplications directly on the memory blocks corresponding to the full matrices. Defining $\tilde{\mathbf{X}} := \mathbf{X} \otimes \hat{\mathbf{X}}^{\beta-2}$, the numerator of the rule (3.66) can be transformed using

$$\begin{aligned} \tilde{\mathbf{X}} (\overset{p\rightarrow}{\mathbf{H}})^\intercal &= \tilde{\mathbf{X}} \left[\mathbf{0}^{R \times p} \, \mathbf{H}_{:,1:T-p}\right]^\intercal \\ &= \tilde{\mathbf{X}}_{:,p+1:T} \mathbf{H}^\intercal_{:,1:T-p} \quad , \end{aligned} \tag{3.70}$$

and the numerator of the rule (3.67) can be reformulated using

$$\begin{aligned} \mathbf{W}(p)^\intercal \overset{\leftarrow p}{\tilde{\mathbf{X}}} &= \mathbf{W}(p)^\intercal \left[\tilde{\mathbf{X}}_{:,p+1:T} \, \mathbf{0}^{F \times p}\right] \\ &= \left[\left(\mathbf{W}(p)^\intercal \tilde{\mathbf{X}}_{:,p+1:T}\right) \, \mathbf{0}^{F \times p}\right] \quad . \end{aligned} \tag{3.71}$$

The denominators of the rules can be transformed accordingly. Finally, the shifts in the computation of (3.63) can be eliminated by exploiting the equality

$$\begin{aligned} \mathbf{W}(p) \overset{p\rightarrow}{\mathbf{H}} &= \mathbf{W}(p) \left[\mathbf{0}^{R \times p} \, \mathbf{H}_{:,1:T-p}\right] \\ &= \left[\mathbf{0}^{F \times p} \; (\mathbf{W}(p) \mathbf{H}_{:,1:T-p})\right] \quad , \end{aligned} \tag{3.72}$$

thereby eliminating all shift operators from the algorithm.

3.5.2 Deep recurrent neural networks

Since audio is sequential, it is not surprising that in recent years, sequence learners such as RNNs have seen a resurgence in popularity for speech and music processing tasks [15, 17, 66, 67, 87, 209]. The combination of deep structures with temporal recurrence for sequence learning yields so-called Deep Recurrent Neural Networks (DRNNs) [67]. The function computed by a K-layer DRNN can be defined by the following iteration[10] (*forward pass*) for $k = 1, \ldots, K-1$ and $t = 1, \ldots, T$:

$$\mathbf{h}_0^{(1,\ldots,K-1)} = \mathbf{0}, \tag{3.73}$$

$$\mathbf{h}_t^{(0)} = \mathbf{x}_t, \tag{3.74}$$

$$\mathbf{h}_t^{(k)} = \mathcal{H}^{(k)}(\mathbf{W}^{(k-1,k)}[\mathbf{h}_t^{(k-1)}; 1] + \mathbf{W}^{(k,k)}\mathbf{h}_{t-1}^{(k)}), \tag{3.75}$$

$$\mathbf{y}_t = \mathcal{H}^{(K)}(\mathbf{W}^{(K-1,K)}[\mathbf{h}_t^{(K-1)}; 1]). \tag{3.76}$$

In the above, $\mathbf{h}_t^{(k)}$ denotes the hidden feature representation of time frame t in the level k units, where $k = 0$ represents the input layer (3.74) and $k = K$ the output layer. $\mathbf{W}^{(k-1,k)}$ and $\mathbf{W}^{(k,k)}$ denote the feed-forward and recurrent weight matrices at layer k, where the feed-forward part also includes the bias weights for simplicity.

To train RNNs, Backpropagation Through Time (BPTT) is typically used. This algorithm is conceptually similar to 'unfolding' the recurrent connections into a partially connected KT-layer DNN where connections exist both across consecutive timesteps and consecutive layers, and weight parameters are tied across timesteps, then performing standard backpropagation. However, this approach suffers from a vanishing or exploding gradient for larger T, making the optimization difficult [10]. As a result, RNNs are often not able to outperform DNNs in practical speech processing tasks [87, 230]. One of the oldest, yet still most effective[11] solutions proposed to remedy this problem is to add structure to the RNN following the Long Short-Term Memory (LSTM) principle as defined in [59, 85]. Formally, in an LSTM-DRNN the mapping from a sequence of input features $\mathbf{x}_t$ to outputs $\mathbf{y}_t$, is defined by the following iteration for layers $k = 1, \ldots, K-1$ and timesteps $t = 1, \ldots, T$:

$$\mathbf{h}_0^{(1,\ldots,K-1)} = \mathbf{0}, \mathbf{c}_0^{(1,\ldots,K-1)} = \mathbf{0}, \tag{3.77}$$

$$\mathbf{h}_t^{(0)} = \mathbf{x}_t, \tag{3.78}$$

$$\mathbf{f}_t^{(k)} = \mathcal{G}\big(\mathbf{W}^{f,(k-1,k)}[\mathbf{h}_t^{(k-1)}; 1] + \mathbf{W}^{f,(k,k)}\mathbf{h}_{t-1}^{(k)} + \mathbf{W}^{f,(k),\mathsf{p}}\mathbf{c}_{t-1}^{(k)}\big), \tag{3.79}$$

$$\mathbf{i}_t^{(k)} = \mathcal{G}\big(\mathbf{W}^{i,(k-1,k)}[\mathbf{h}_t^{(k-1)}; 1] + \mathbf{W}^{i,(k,k)}\mathbf{h}_{t-1}^{(k)} + \mathbf{W}^{i,(k),\mathsf{p}}\mathbf{c}_{t-1}^{(k)}\big), \tag{3.80}$$

$$\mathbf{c}_t^{(k)} = \mathbf{f}_t^{(k)} \otimes \mathbf{c}_{t-1}^{(k)}$$

[10] There are other possibilities to construct DRNNs, which is an emerging subject at the time of this writing, cf., e.g., [146].

[11] For example, at the time of this writing, LSTM-DRNNs set the benchmark [67] on the TIMIT phoneme recognition task [47].

$$+ \mathbf{i}_t^{(k)} \otimes \mathcal{H}\big(\mathbf{W}^{c,(k-1,k)}[\mathbf{h}_t^{(k-1)}; 1] + \mathbf{W}^{c,(k,k)}\mathbf{h}_{t-1}^{(k)}\big), \tag{3.81}$$

$$\mathbf{o}_t^{(k)} = \mathcal{G}\big(\mathbf{W}^{i,(k-1,k)}[\mathbf{h}_t^{(k-1)}; 1] + \mathbf{W}^{i,(k,k)}\mathbf{h}_{t-1}^{(k)} + \mathbf{W}^{i,(k),\mathsf{p}}\mathbf{c}_t^{(k)}\big), \tag{3.82}$$

$$\mathbf{h}_t^{(k)} = \mathbf{o}_t^{(k)} \otimes \mathcal{H}(\mathbf{c}_t^{(k)}), \tag{3.83}$$

$$\mathbf{y}_t = \mathcal{H}^{(K)}\big(\mathbf{W}^{(K-1,K)}[\mathbf{h}_t^{(K-1)}; 1]\big). \tag{3.84}$$

Again, $\mathbf{h}_t^{(k)}$ denotes the hidden feature representation of time frame t in the layer k units ($k = 0$: input layer). Analogously, $\mathbf{c}_t^{(k)}$, $\mathbf{f}_t^{(k)}$, $\mathbf{i}_t^{(k)}$, and $\mathbf{o}_t^{(k)}$ denote the dynamic cell state, forget gate, input gate, and output gate activations. The hidden layer activations correspond to the state variables, 'squashed' by the activation function and scaled by the output gate activations (3.83). $\mathbf{W}^{\cdot,(k-1,k)}$ and $\mathbf{W}^{\cdot,(k,k)}$ denote feed-forward and recurrent weight matrices at layer k ($k = K$: output layer). The superscript $\cdot$ is used to refer to cell states (c), forget gates (f), input gates (i), and output gates (o). $\mathbf{W}^{\cdot,(k),\mathsf{p}}$ denote the diagonal matrices of *peephole* weights [59], which provide a connection from the memory cell to the gate units with scalar weight. $\mathcal{H}$ is the cell activation function (typically tanh) and $\mathcal{G}$ is the gate activation function (typically σ). $\mathcal{H}^{(K)}$ is, again, an arbitrary output layer activation function.

In comparison to a standard RNN, the computation of $\mathbf{h}_t^{(k)}$ is now performed by a differentiable function $\mathcal{L}^{(k)}(\mathbf{h}_t^{(k)}; \mathbf{h}_{t-1}^{(k)})$ which performs 'soft' (differentiable) versions of read, write, and delete operations on a state variable $\mathbf{c}_t^{(k)}$. The latter is implemented as a recurrent unit with weight 1, allowing the RNN to exploit an unbounded amount of context. It can be shown that the LSTM approach avoids the vanishing gradient problem, thus allowing to effectively train DRNNs using gradient descent, and being able to learn long-term dependencies.

Figure 3.6 shows a visualization of a single LSTM cell (index i in layer k), which calculates its hidden activation $h_{i,t}^{(k)}$ from $\mathbf{h}_t^{(k-1)}$ and $\mathbf{h}_{t-1}^{(k)}$. $c_{i,t}^{(k)}$, $i_{i,t}^{(k)}$, $o_{t,i}^{(k)}$, $f_{i,t}^{(k)}$ denote the state, input gate, output gate, and forget activation of the cell i in layer k.

3.5.3 Backpropagation for LSTM-RNNs

Gradient descent for LSTM-RNNs is implemented by adapting BPTT accordingly. For a deep LSTM-RNN with $K - 1$ hidden LSTM layers and a single feedforward output layer, which is the configuration used for the experiments in this thesis, first the deltas and weight updates are computed for the output layer according to (3.49) and (3.47) substituting $k \mapsto K$.

Then, in order to get the deltas $\boldsymbol{\Delta}$ for a sequence with T timesteps, the LSTM-RNN training algorithm executes the following iteration for $t = T, \ldots, 1$ and $k = K - 1, \ldots, 1$, i.e., exactly in reverse order as the RNN forward pass described above:

$$\boldsymbol{\Delta}_{T+1}^{\cdot,(1,\ldots,K-1)} = \mathbf{0}, \tag{3.85}$$

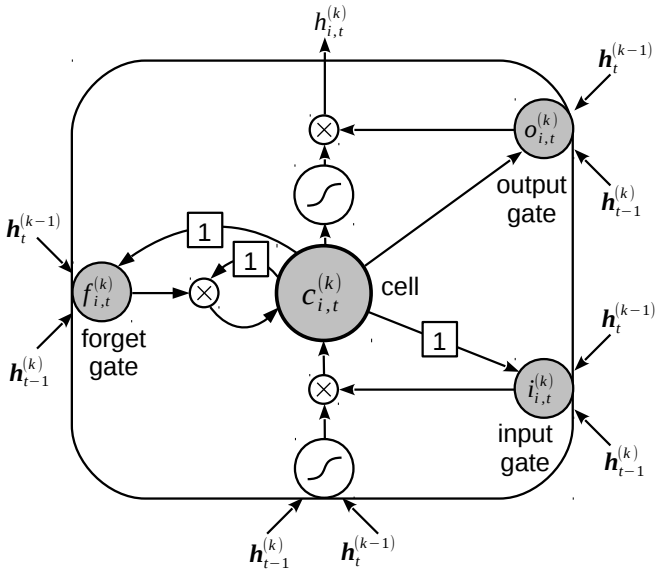


Figure 3.6: Visualization of the i-th cell in the k-th layer of a deep LSTM-RNN. Arrows denote data flow and 1 denotes a delay of one timestep.

$$
\begin{aligned}
\mathbf{e}_t^{(k)} =& \boldsymbol{\Delta}_t^{(k+1)} \\
&+ \mathbf{W}^{c,(k,k)} \boldsymbol{\Delta}_{t+1}^{c,(k)} + \mathbf{W}^{i,(k,k)} \boldsymbol{\Delta}_{t+1}^{i,(k)} \\
&+ \mathbf{W}^{f,(k,k)} \boldsymbol{\Delta}_{t+1}^{f,(k)} + \mathbf{W}^{o,(k,k)} \boldsymbol{\Delta}_{t+1}^{o,(k)}, && (3.86) \\
\boldsymbol{\Delta}_t^{o,(k)} =& \mathcal{G}'(\mathbf{a}_t^{o,(k)}) \otimes \mathcal{H}(\mathbf{c}_t^{(k)}) \otimes \mathbf{e}_t^{(k)}, && (3.87) \\
\mathbf{e}_t^{c,(k)} =& \mathbf{o}_t^{(k)} \otimes \mathcal{H}'(\mathbf{c}_t^{(k)}) \otimes \mathbf{e}_t^{(k)} \\
&+ \mathbf{f}_t^{(k)} \mathbf{e}_{t+1}^{c,(k)} \\
&+ \mathbf{W}^{i,(k),\mathsf{p}} \boldsymbol{\Delta}_{t+1}^{i,(k)} + \mathbf{W}^{f,(k),\mathsf{p}} \boldsymbol{\Delta}_{t+1}^{f,(k)} + \mathbf{W}^{o,(k),\mathsf{p}} \boldsymbol{\Delta}_t^{o,(k)}, && (3.88) \\
\boldsymbol{\Delta}_t^{c,(k)} =& \mathbf{i}_t^{(k)} \otimes \mathcal{H}'(\mathbf{a}_t^{c,(k)}) \otimes \mathbf{e}_t^{c,(k)}, && (3.89) \\
\boldsymbol{\Delta}_t^{f,(k)} =& \mathcal{G}'(\mathbf{a}_t^{f,(k)}) \otimes \mathbf{c}_{t-1}^{(k)} \otimes \mathbf{e}_t^{c,(k)}, && (3.90) \\
\boldsymbol{\Delta}_t^{i,(k)} =& \mathcal{G}'(\mathbf{a}_t^{i,(k)}) \otimes \mathcal{H}(\mathbf{a}_t^{c,(k)}) \otimes \mathbf{e}_t^{c,(k)}, && (3.91) \\
\boldsymbol{\Delta}_t^{(k)} =& \mathbf{W}^{c,(k-1,k)\intercal} \boldsymbol{\Delta}_t^{c,(k)} + \mathbf{W}^{i,(k-1,k)\intercal} \boldsymbol{\Delta}_t^{i,(k)} \\
&+ \mathbf{W}^{f,(k-1,k)\intercal} \boldsymbol{\Delta}_t^{f,(k)} + \mathbf{W}^{o,(k-1,k)\intercal} \boldsymbol{\Delta}_t^{o,(k)}. && (3.92)
\end{aligned}
$$

Note that (3.86) and (3.92) handle the back-propagation of the deltas between layers, in analogy to the non-recurrent DNN. From the deltas, the weight updates for the LSTM layers (recurrent and feedforward connections, as well as peephole weights) are obtained as:

$$
\Delta w_{i,j}^{\cdot,(k-1,k)} = \sum_t h_{i,t}^{(k-1)} \delta_{j,t}^{\cdot,(k)}, \tag{3.93}
$$

$$\Delta w_{i,j}^{\cdot,(k,k)} = \sum_{t<T} h_{i,t}^{(k)} \delta_{j,t+1}^{\cdot,(k)}, \tag{3.94}$$

$$\Delta w_{i,i}^{i,(k),\mathsf{p}} = \sum_{t<T} c_{i,t}^{(k)} \delta_{i,t+1}^{i,(k)}, \tag{3.95}$$

$$\Delta w_{i,i}^{f,(k),\mathsf{p}} = \sum_{t<T} c_{i,t}^{(k)} \delta_{i,t+1}^{f,(k)}, \tag{3.96}$$

$$\Delta w_{i,i}^{o,(k),\mathsf{p}} = \sum_{t} c_{i,t}^{(k)} \delta_{i,t}^{f,(k)}, \tag{3.97}$$

for $k = 1, \ldots, K-1$. Of these, (3.93) can be re-written in matrix form as:

$$\Delta \mathbf{W}^{\cdot,(k-1,k)} = \mathbf{H}^{(k-1)} \boldsymbol{\Delta}^{\cdot,(k)\mathsf{T}}, \tag{3.98}$$

with $\mathbf{H}^{(k-1)}$ representing the column-wise concatenation of the hidden layer activations from layer $k-1$ (as row vectors), for all timesteps. This formulation lends itself to efficient parallel implementation (cf. Section 3.5.5).

3.5.4 Bidirectional RNNs

The LSTM-DRNN as introduced above can exploit context from previous feature frames. In cases that real-time processing is not required, future context can be used as well. One method to take into account an unbounded amount of future context[12] is to split each hidden layer into two parts, one of which executes the forward pass in the order $t = 1, \ldots, T$ as above ('forward layer') and the other in the reverse order, i.e., replacing $t-1$ by $t+1$ for the recurrent connections and iterating over $t = T, \ldots, 1$ ('backward layer'). The forward layer and backward layer have separate weight matrices, $\overrightarrow{\mathbf{W}}$ and $\overleftarrow{\mathbf{W}}$. This yields a Bidirectional Deep Recurrent Neural Network (BDRNN) or, if the hidden units are designed as LSTM units, a Bidirectional Long Short-Term Memory (BLSTM)-DRNN.

For each time step t, the activations of the k-th forward ($\rightarrow$) and backward ($\leftarrow$) layer are collected in a single vector

$$\mathbf{h}_t^{(k)} = \left[\overrightarrow{\mathbf{h}_t^{(k)}}; \overleftarrow{\mathbf{h}_t^{(k)}}\right]. \tag{3.99}$$

Both the forward and backward layers in the next level $(k+1)$ process this entire vector as input. Thus, conceptually, in a deep BLSTM network one processes the sequence in both directions, collects the activations and uses them as input for a bidirectional pass on the sequence on the next level, etc. Alternatively to (3.99), one

[12] A system similar to an LSTM-RNN with *fixed* lookahead can be implemented by delaying the training targets by a fixed number $D \in \mathbb{N}$ of timesteps, i.e., $y_t^* \rightsquigarrow y_{t-D}^*$.

can consider 'subsampling layers' [62] performing the operation

$$\mathbf{h}_t^{(k)} = \mathcal{H}^{\text{sub},k}\left(\mathbf{W}^{\text{sub},(k)}\left[\overrightarrow{\mathbf{h}}_t^{(k)}; \overleftarrow{\mathbf{h}}_t^{(k)}\right]\right), \tag{3.100}$$

with trainable low-rank weight matrices $\mathbf{W}^{\text{sub},(k)}$, for $k = 1, \ldots, K-1$. This can be useful for information reduction between the layers. For example, in [230] using subsampling layers with tanh activation functions was found to reduce training time without decreasing performance, in contrast to simply using less hidden units.

Backpropagation for BLSTM-RNNs can be easily implemented due to the untying of the forward layer and backward layer weights. The forward layer weights $\overrightarrow{\mathbf{W}}$ are obtained in analogy to the unidirectional LSTM-RNN case, as described above, based on the 'forward activations' $\overrightarrow{\mathbf{h}}_t^{(k)}$. The backward layer weights $\overrightarrow{\mathbf{W}}$ are obtained by BPTT for LSTM-RNN, reversing the temporal dependencies ($t-1 \rightsquigarrow t+1$, $t+1 \rightsquigarrow t+1$) and using the 'backward activations' $\overleftarrow{\mathbf{h}}_t^{(k)}$ accordingly.

3.5.5 Efficient parallel implementation of RNNs

Having introduced the theoretical foundations, this section now presents an approach to efficient GPU-based training of LSTM-RNNs, as it is implemented in the author's and his colleagues' open-source CUDA RecurREnt Neural Network Toolkit (CURRENNT) [228]. The CURRENNT software can be obtained from `http://currennt.sf.net`. To the knowledge of the author, it is the first of its kind open-source parallel implementation of LSTM-RNNs in C++.

Despite that fact that recent research demonstrates that deep LSTM-RNNs exhibit superior performance in speech recognition in comparison to state-of-the-art deep feed forward networks [67, 163], RNNs are still not widely adopted by the research community at large, in contrast to the growing interest in DNNs [82]. It can be argued that one of the major barriers is the lack of high-performance implementations for training RNNs. At the same time, such implementations are non-trivial due to the limited parallelism caused by time dependencies. This is in contrast to DNNs, for which the backpropagation algorithm can be performed in parallel for all time steps, resulting in large matrix-matrix multiplications which can be efficiently done on GPUs (cf. also Section 3.2.6).

To refer to a few studies on parallelization and freely available implementations: A 'reference' CPU implementation of LSTM-RNNs as used by Graves [62] is available as open-source C++ code [65]. A Python library for many machine learning algorithms including LSTM-RNN has been introduced by Schaul et al. [165]; however, it does not directly support parallel processing. Multi-core training of (standard) RNNs has been investigated by Cernanský [20], but the source code is not available, and LSTM is not supported. Pascanu et al. have recently released a Python implementation

('GroundHog') of various RNN types described in their study [146], exploiting GPU-accelerated training through Theano [12]; yet, it does not provide LSTM-RNNs, and at the moment there is no user-friendly interface. Freely avalailable CUDA C++ implementations of feedforward, but not recurrent, neural networks are provided by Donati [32] and Lopes et al. [118].

Below, the principles of the parallel mini-batch learning algorithm implemented in CURRENNT are described. More details can be found in [11].

3.5.5.1 Parallel mini-batch learning for DRNNs

From the dependencies between layers ($k-1 \rightsquigarrow k$) and time steps ($t-1 \rightsquigarrow t$) in the DRNN forward pass outlined above, it is obvious that parallel computation of feedforward activations cannot be performed across layers; further, parallel computation of recurrent activations is not possible across time steps.

Thus, parallelization has to be performed by considering multiple *sequences* in parallel – an obvious choice for the set of parallel sequences are mini-batches (cf. Section 3.3) of size $|\mathcal{B}| < N$. All of the features, hidden layer variables, network outputs, and training targets (if needed) are stored in large matrices in order to exploit BLAS. For instance, a cell state matrix $\mathbf{C}^{(k)}$ for the k-th layer is given as

$$\mathbf{C}^{(k)} = [\mathbf{c}_{1,p}^{(k)} \cdots \mathbf{c}_{1,j+|\mathcal{B}|-1}^{(k)} \cdots \mathbf{c}_{T,p}^{(k)} \cdots \mathbf{c}_{T,j+|\mathcal{B}|-1}^{(k)}], \tag{3.101}$$

where $\mathbf{c}_{t,j}^{(k)}$ are the cell states for sequence j in layer k at time t.

In the above, it is assumed assume that every sequence has exactly T time steps. For shorter sequences, 'dummy' time steps are introduced, which are neglected in the error calculation (3.40). In practice, the largest T per batch $\mathcal{B}_i$, $T_{\mathcal{B}_i}$, can be determined, and computations are only performed up to that $T_{\mathcal{B}_i}$, which is easy due to the temporal ordering in (3.101): The timesteps $t = 1, \ldots, T_{\mathcal{B}_i}$ form a contiguous sub-matrix of (3.101).

Furthermore, instead of determining mini-batches randomly, the set of sequences is split in a way such that sequences of similar length are in the same batch. This minimizes the number of dummy time steps which have to be introduced. Consequently, sequence shuffling, which is commonly applied in training to help generalization (cf. Section 3.3), is performed by randomizing the order of mini-batches, as well as the order of sequences within mini-batches, but not by exchanging sequences across batches, which would violate the constraint of equal sequence lengths.

The realization of the LSTM-DRNN forward pass shall be exemplified again by the state variables. The update (3.81) is performed on $\mathbf{C}^{(k)}$ (3.101) by first computing the feedforward part for all time steps and $|\mathcal{B}|$ sequences in parallel, simply by pre-multiplication of the hidden layer activations, which are stored in analogy to (3.101), with $\mathbf{W}^{(k-1),(k)}$. Second, for the recurrent part of (3.81), one can update the submatrices of $\mathbf{C}^{(k)}$ for each timestep from 'left to right' using $\mathbf{W}^{(k),(k)}$. There, the

Table 3.2: Performance (WER / speedup) on the CHiME-2013-SV noisy word recognition task. In each epoch, roughly 10 h of speech are processed (training and validation set).

	RNNLIB [65]	CURRENNT			
# Parallel sequences ($\|\mathcal{B}\|$)	1	1	10	50	200
Validation set error (10 ep.)	0.138	0.138	0.135	0.137	0.144
Validation set error (50 ep.)	0.120	0.119	0.116	0.118	0.119
Training time / epoch [s]	7 420	3 805	580	392	334
Speedup	(1.0)	2.0	12.8	18.9	22.2

matrix structure (3.101) ensures memory locality of the data corresponding to one time step (matrices are stored in column-major order). For the matrix multiplications, the CUBLAS routines are used, as in the efficient NMF implementation presented in this thesis (cf. Section 3.2.6). The element-wise operations (addition, multiplication, application of the activation function) are realized by means of the Thrust framework which is part of CUDA, and provides a parallel GPU implementation of routines which are similar to C++' Standard Template Library (STL). It is straightforward to coerce the remaining hidden variables in the forward pass (hidden and gate unit activations) into matrices of the same dimension as (3.101).

Input, output and forget gate activations are calculated in parallel in analogy to the state variables. For bidirectional layers, the above matrix structure is replicated at each layer. In the 'reverse' part, the recurrent parts are updated from 'right to left' according to the procedure outlined in Section 3.5.4.

During network training, the *backward pass* for the hidden layers is realized similarly, by splitting the matrix of weight changes into a part propagated to the preceding layer and a recurrent part propagated to the previous time step, resulting in a parallel implementation of the BPTT algorithm. Details of the implementation are described by Bergmann [11].

The weight changes are summed for all sequences (batch learning) or applied after each mini-batch. In either case, only $|\mathcal{B}|$ sequences have to be kept in memory at once, allowing for learning from large data sets – a practical prerequisite for this is a data file format supporting random access, which is fulfilled by the NetCDF data format chosen for CURRENNT.

3.5.5.2 A word recognition benchmark

Let us illustrate the performance of parallel LSTM-DRNN training by a benchmark of the CURRENNT software on a word recognition task in convolutive non-stationary noise (CHiME-2013-SV task, cf. Section 2.2.1.1). There, BLSTM networks have been shown to yield best ASR performance in a multi-stream Hidden Markov Model (HMM) [52]. The frame-wise WER as well as the computation speedup

in training a BLSTM-DRNN are reported with respect to the open-source C++ reference implementation by Graves [65] running in a single CPU thread on an Intel Core2Quad PC with 4 GB of RAM. The GPU is an NVIDIA GTX 560 with 2 GB of RAM. The results are compared for different values of $|\mathcal{B}|$ while fixing the other training parameters. The corresponding NetCDF, network configuration, and training parameter files are distributed with CURRENNT. The results (Table 3.2) show that the error rate after 50 epochs is not heavily influenced by the size of the data fractions in hybrid batch/on-line learning, while speedups of up to 22.2 can be achieved.

3.6 Discriminative training of source separation

It is notable that neither of the objectives considered so far for supervised training in source separation, namely those for DNN mask prediction (3.58) and NMF dictionary learning, match the actual objective of source separation, which is to optimally recover one or more desired sources from a mixture signal. Introducing this kind of *discriminative* objective in source separation training is one major contribution of this thesis and has been proposed by the author and his colleagues [81, 215, 216].

Recall that the objective of NMF dictionary training (without regularization) can be formulated as

$$\mathbf{W} = \arg\min_{\mathbf{W}} \sum_{l} D(\mathbf{S}^l \mid \mathbf{W}^l \mathbf{H}^l). \tag{3.102}$$

In DNN training, the objective is

$$\mathbf{W} = \arg\min_{\mathbf{W}} \sum_{t} D(\mathbf{y}_t^* \mid \mathbf{y}_t), \tag{3.103}$$

where $\mathbf{y}_t \in [0,1]^F$ is the estimated time-frequency mask.

Let us now consider the case of extraction of one desired source s from a mixture, thus dropping the source index l for readability. For a time-frequency masking approach such as NMF or regression-based separation, the optimal reconstruction objective can be written as a generic cost function for the time-frequency mask $\mathbf{y}$:

$$D^{\mathsf{DT}}(\mathbf{y}) = \frac{1}{2} \sum_{f,t} \left(\hat{s}_{f,t} - s_{f,t}\right)^2 = \frac{1}{2} \sum_{f,t} \left(y_{f,t} m_{f,t} - s_{f,t}\right)^2. \tag{3.104}$$

Thus, it is not clear if the NMF and DNN training objectives (3.102) and (3.103) are related to the actual objective (3.104) of source separation.

In the DNN case, optimizing (3.104) can be easily achieved by backpropagation. Then,

$$\frac{\partial D^{\mathsf{DT}}}{\partial y_{f,t}} = \left(y_{f,t} m_{f,t} - s_{f,t}\right) m_{f,t} \tag{3.105}$$

is the gradient of the cost function with respect to the network outputs. The weight updates are then simply determined by backpropagation to the output and hidden layers, as outlined above.

For DNNs, it is also easy to derive a discriminative training algorithm for the case of Mel-domain separation, where a mask $\mathbf{y}_t^{\mathsf{Mel}}$ is computed. To this end, one can simply substitute the Mel reconstruction (3.61) into (3.104):

$$D^{\mathsf{DT,Mel}}(\mathbf{y}) = \frac{1}{2}\sum_{f,t}(\hat{s}_{f,t} - s_{f,t})^2 = \frac{1}{2}\sum_{f,t}\left(\left(\sum_i b_{f,i} y_{i,t}^{\mathsf{Mel}}\right) m_{f,t} - s_{f,t}\right)^2. \quad (3.106)$$

It is then straightforward to adapt backpropagation to the Mel-domain objective (3.106). In fact, the Mel reconstruction (3.61) can be thought of as passing the Mel-domain mask through a deterministic linear layer.

To derive a discriminative training algorithm for supervised NMF, let us start by rewriting the Wiener-like reconstruction (3.26):

$$y_{f,t} = \frac{\sum_{r=1}^{R_s} w_{f,r} h_{r,t}}{\sum_{r=1}^{R} w_{f,r} h_{r,t}}, \quad (3.107)$$

where it is assumed – without loss of generality – that the dictionary of the desired source comes first in the matrix $\mathbf{W}$, and R_s is the number of components of that source's dictionary. The crux in optimizing (3.104) with respect to the NMF 'weights' $w_{f,r}$ is that $h_{r,t}$ itself depends on $w_{f,r}$, yet there is no closed form solution for $h_{r,t}(w_{f,r})$ – instead, $h_{r,t}$ is found by an optimization algorithm (supervised NMF using multiplicative updates). Thus, directly optimizing for $w_{f,r}$ is a *bi-level* optimization problem. However, it is easy to get around the bi-level optimization problem by considering $h_{r,t}^K$ – the NMF activations after executing K iterations of the multiplicative update – instead of $h_{r,t}$ in the above. Then, $h_{r,t}^K$ is a deterministic function of $w_{f,r}$ and the mixture $m_{f,t}$. Still, the gradient of this function is rather cumbersome. A much easier solution is to dispense with the tying of the parameters $w_{f,r}$ used in (3.107) and in the multiplicative updates. Instead, one can simply define an independent 'reconstruction' matrix $\mathbf{W}^K = (w_{f,r}^K)$ to be used in (3.107):

$$y_{f,t} = \frac{\sum_{r=1}^{R^s} w_{f,r}^K h_{r,t}^K}{\sum_{r=1}^{R} w_{f,r}^K h_{r,t}^K}. \quad (3.108)$$

Inserting (3.108) into (3.104) yields the cost function to be minimized,

$$D_2^{\mathsf{DT,NMF}}(\mathbf{W}^K) = \frac{1}{2}\sum_{f,t}\left(\frac{\sum_{r=1}^{R^s} w_{f,r}^K h_{r,t}^K}{\sum_{r=1}^{R} w_{f,r}^K h_{r,t}^K} m_{f,t} - s_{f,t}\right)^2 \quad (3.109)$$

Since $h^K_{r,t}$ does not depend on $w^K_{f,r}$ but only $w_{f,r}$ and $m_{f,t}$, the gradient computation is greatly simplified. Defining $\lambda_{f,t} = \sum_r w^K_{f,r} h^K_{r,t}$, $\lambda^s_{f,t} = \sum_{r=1}^{R^s} w^K_{f,r} h^K_{r,t}$ and $\lambda^{\bar{s}}_{f,t} = \lambda_{f,t} - \lambda^s_{f,t}$, for $r' \leq R^s$,

$$\frac{\partial D_2^{\mathsf{DT,NMF}}}{\partial w^K_{f',r'}} = \sum_t \left(s_{f',t} - \frac{\lambda^s_{f',t}}{\lambda_{f',t}} m_{f',t} \right) \left(- \frac{h^K_{r',t} \lambda_{f',t} - \lambda^s_{f',t} h^K_{r',t}}{\lambda^2_{f',t}} m_{f',t} \right) \tag{3.110}$$

$$= \sum_t \frac{\lambda^s_{f',t} \lambda^{\bar{s}}_{f',t} h^K_{r',t}}{\lambda^3_{f',t}} m^2_{f',t} - s_{f',t} \frac{\lambda^{\bar{s}}_{f',t} h^K_{r',t}}{\lambda^2_{f,t}} m_{f',t}, \tag{3.111}$$

and for $r' > R^s$,

$$\frac{\partial D_2^{\mathsf{DT,NMF}}}{\partial w^K_{f',r'}} = \sum_t \left(s_{f',t} - \frac{\lambda^s_{f',t}}{\lambda_{f',t}} m_{f',t} \right) \frac{\lambda^s_{f',t} h^K_{r',t}}{\lambda^2_{f',t}} m_{f',t} \tag{3.112}$$

$$= \sum_t s_{f',t} \frac{\lambda^s_{f',t} h^K_{r',t}}{\lambda^2_{f',t}} m_{f',t} - \frac{\left(\lambda^s_{f',t}\right)^2 h^K_{r',t}}{\lambda^3_{f',t}} m^2_{f',t}. \tag{3.113}$$

Rewriting the gradient in matrix form and splitting into positive and negative parts (in analogy to the derivation of the original NMF algorithm) yields the following multiplicative update[13] for the reconstruction matrix $\mathbf{W}^K = [\mathbf{W}^{K,s} \mathbf{W}^{K,\bar{s}}]$:

$$\mathbf{W}^{K,(q+1)} = [\mathbf{W}^{K,s,(q)} \mathbf{W}^{K,\bar{s},(q)}] \otimes \left[\begin{array}{cc} \frac{\mathbf{M} \otimes \mathbf{S} \otimes \mathbf{\Lambda}^{\bar{s}}}{\mathbf{\Lambda}^2} \mathbf{H}^{K,s\mathsf{T}} & \frac{\mathbf{M}^2 \otimes (\mathbf{\Lambda}^s)^2}{\mathbf{\Lambda}^3} \mathbf{H}^{K,\bar{s}\mathsf{T}} \\ \frac{\mathbf{M}^2 \otimes \mathbf{\Lambda}^s \otimes \mathbf{\Lambda}^{\bar{s}}}{\mathbf{\Lambda}^3} \mathbf{H}^{K,s\mathsf{T}} & \frac{\mathbf{M} \otimes \mathbf{S} \otimes \mathbf{\Lambda}^s}{\mathbf{\Lambda}^2} \mathbf{H}^{K,\bar{s}\mathsf{T}} \end{array} \right]. \tag{3.114}$$

One can also derive an objective similar to (3.109), but using the KL divergence D_1 instead of D_2:

$$D_1^{\mathsf{DT,NMF}}(\mathbf{W}^K) = \sum_{f,t} s_{f,t} \log \frac{s_{f,t}}{m_{f,t} \frac{\lambda^s_{f,t}}{\lambda_{f,t}}} + m_{f,t} \frac{\lambda^s_{f,t}}{\lambda_{f,t}} - s_{f,t} \tag{3.115}$$

$$= \sum_{f,t} s_{f,t} \log \frac{s_{f,t}}{m_{f,t} \frac{\sum_{r \leq R^s} w_{f,r} h^K_{r,t}}{\sum_r w_{f,r} h^K_{r,t}}} + m_{f,t} \frac{\sum_{r \leq R^s} w_{f,r} h^K_{r,t}}{\sum_r w_{f,r} h^K_{r,t}} - s_{f,t}. \tag{3.116}$$

For this, the partial derivatives become:

$$\frac{\partial D_1^{\mathsf{DT,NMF}}}{\partial w^K_{f',r'}} = \sum_t s_{f',t} \left(\frac{h^K_{r',t}}{\lambda_{f',t}} - \frac{h^K_{r',t}}{\lambda^s_{f',t}} \right) + m_{f',t} \frac{h^K_{r',t} \lambda_{f',t} - \lambda^s_{f',t} h^K_{r',t}}{\lambda^2_{f',t}} \tag{3.117}$$

$$= \sum_t \frac{m_{f',t} \lambda^{\bar{s}}_{f',t}}{\lambda^2_{f',t}} h^K_{r',t} - \frac{s_{f',t} \lambda^{\bar{s}}_{f',t}}{\lambda^s_{f',t} \lambda_{f',t}} h^K_{r',t} \tag{3.118}$$

[13] In [216], it is formulated as separate updates for $\mathbf{W}^{K,s}$ and $\mathbf{W}^{K,\bar{s}}$, but by the notation used here it is emphasized that the entire $\mathbf{W}^K$ matrix is updated simultaneously.

for $r' \leq R^s$ and

$$\frac{\partial D_1^{\mathsf{DT,NMF}}}{\partial w^K_{f',r'}} = \sum_t \frac{s_{f',t}}{\lambda^s_{f',t}} h^K_{r',t} - \frac{m_{f',t}\lambda^s_{f',t}}{\lambda^2_{f',t}} h^K_{r',t} \tag{3.119}$$

for $r' > R^s$. This leads to the multiplicative update,

$$\mathbf{W}^{K,(q+1)} = [\mathbf{W}^{K,s,(q)}\mathbf{W}^{K,\bar{s},(q)}] \otimes \begin{bmatrix} \frac{\mathbf{S}\otimes\mathbf{\Lambda}^{\bar{s}}}{\mathbf{\Lambda}^s\otimes\mathbf{\Lambda}}\mathbf{H}^{K,s\mathsf{T}} & \frac{\mathbf{M}\otimes\mathbf{\Lambda}^s}{\mathbf{\Lambda}^2}\mathbf{H}^{K,\bar{s}\mathsf{T}} \\ \frac{\mathbf{M}\otimes\mathbf{\Lambda}^{\bar{s}}}{\mathbf{\Lambda}^2}\mathbf{H}^{K,s\mathsf{T}} & \frac{\mathbf{S}}{\mathbf{\Lambda}}\mathbf{H}^{K,\bar{s}\mathsf{T}} \end{bmatrix}. \tag{3.120}$$

This multiplicative update is executed for a maximum number of training epochs Q or until convergence. The most straightforward initialization for $\mathbf{W}^K$ is the 'analysis matrix' $\mathbf{W}$.

Relation between discriminative NMF and DNN It is notable that by fixing the number of NMF iterations and de-coupling the analysis and reconstruction matrix, a system similar to a deep neural network which outputs a time-frequency mask is obtained:

$$\mathbf{y}_t^{\mathsf{DNMF}} = \mathcal{W}\left(\mathbf{W}^K, \mathcal{U}\left(\mathbf{W}, \mathbf{m}_t, \mathcal{U}\left(\cdots\mathcal{U}\left(\mathbf{W}, \mathbf{m}_t, \mathbf{h}_t^{(0)}\right)\right)\right)\right), \tag{3.121}$$

The number of hidden layers, $K-1$ corresponds to the number of NMF iterations; the activation function $\mathcal{W}$ at the output layer corresponds to the computation of the Wiener-like filter in (3.108), and $\mathcal{U}$ is the multiplicative update rule formulated as a non-linear function with weights $\mathbf{W}$ and a deterministic (non-trainable) connection to the input 'layer' containing the mixture features $\mathbf{m}_t$. Since $\mathbf{h}_t^{(0)}$ is initialized deterministically in supervised NMF, it can be regarded as a bias input. The weights up to the output layer are tied in this approach, and 'pre-trained' according to the NMF dictionary learning (Section 3.2.4), which is similar to a generative model – thus, the training process for $\mathbf{W}$ bears some resemblance to generative pre-training [83]. In contrast, the weights $\mathbf{W}^K$ are discrminatively fine-tuned. Generalizing this concept to untied and discriminatively trained $\mathbf{W}^k$ for $k < K$ was proposed by the author and his colleagues in [81], introducing multiplicative updates for non-negative backpropagation, but the practical advantages of this concept have not been fully investigated yet. A related approach, yet using *tied* weights for all 'layers', and not being restricted to non-negative weights, is known in the area of sparse coding [68, 188].

3.7 Dynamic classification with Hidden Markov Models

Having introduced machine learning models for frame-by-frame classification and regression, let us now move on to the de-facto standard approach to *dynamic*

classification, which is based on HMMs[14]. In dynamic audio classification, the goal is to find a mapping from a sequence of acoustic features $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$ to a sequence of symbols $w = (w_1, \ldots, w_U)$. The crux is that there is no one-to-one correspondence between feature frames and symbols – in contrast, every symbol can correspond to one or more feature frames, and finding this correspondence (*alignment*) is one of the major challenges. Probably the most important application of dynamic classification in audio processing is ASR, yet other audio recognition tasks, such as chord recognition in music [102], can be cast into the same framework.

In this section, a 'top-down' approach to presentation will be taken, focusing on the structural aspects that are required for understanding the application of HMMs to ASR in multi-source environments, as presented later on in Chapters 5 and 6. As a consequence, mostly *acoustic modeling* will be discussed below. For a more in-depth discussion, the interested reader is referred to the large body of literature on HMM-based speech and audio recognition, e.g., [18, 100, 155, 156], and many technical details can be found in the documentation of the Hidden Markov Model Toolkit (HTK) [245] and the Kaldi software [152].

Typically, a Maximum-A-Posteriori (MAP) approach is used to find an optimal w given $\mathbf{X}$:

$$w = \arg\max_{w'} p(w'|\mathbf{X}). \tag{3.122}$$

Using Bayes' rule, this can be transformed to

$$w = \arg\max_{w'} \frac{p(\mathbf{X}|w')p(w')}{p(\mathbf{X})} = \arg\max_{w'} p(\mathbf{X}|w')p(w'), \tag{3.123}$$

where the second equality comes from $p(\mathbf{X})$ being a constant in the maximization. Thus, the likelihood of the symbol sequence w' is factored into an *acoustic model likelihood* $p(\mathbf{X}|w')$ and a *language model likelihood* $p(w')$.

In case of dynamic classification, the acoustic model likelihood is determined by Hidden Markov modeling. Each symbol p is represented by an HMM with parameters $\lambda^p = (\pi^p, f^p)$ representing state transition and emission probability densities. The state transition probabilities correspond to the underlying Markov model. In this thesis, *left-to-right* Markov models are used with non-zero $\pi^p_{s_1,s_2} = 0$ only for $s_2 = s_1$ and $s_2 = s_1 + 1$. The emission probability densities model the distribution of possible acoustic realizations of the symbol, and the state transition probabilities model possible time-warping. From λ^p, it is easy to (conceptually) construct an HMM for a symbol sequence w with parameters λ^w, by concatenating the symbol HMMs using a constant transition probability between HMMs [245]. Having determined the

[14]There are initial studies demonstrating the feasibility of RNN-based dynamic classification [66, 242], but at the time of this writing, it is not fully clear how these can be generalized to large-scale recognition tasks.

sequence HMM, one obtains the acoustic model likelihood,

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_T|w) = \sum_{S \in \mathcal{S}(w)} p(\mathbf{x}_1, \ldots, \mathbf{x}_T|S) = \sum_{S \in \mathcal{S}(w)} \pi^w_{s_0,s_1} \prod_{t=1}^{T} p^w(\mathbf{x}_t|s_t)\pi^w_{s_t,s_{t+1}}, \tag{3.124}$$

where $\pi^w_{s_t,s_{t+1}}$ denotes the transition probability from state s_t to s_{t+1} in the HMM for w and $S = (s_0, \ldots, s_{T+1})$ is a state sequence starting in the starting state and ending in the end state and $\mathcal{S}(w)$ is the set of possible state sequences corresponding to w. In modern speech recognizers such as Kaldi [152], state transition probabilities are assumed to be constant, which significantly simplifies the likelihood computations – in practice, without compromising accuracy [152]. Since a naïve calculation of (3.124) takes exponential time, most often dynamic programming is used for a more efficient computation.

For the specific case of medium-to-large vocabulary ASR, there are a couple of specific adjustments made to the general framework presented above. First, there is an additional level of abstraction since the language model is usually defined on words while HMMs are defined for phonemes. The mapping from word to phoneme sequences (*dictionary*) can be implemented deterministically or probabilistically (using pronunciation alternatives) – in either case, it is straightforward to construct a sequence of phoneme HMMs from a word sequence and a dictionary. Second, to alleviate the restriction of the Markov assumption, context is modeled on the HMM level as well as on the emission probability level (cf. below). To model context on the HMM level, symbols correspond to context-expanded phonemes. There, the phonemes ('monophones') p_i are mapped to the n-phones $(p_{i-Q}, \ldots, p_i, \ldots, p_{i+Q})$ where Q is the context size ($Q = 1$: triphones, $Q = 2$: quinphones). Since the number of resulting HMMs would be exponential in Q, n-phones are clustered to reflect acoustic similarities, and the parameters of phonetically similar HMMs are *tied*. Details can be found in [245].

In the experiments described in this thesis, the language model likelihood $p(w)$ is determined by simple n-gram modeling, i.e.,

$$p(w) = \prod_{i=1}^{U} p(w_i|w_1, \ldots, w_{i-1}) \approx \prod_{i=1}^{U} p(w_i|w_{i-(n-1)}, \ldots, w_{i-1}), \tag{3.125}$$

which is similar to an $n-1$-th-order Markov chain. Typical values in n-gram language modeling are $n = 2$ to $n = 5$. The probabilities $p(w_i|w_{i-(n-1)})$ are learnt from large text corpora, i.e., hundreds of millions of words. In order to avoid combinatorial explosion for larger n and enable more reliable parameter estimation, a strategy is required for pruning rare or unseen n-grams from the language model, such as 'back-off' language models [101].

In typical ASR systems, the acoustic and language models are not weighted

equally. Instead, an exponential scaling factor (language model weight[15]) LMW is introduced in (3.123):

$$w = \arg\max_{w'} p(\mathbf{X}|w')p(w')^{\mathsf{LMW}}, \tag{3.126}$$

Typical choices of LMW range from 5 to 20.

Instead of using the above MAP approach to speech decoding, one can also use Minimum Bayes Risk (MBR) decoding. In MBR decoding, one looks for the hypothesis $\tilde{w}$ fulfilling

$$\tilde{w} = \arg\min_{w} \sum_{w'} p(w'|\mathbf{X}) D_L(w, w'), \tag{3.127}$$

where $D_L(w, w')$ is the Levenshtein (edit) distance of w and w'. The intuition is that it is 'risky' to choose a hypothesis that is far from other hypotheses that are also likely given the model, and one wants to minimize that risk. It can be shown that under assumption of model correctness, the above is equivalent to minimizing the expected WER, while MAP corresponds to minimizing expected sentence error rate [243]. Thus, if it is agreed upon using WER as the ASR performance measure, MBR decoding will improve the results over standard MAP. As for MAP decoding, efficient approximations are needed since calculating the above sum requires exponential time. In [243], an efficient forward-backward algorithm operating on lattices is described, which is used in the experiments described in this thesis.

3.7.1 GMM-HMM acoustic modeling

Today's acoustic modeling approaches, as detailed below, mostly differ in the way the emission probabilities $p(\mathbf{x}_t|s_t)$ are computed within the HMM framework. For years, acoustic modeling by GMM-HMMs has been prevalent. There, the emission probabilities for each state are defined as weighted sums of multivariate Gaussian densities,

$$p(\mathbf{x}_t|s_t) = \sum_{m=1}^{M} \alpha_{s_t,m} \mathcal{N}(\mathbf{x}_t; \mu_{s_t,m}, \mathbf{\Sigma}_{s_t,m}), \tag{3.128}$$

where $\alpha_{s_t,m} \in]0,1[$, $\mu_{s_t,m} \in \mathbb{R}^F$ and $\mathbf{\Sigma}_{s_t,m} \in \mathbb{R}^{F\times F}$ are the mixture weights, means and covariance matrices for state s_t and mixture m. To reduce the model complexity, typically diagonal covariance matrices are used,

$$\mathbf{\Sigma}_{s_t,m} = \mathbf{\Sigma}_{s_t,m}^{\mathsf{diag}} = \mathrm{diag}(\sigma^2_{1,s_t,m}, \ldots, \sigma^2_{F,s_t,m}), \tag{3.129}$$

where σ_f is the standard deviation of the acoustic feature f, $f = 1, \ldots, F$ in the mixture component m and the state s_t.

[15] Obviously, LMW becomes a weight when considering the *log*-likelihood.

3.7.1.1 Context expansion

A typical state-of-the-art front-end for GMMs considers context expansion ('frame stacking') and subsequent transformation by Linear Discriminant Analysis (LDA) [125]. Frame stacking is used to incorporate context on the feature level, which, however, results in correlated features. Thus, LDA is used to simultaneously decorrelate the features from consecutive frames and increase the correlation of the transformed features with the class label, e.g., phonemes or n-phones. Note that by LDA, robustness to noise and reverberation can be addressed, assuming that these distortions occur in regular temporal patterns which can be expressed as feature dimensions not related to phonetic information, and hence can be easily removed [192]. Functionally, LDA maps the $F' = F \times P$-dimensional sliding window of acoustic features to an F-dimensional acoustic feature vector $\mathbf{x}_t^{'}$ by means of a linear transformation with a matrix $\mathbf{A}^{\mathsf{LDA}}$:

$$\mathbf{x}_t^{'} = \mathbf{A}^{\mathsf{LDA}} \left[\mathbf{x}_{t-T_L}; \cdots ; \mathbf{x}_{t+T_R}\right]. \tag{3.130}$$

The LDA matrix is obtained by considering a split of the covariance matrix $\mathbf{C}_{\mathbf{XX}}$ of the input features into two summands:

$$\mathbf{C}_{\mathbf{XX}} = \mathbf{C}_{\mathbf{XX}}^{\mathsf{intra}} + \mathbf{C}_{\mathbf{XX}}^{\mathsf{inter}}, \tag{3.131}$$

of which the first represents the *intra-class*-covariances:

$$\mathbf{C}_{\mathbf{XX}}^{\mathsf{intra}} = \sum_{i=1}^{C} p(i) \mathbf{C}_{XX}^{i}, \tag{3.132}$$

where $p(i)$ is the prior probability of class i and $\mathbf{C}_{\mathbf{XX}}^{i}$ is the covariance matrix of the training vectors belonging to class i. In the context of ASR, the assignment of classes to feature vectors is determined by forced alignment[16]. Conversely, $\mathbf{C}_{\mathbf{XX}}^{\mathsf{inter}}$ represents the *inter-class*-covariances, i.e., the dispersion of the class centers μ_i in the feature space,

$$\mathbf{C}_{\mathbf{XX}}^{\mathsf{inter}} = \sum_{i=1}^{C} p(i) \left(\mu_i - \mu\right) \left(\mu_i - \mu\right)^{\mathsf{T}}, \tag{3.133}$$

where $\mu = E\{\mathbf{x}\}$. The LDA matrix consists of the cascade of two linear transformations:

$$\mathbf{A}^{\mathsf{LDA}} = \mathbf{A}^{\mathsf{inter}} \mathbf{A}^{\mathsf{intra}}, \tag{3.134}$$

where

$$\mathbf{A}^{\mathsf{intra}} = \mathrm{diag}(\lambda_1^{\mathsf{intra}}, \ldots, \lambda_F^{\mathsf{intra}})^{-1/2} \left[\mathbf{v}_1^{\mathsf{intra}} \cdots \mathbf{v}_F^{\mathsf{intra}}\right]^{\mathsf{T}} \tag{3.135}$$

and

$$\mathbf{A}^{\mathsf{inter}} = \left[\mathbf{v}_1^{\mathsf{inter}} \cdots \mathbf{v}_F^{\mathsf{inter}}\right]^{\mathsf{T}}, \tag{3.136}$$

[16] A forced alignment corresponds to maximization of $p(\mathbf{X}|S)$ with respect to the state sequence S, under the constraint that S represents a given word sequence.

where $\mathbf{v}_f^{\mathsf{intra}}$ and $\lambda_f^{\mathsf{intra}}$, $f = 1, \ldots, F$ are the eigenvectors and corresponding eigenvalues of $\mathbf{C}_{\mathbf{XX}}^{\mathsf{intra}}$, whereas $\mathbf{v}_f^{\mathsf{inter}}$, $f = 1, \ldots, F$ are the eigenvectors of the *transformed* inter-class covariance matrix $\mathbf{A}^{\mathsf{intra}}\mathbf{C}_{\mathbf{XX}}^{\mathsf{inter}}\mathbf{A}^{\mathsf{intra}\,\mathsf{T}}$. It can be shown that after applying (3.130), the covariance matrix of the transformed features fulfills

$$\mathbf{C}_{\mathbf{X}'\mathbf{X}'} = \mathbf{I} + \mathrm{diag}\left(\lambda^{\mathsf{inter}'}{}_1, \ldots, \lambda^{\mathsf{inter}'}{}_F\right), \tag{3.137}$$

with $\lambda^{\mathsf{inter}'}{}_f$, $f = 1, \ldots, F$ being the eigenvalues of the transformed inter-class covariance matrix. Thus, the features $\mathbf{x}_t'$ are decorrelated, but with a high dispersion of the class centers along the axes of the transformed feature space.

Note that the once popular delta feature computation [48] can also be seen as a linear transformation of a sliding window of input features similar to (3.130), yet with a transformation matrix $\mathbf{A}^{\Delta}$ that is hand-crafted instead of being discriminatively trained. In the case of delta feature computations, the output features can be interpreted as discrete derivatives of the input feature contours. The matrix $\mathbf{A}^{\Delta} \in \mathbb{Q}^{F \times F'}$ can be written as

$$\mathbf{A}^{\Delta} = \begin{pmatrix} -\delta_W & \overbrace{0 \cdots 0}^{F-1} & -\delta_{W-1} & 0 \cdots 0 & \cdots & \delta_{W-1} & 0 \cdots 0 & \delta_W & 0 \cdots 0 \\ & & & & \vdots & & & & \\ 0 \cdots 0 & -\delta_W & 0 \cdots 0 & -\delta_{W-1} & 0 \cdots 0 & \cdots & \delta_{W-1} & 0 \cdots 0 & \delta_W \end{pmatrix} \tag{3.138}$$

with

$$\delta_w = \frac{w}{\sum_{u=1}^{W} u^2}, w = 0, \ldots, W \tag{3.139}$$

and W being the delta window size, i.e., effectively a sliding window of size $P = 2W+1$ is used. A typical choice for W is 2 [245]. In modern speech recognizers such as Kaldi [152], LDA features are preferred.

3.7.1.2 Semi-Tied Covariance Matrices

Traditionally, diagonal covariance GMM-HMM acoustic models have been used with acoustic features such as MFCC, delta features, and LDA features. These features are *globally* uncorrelated (cf. the note on LDA above). However, this property does not necessarily hold for the features encountered in a single HMM state. Conversely, requiring a local decorrelating transform for each state and mixture component would result in a chicken-and-egg problem in decoding. In order to model potential correlations within HMM states without having to resort to full-covariance matrices, Semi-Tied Covariance (STC) matrices have been proposed by Gales [50]. There, the covariance matrices for state s and mixture component m are modeled as

$$\mathbf{\Sigma}_{s,m}^{\mathsf{STC}} = \mathbf{A}_{r(s,m)}\mathbf{\Sigma}_{s,m}^{\mathsf{diag}}\mathbf{A}_{r(s,m)}^{\mathsf{T}}, \tag{3.140}$$

where $r(s,m)$ defines the tying of parameters, for example, across all states corresponding to a certain monophone. The matrices $\mathbf{A}_{r(s,m)}$ are estimated by maximizing the likelihood of the training data $\mathbf{x}_t$, $t \in \mathcal{T}$ given the GMM-HMM model parameters with covariances $\mathbf{\Sigma}_{s,m}^{\mathsf{STC}}$. The estimation algorithm in Expectation-Maximization (EM) form is described in [50].

3.7.1.3 Discriminative training

After conventional Maximum Likelihood (ML) estimation of the acoustic model parameters, often discriminative training is employed, such as by the boosted Maximum Mutual Information (bMMI) criterion [154]. The bMMI training objective $\mathcal{O}_b$ for the acoustic model parameters $\mathcal{Q}$ is to maximize the mutual information between the correct transcription and the decoding lattice, given by

$$\mathcal{O}_b(\mathcal{Q}) = \log \sum_u \frac{p(\mathbf{X}^u|\mathcal{Q}, h^{w_u^*}) p(w_u^*)^{\mathsf{LMW}}}{\sum_{w_u} p(\mathbf{X}^u|\mathcal{Q}, h^{w_u}) p(w_u)^{\mathsf{LMW}} e^{-b\varrho(w_u, w_u^*)}} \tag{3.141}$$

where the outer sum is taken over all training utterances u, $\mathbf{X}^u$ denotes the acoustic features of utterance u, w_u^* is the reference transcription, w_u is a word hypothesis, and h^w denotes the HMM state sequence corresponding to the hypothesis w. The set $\{w_u\}$ in the denominator corresponds to the lattice of word hypotheses, while $p(\cdot)$ denotes the language model likelihood. $\varrho(w_u, w_u^*)$ denotes the phoneme accuracy of w_u with respect to the reference w_u^*. Thus, the term $e^{-b\varrho(w_u, w_u^*)}$ with a 'boosting factor' $b > 0$ emphasizes the weight of wrong hypotheses in the denominator calculation.

3.7.1.4 Adaptation

Model adaptation is a powerful method to achieve environmental robustness and to adapt to speaker variation. Concerning practical applications, mainly unsupervised adaptation is of interest. There, an initial transcription is obtained from the acoustic features, which is then used to either adapt the model parameters to fit the data (acoustic features and assumed transcription) or shifting the features to fit the model (sequence) corresponding to the assumed transcription. In this thesis, MAP adaptation and feature-space Maximum Likelihood Linear Regression (fMLLR) are considered.

In MAP adaptation, typically only the means of the GMM are adapted. The adapted mean $\hat{\mu}_m$ of mixture component[17] m can be written as a weighted sum of the mean and the 'adaptation mean' μ_m^{a} for m,

$$\hat{\mu}_m = \frac{\tau}{N_m + \tau} \mu_m + (1 - \frac{\tau}{N_m + \tau}) \mu_m^{\mathsf{a}}, \tag{3.142}$$

[17] In practice, to reduce the number of parameters to be estimated, parameter tying is applied by clustering mixture components according to acoustic similarity (Euclidean distance of the means) [245]. To simplify notation, it will be assumed that each cluster contains exactly one mixture component.

where $N_m = \sum_{t=1}^{T^{\mathsf{a}}} L_m(t)$ T^{a} denotes the number of feature frames in the adaptation data, and τ is the weight of the original model parameters with respect to the adaptation data. $L_m(t) = \Pr(m, t \mid \mathcal{Q}, \mathbf{X}^{\mathsf{a}})$ is the occupation likelihood of component m at time frame t, which is determined by the forward-backward algorithm [156] on the adaptation data $\mathbf{X}^{\mathsf{a}}$, using the original model parameters (Gaussian PDFs) $\mathcal{Q}$. It is easy to see that for low occupancy N_m, the adapted mean will be close to the original mean. Using $L_m(t)$, the adaptation mean is defined as

$$\mu_m^{\mathsf{a}} = \frac{\sum_t L_m(t)\mathbf{x}_t^{\mathsf{a}}}{\sum_t L_m(t)}. \tag{3.143}$$

In MLLR, both means and covariances of the GMMs are linearly transformed. For MLLR, the generic objective function $\mathcal{O}$ to be maximized reads:

$$\mathcal{O}(\hat{\mathcal{Q}}) = -\sum_{m=1}^{M}\sum_{t=1}^{T^{\mathsf{a}}} L_m(t)\left[\log\left|\hat{\mathbf{\Sigma}}_m\right| + (\mathbf{x}_t - \hat{\mu}_m)^{\intercal}\hat{\mathbf{\Sigma}}_m^{-1}(\mathbf{x}_t - \hat{\mu}_m)\right] \tag{3.144}$$

with adapted model parameters (Gaussian PDFs) $\hat{\mathcal{Q}}$. For *constrained maximum likelihood linear regression* (CMLLR), which will be the adaptation method of choice for several applications in this thesis, the adapted means and variances are obtained by linear transformation with the same matrix $\mathbf{A}_m^{-1}$,

$$\hat{\mu}_m = \mathbf{A}_m^{-1}\mu_m + \mathbf{b}_m, \qquad \hat{\mathbf{\Sigma}}_m = \mathbf{A}_m^{-1}\mathbf{\Sigma}_m\mathbf{A}_m^{-1\intercal}, \tag{3.145}$$

where $\mathbf{b}_m$ is a bias vector. Substituting this into (3.144) yields the objective function to maximize. It can be shown that the above is equivalent to linearly transforming the acoustic features by

$$\hat{\mathbf{x}}_t = \left[-\mathbf{A}_m\mathbf{b}_m \quad \mathbf{A}_m\right][\mathbf{x}_t; 1]\,. \tag{3.146}$$

For this reason, CMLLR is often referred to as *feature-space MLLR* (fMLLR).

For many practical ASR tasks, it is desirable to adapt with little data – if possible, single utterances. A popular method to cope with little adaptation data is basis fMLLR, which performs well even on very small amounts of adaptation data according to Povey and Yao [151]. In this approach, the acoustic features $\mathbf{X}$ of a speech utterance are transformed by a matrix $\mathbf{A}$

$$\mathbf{X} \mapsto \mathbf{A}[\mathbf{X}; 1] \tag{3.147}$$

which itself is a linear combination of basis matrices:

$$\mathbf{A} = \mathbf{A}_0 + \sum_{i=1}^{B}\beta_i\mathbf{B}_i. \tag{3.148}$$

The matrices $\mathbf{B}_i$ are estimated by Principal Component Analysis (PCA) on statistics derived from the fMLLR matrices obtained on the training utterances. At test time, the number J of basis matrices is varied depending on the utterance length. To this end, the top J eigenvectors $\mathbf{b}_i$ (representing row-stacked matrices $\mathbf{B}_i$) are used as basis. It is shown in [151] that for any $J \leq F(F+1)$ this is equivalent to Maximum Likelihood estimation under reasonable assumptions. J is chosen proportial to the amount of frames in the utterance to be decoded, i.e., $J = \min\{\gamma T, F(F+1)\}, 0 < \gamma \ll 1$ where F is the acoustic feature dimension. The corresponding Maximum Likelihood coefficients β_i are estimated by Newton's method. Thus, if γT is small compared to $F(F+1)$, the amount of adaptation parameters to be estimated is greatly reduced compared to conventional fMLLR (which requires an $F \times (F+1)$ matrix to be estimated). This enables robust adaptation on single utterances as short as three seconds [151].

3.7.2 Connectionist methods

Connectionist methods for acoustic modeling integrate DNNs into the HMM or GMM-HMM ASR framework. They are increasingly becoming industry standard [27, 29, 163] for the simple fact that they allow for exploiting the non-linear modeling power of DNNs while largely preserving existing speech decoders.

3.7.2.1 Tandem GMM-HMM

In tandem ASR systems, the activations of neural networks trained on phoneme or phoneme state targets are used as features, alternatively to (or in combination with) standard MFCC features. A forced alignment (cf. above) yields the most likely HMM state sequence S_u^* for each training utterance u. From this, a sequence of training targets is generated which corresponds to HMM states, $(s_1^*, \ldots, s_T^*)$, or other labels derived from the states, such as phonemes, $(P(s_1^*), \ldots, P(s_T^*))$, where P is the mapping from HMM states to (n-)phones. These training targets are used in cross-entropy training of a DNN (cf. (3.43)). Once the DNN is trained, the label log likelihoods, $\log \mathbf{y}(\mathbf{x}_t)$ are computed per frame. These are then concatenated with the acoustic features to form a tandem feature vector, and PCA is used for decorrelation, yielding the acoustic features

$$\mathbf{u}_t = \mathbf{A}^{\mathsf{PCA}} \left[\mathbf{x}_t; \log \mathbf{y}(\mathbf{x}_t)\right]. \tag{3.149}$$

In the *bottleneck* approach [69, 70], the activations from a (usually narrow) hidden layer are used instead of the output activations,

$$\mathbf{b}_t = \mathbf{A}^{\mathsf{PCA}}[\mathbf{h}_t^{K-B}; \mathbf{x}_t] \tag{3.150}$$

where

$$\mathbf{h}_t^{K-B} = \mathcal{H}^{K-1}\left(\mathbf{W}^{K-B} \ldots \mathcal{H}^1\left(\mathbf{W}^1[\mathbf{x}_t; 1]\right)\right) \tag{3.151}$$

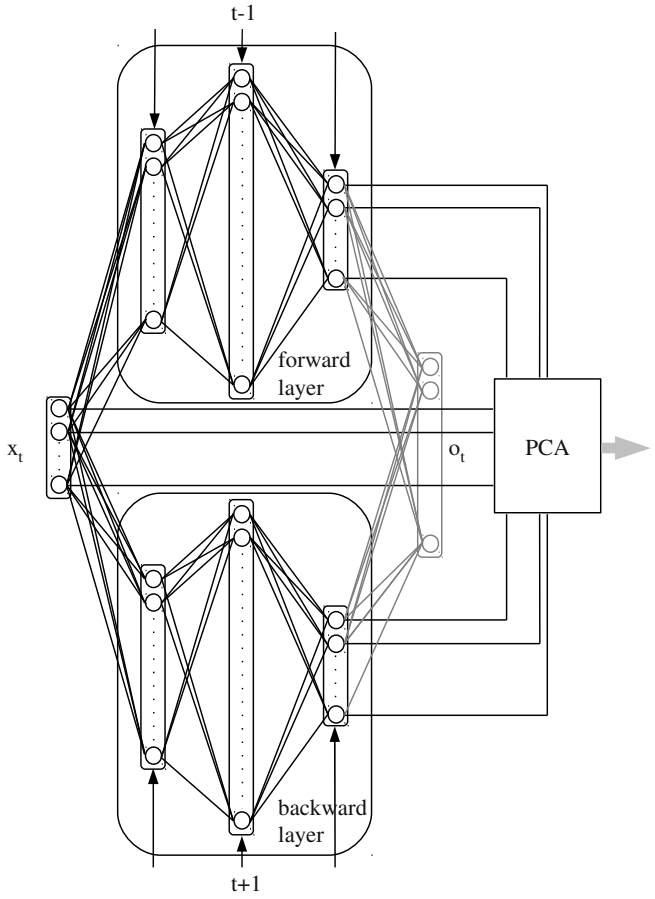


Figure 3.7: Architecture of the Bottleneck-BLSTM front-end [223]. $\mathbf{o}_t$: Output layer activations.

is the hidden layer activation of the bottleneck layer ($B-1$-th last layer) and $\mathbf{A}^{\mathsf{PCA}}$ is determined by PCA on a training set. The HMM emission probability is then simply $p(\mathbf{b}_t|s_t)$ respectively $p(\mathbf{u}_t|s_t)$ in analogy to (3.128).

For tandem feature generation, various DNN architectures can be used. The author's colleague introduced bottleneck feature generation by bidirectional LSTM-DRNN, which was shown to lead to lower error rates than standard DNN features in spontaneous speech recognition [235]. In particular, using LSTM-DRNN captures co-articulation effects in human speech (for more details, see [63]). In the context of this thesis, the noise robustness of the bottleneck LSTM-DRNN approach will be assessed in Section 5.1.2.

Figure 3.7 exemplifies the bottleneck feature extraction procedure using a BLSTM-DRNN. In bidirectional processing, there are two bottleneck layers: one within the network processing the speech sequence in forward direction and one within the network for backward processing. In Figure 3.7, the connections between the

bottleneck layers and the output layer are depicted in grey, indicating that the activations of the output layer ($\mathbf{o}_t = \mathbf{y}(\mathbf{x}_t)$) are only used during network training and not during Bottleneck-BLSTM feature extraction.

3.7.2.2 Hybrid DNN-HMM

In the *hybrid* approach [82], a DNN is trained in the same way as for the Tandem approach. However, instead of using the output activations as features in a GMM, they are used to calculate the HMM emission probabilities directly. Note that the output activations $\mathbf{y}(\mathbf{x}_t)$ of a DNN with a softmax output layer, trained on HMM state targets s_t^*, can be interpreted probabilistically as

$$p(s_t|\mathbf{x}_t) = \frac{\exp\left(\sum_i w_{s_t,i}^K h_{i,t}^{K-1}\right)}{\sum_{s'} \exp\left(\sum_i w_{s',i}^K h_{i,t}^{K-1}\right)} \tag{3.152}$$

where $\mathbf{W}^K = (w_{s,i}^K)$ is the output layer matrix of the DNN and $\mathbf{h}_t^{K-1}$ is the $K-1$-th hidden layer activation of the DNN in analogy to (3.150) . To convert this to HMM emission probabilities, Bayes' rule is used:

$$p(\mathbf{x}_t|s_t) = \frac{p(s_t|\mathbf{x}_t)p(\mathbf{x}_t)}{p(s_t)} \tag{3.153}$$

Since $\mathbf{x}_t$ is fixed in the maximization (3.123), $p(\mathbf{x}_t)$ can be assumed as an arbitrary constant, whereas $p(s_t)$ is determined from a forced alignment of the training data.

3.7.2.3 Multi-stream HMM

In the multi-stream HMM approach, a D(R)NN with a softmax output layer is trained on a frame-wise phoneme classification task. Thus, the output activations $\mathbf{y}_t$ of the DNN correspond to pseudo posteriors $\tilde{\mathbf{y}}_t \in [0,1]^P$ where P is the number of phonemes. From this, a frame-wise phoneme prediction $b_t \in \{1,\ldots,P\}$ is derived as

$$b_t = \arg\max_i \tilde{y}_{t,i}. \tag{3.154}$$

The phoneme prediction is decoded along with the acoustic feature vector (e.g., MFCCs) in a multi-stream HMM. The joint probability of observing an acoustic feature vector $\mathbf{x}_t$ and a DNN phoneme prediction b_t in the HMM state s_t is obtained as

$$p(\mathbf{x}_t, b_t|s_t) = p(\mathbf{x}_t|s_t)^\nu p(b_t|s_t)^{2-\nu} \tag{3.155}$$

where $\nu \in]0,2[$ is a stream weight. The emission probabilities $p(\mathbf{x}_t|s_t)$ are modeled by conventional GMMs whereas the probabilities $p(b_t|s_t)$ are determined from the row-normalized state-phoneme confusion matrix on a held out part of the training data. The multi-stream approach allows for probabilistic modeling of possible phoneme confusions, i.e., uncertainties in the DNN predictions [238].

Part III

Applications

4

Speech enhancement

The articulate voice is more distracting than mere noise. – Seneca

In this chapter, the application of machine learning techniques to the problem of speech enhancement, i.e., extracting a speech source from a mixture of speech and generally non-stationary noise, will be discussed. These techniques rely on spectral properties of the speech and the noise that are learnt from training data, as opposed to 'blind' de-noising schemes such as minimum statistics spectral subtraction [124], which rely on global assumptions such as stationarity of the noise, which are not always met in practice.

In the first section, semi-supervised online learning of noise models will be discussed, which can be used to cope with highly variable 'noise' such as background music, as was shown in a study by the author and his colleagues [220]. Then, supervised discriminative training of NMF and DNN speech enhancement models will be evaluated in a large-scale benchmark, in order to demonstrate the efficacy of formulating speech enhancement as a regression task, as was done by the author and his colleagues in [215].

4.1 Speech / music separation by NMF

Separation of speech overlaid with music in monaural signals remains a challenging problem, especially due to the large similarity between voiced (harmonic instruments, vowels) and unvoiced (drums, consonants) parts. In turn, robust suppression of background music can be immediately exploited in a variety of applications, comprising speech enhancement for in-car human-machine interfaces or mobile telephony in highly noisy environments such as discotheques, speech recognition for multimedia information retrieval in TV series or on-line videos, or even lyrics transcription of rap/hip-hop music.

First promising results for monaural background music suppression have been obtained by Ozerov et al. [142], indicating that NMF is a promising approach to

ensure intelligibility of the extracted speech signal. A more comprehensive evaluation has been carried out by Raj et al. [157], using an exemplar-based approach based on supervised NMF (cf. Section 3.2.4), i.e., predefining a large set of speech and music spectra extracted from training data. Still, from these results it is not clear whether information about the music signal as in fully supervised NMF is required for optimal separation.

Hence, in this section, semi-supervised NMF (cf. Section 3.2.5) is considered, where only speech dictionaries are pre-defined, whereas the music dictionaries are not characterized a priori, in order to cope with variability of the music over time, as in [142]. The semi-supervised method is compared against an 'upper bound' for the performance of (supervised) NMF where music dictionaries are estimated from parts of the ground truth music, and the influence of sparsity constraints is evaluated. Unlike in [142], sparsity constraints are enforced on the NMF activations – similar to the algorithm proposed by Virtanen [205] – to improve discrimination of speech and music spectra, and this algorithm is extended to dictionaries with sparse spectra in order to model harmonicity of music. Furthermore, a rather small set of speech dictionary atoms is used for initialization that are learnt from training data by NMF. This is done to vastly decrease computational effort compared to exemplar-based approaches such as [157]. A speaker-dependent scenario with 168 speakers from the TIMIT database is chosen for evaluation, and different music styles are investigated.

4.1.1 Experiments

The experiments rely on semi-supervised and supervised NMF as introduced in Sections 3.2.3 and 3.2.5, where the sources correspond to speech ($l = 1$) and music ($l = 2$). For clarity, the superscripts (s) and (m) will be used instead of 1 and 2. The cost function of sparse semi-supervised NMF (3.34) is slightly modified as follows:

$$\begin{aligned} D(\mathbf{W}^{(m)}, \mathbf{H}) &= D_r(\mathbf{W}^{(m)}, \mathbf{H}) \\ &+ \lambda\, D_s^{\mathbf{H}}(\mathbf{H}^{(m)}) + \mu D_s^{\mathbf{W}}(\mathbf{W}^{(m)}) \end{aligned} \tag{4.1}$$

where D_r corresponds to the reconstruction error as $D_1(\mathbf{M} \mid \mathbf{WH})$ (generalized Kullback-Leibler divergence) and

$$D_s^{\mathbf{H}}(\mathbf{H}^{(m)}) = \sum_{r=1}^{R^{(m)}} \frac{1}{\sigma(\mathbf{H}^{(m)}_{r,:})} \sum_{t=1}^{T} \mathbf{H}^{(m)}_{r,t}, \tag{4.2}$$

$$D_s^{\mathbf{W}}(\mathbf{W}^{(m)}) = \sum_{r=1}^{R^{(m)}} \frac{1}{\sigma(\mathbf{W}^{(m)}_{:,r})} \sum_{f=1}^{F} \mathbf{W}^{(m)}_{f,r} \tag{4.3}$$

are sparsity constraints with weights λ and $\mu \geq 0$. $\sigma(\mathbf{W}^{(m)}_{:,r})$ and $\sigma(\mathbf{H}^{(m)}_{r,:})$ are standard deviation estimates for the r-th column of $\mathbf{W}^{(m)}$ and the r-th row of $\mathbf{H}^{(m)}$, that are

introduced to avoid dependency on the scaling of the matrices, following [205]. This kind of sparsity constraint provides similar performance to L1-norm constraints [97].

Informally, D_s is a sparsity constraint that is only enforced on the music part: The purpose of imposing sparsity on $\mathbf{H}^{(m)}$ is to mitigate the fact that the algorithm can 'mis-use' the bases designated to isolate the music for modeling the speech parts; additionally, sparsity on $\mathbf{W}^{(m)}$ is imposed to increase the discrimination between speech and music, as the latter is arguably characterized by higher harmonicity compared to speech. For low-dimensional speech dictionaries as used in the experiments below, there is no apparent reason to impose sparsity on the speech activations during learning or separation.

The cost function (4.1) is minized by applying component-wise multiplicative updates to $\mathbf{W}^{(m)}$, $\mathbf{H}^{(s)}$ and $\mathbf{H}^{(m)}$ based on the algorithm proposed in [205]. It is straightforward to extend the algorithm to the semi-supervised case, including the sparsity constraint for the spectra $\mathbf{W}^{(m)}$ which was not considered in [205], yielding the following update rule for $\mathbf{W}^{(m)}$:

$$\mathbf{W}^{(m)} \leftarrow \mathbf{W}^{(m)} \otimes \frac{\nabla D^{-}(\mathbf{W}^{(m)}, \mathbf{H})}{\nabla D^{+}(\mathbf{W}^{(m)}, \mathbf{H})}, \tag{4.4}$$

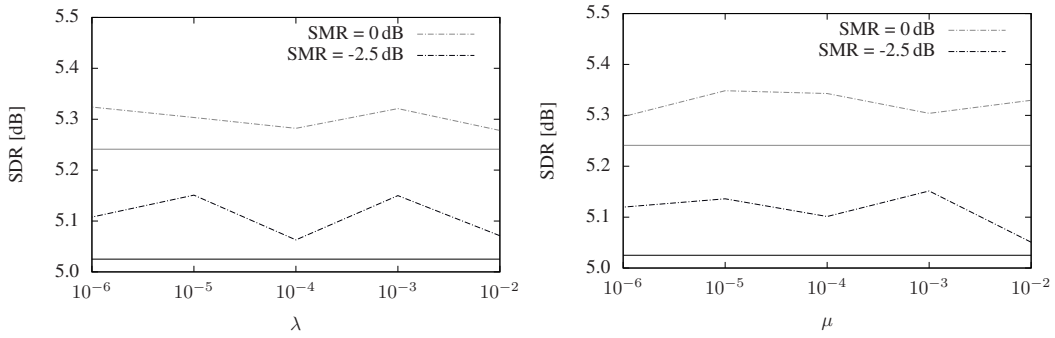
where ∇^+ and ∇^- indicate the positive and negative parts of the gradient, which, in turn, are composed of $\nabla D_r^+(\mathbf{W}^{(m)})$ and $\nabla D_r^-(\mathbf{W}^{(m)})$ (cf. (3.23)) and

$$[\nabla D_s^{\mathbf{W}^+}(\mathbf{W}^{(m)})]_{f,r} = \frac{\sqrt{F}}{\sqrt{\sum_{f=1}^{F} w_{f,r}^{(m)2}}}, \tag{4.5}$$

$$[\nabla D_s^{\mathbf{W}^-}(\mathbf{W}^{(m)})]_{f,r} = w_{f,r}^{(m)} \frac{\sqrt{F} \sum_{f=1}^{F} w_{f,r}^{(m)}}{\left(\sum_{f=1}^{F} w_{f,r}^{(m)2}\right)^{3/2}}. \tag{4.6}$$

The update rules are applied for $K = 100$ iterations starting from a (Gaussian) random solution. Finally, the estimated clean speech spectrogram $\widehat{\mathbf{S}}^{(s)}$ is obtained by Wiener-like filtering (3.26). Since the asymptotic complexity of the NMF multiplicative update is polynomial ($O(RFT)$), and linear in each of R, F and T, it is desirable to keep the number of components R as low as possible at a reasonable separation quality for performance critical applications. All experiments for this paper are based on the NMF implementations found in the open-source toolkit openBliSSART co-authored by the author of this thesis [212, 217].

The aim of the following experiments is to evaluate the extraction of the speech from mixed speech and music audio signals, as well as to determine the influence of sparsity weights, Discrete Fourier Transform (DFT) window size and music style.



(a) Sparsity on $\mathbf{H}^{(m)}$ (b) Sparsity on $\mathbf{W}^{(m)}$

Figure 4.1: Average signal-to-distortion ratio (SDR) on TIMIT test set (1 680 sentences overlaid with waltzes from BRD database) by sparse semi-supervised NMF (dashed-dotted lines) for SMRs of -2.5 and 0 dB. Variation of the sparsity weights λ for $\mu = 0$ (a), and of μ for $\lambda = 0$ (b). Continuous lines: semi-supervised NMF with $\lambda = \mu = 0$, i. e., no sparsity constraints. DFT window size 128 ms.

4.1.1.1 Evaluation Data Set

The evaluation set is formed by 1 680 audio signals (sentences) spoken by 168 different subjects (56 females and 112 males) from the TIMIT database test set, i. e., there are 10 sentences of typically 2–3 seconds length for each speaker. The TIMIT database [47] is chosen for its rich phonetic content. Each of the TIMIT test sentences is artificially mixed with a random segment of music of the same length at various speech-to-music ratios (SMRs) from -7.5 dB to +5 dB in intervals of 2.5 dB. These SMRs correspond to typical application scenarios as indicated above. To demonstrate the performance on a variety of music styles, the experiment was first carried out with the 136 Viennese Waltzes from the Ballroom Dance (BRD) Database [169] as an example for classical music, then it was repeated with 136 pieces of each of the latin and rock genres. Note that these frequently contain segments with sung vocals, which makes separation of speech particularly challenging.

4.1.1.2 Protocol

Evaluation is carried out in a speaker-dependent cross-validation, i. e., for each TIMIT test sentence all other sentences of the same speaker are concatenated, and their spectrogram is reduced to a NMF speech dictionary $\mathbf{W}^{(s)}$. As this results in aproximately 20–30 seconds training material for separation of each test instance, this methodology resembles a practicable way of speaker adaptation. For supervised NMF, music dictionaries $\mathbf{W}^{(m)}$ are computed from a disjoint random section of 25 seconds of the same music track used for mixing the test file – this yields an upper benchmark on the performance of supervised NMF assuming the exact characteristics of the

music are known during separation. The number of music and speech components in NMF, $R^{(m)}$ and $R^{(s)}$, were set to 10 and 20. 20 speech components have been found to represent a good compromise between separation quality and computational complexity [185], and the ratio 2/1 for the speech and music dictionaries was chosen empirically in preliminary experiments. The experiment was repeated for different Hann window sizes from 8 ms to 512 ms whereas the overlap between consecutive DFT frames in the time domain remained fixed at 75 %.

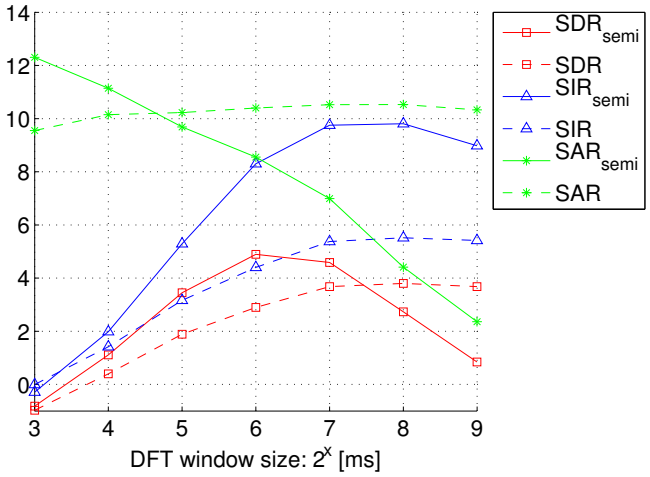
To assess the characteristics of semi-supervised and supervised speech and music separation in detail, SDR (2.10), SIR (2.11) and SAR (2.12) of the extracted speech source are measured. Measurements were carried out using the open-source BSS_Eval toolkit [200].

4.1.2 Results

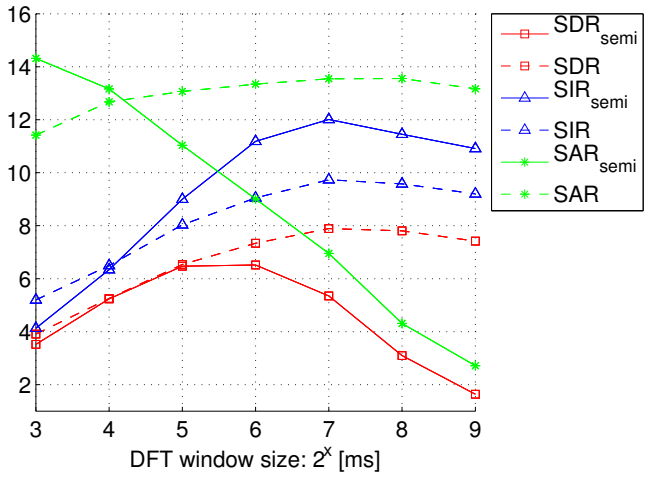
Figure 4.1 shows the performance in terms of SDR for semi-supervised NMF when varying the sparsity weight λ for the music activations $\mathbf{H}^{(m)}$ while keeping the sparsity weight μ for the music spectra $\mathbf{W}^{(m)}$ constant at zero (4.1a), and vice versa (4.1b). In both evaluation scenarios, sparsity constraints slightly – yet consistently – increase performance by over 0.1 dB absolute at all SMRs. Across SMRs, best results are obtained at $\mu = 10^{-5}$. Evidently, this choice of sparsity weight is highly 'non-aggressive' – larger values of μ would probably force a reduction of the $\mathbf{W}^{(m)}$ to single harmonics which is not desirable in the case of complex music. Additional experiments showed that using both $\lambda, \mu > 0$ could not further improve the results.

Next, Figures 4.2a and 4.2b show the results in terms of SDR (indicated by squares), SIR (triangles) and SAR (asterisks) for both supervised and (non-sparse) semi-supervised NMF at SMRs of -5 dB and 0 dB, for varying window sizes (8, 16, 32, 64, 128, 256 and 512 ms), corresponding to DFT sizes of 128–8192 points. For both supervised and semi-supervised NMF, the best suppression (SIR) is achieved at a window size of 128 ms while small window sizes (< 32 ms) do not enable robust suppression in general. For a SMR of -5 dB, the semi-supervised method improves SIR by more than 4 dB compared to the supervised case, boosting the average SIR to almost 10 dB. At 0 dB SMR the improvement by semi-supervised NMF is smaller but still clearly visible, achieving over 12 dB average SIR. In terms of overall quality (SDR), at -5 dB semi-supervised NMF delivers equal or higher SDR for up to 128 ms window size; at 0 dB, larger window sizes (> 64 ms) decrease the SAR – and hence the SDR – of the separated speech for semi-supervised NMF. It can be concluded that semi-supervised NMF seems to be prone to lose some speech information at higher SMRs. Since this effect does not occur for supervised NMF, it can be argued that larger window sizes help to create a more precise model of the true music in supervised NMF.

The rather slight effect of imposing sparsity constraints on the music dictionaries deserves some further investigation. Figures 4.3a through 4.3c show how the music



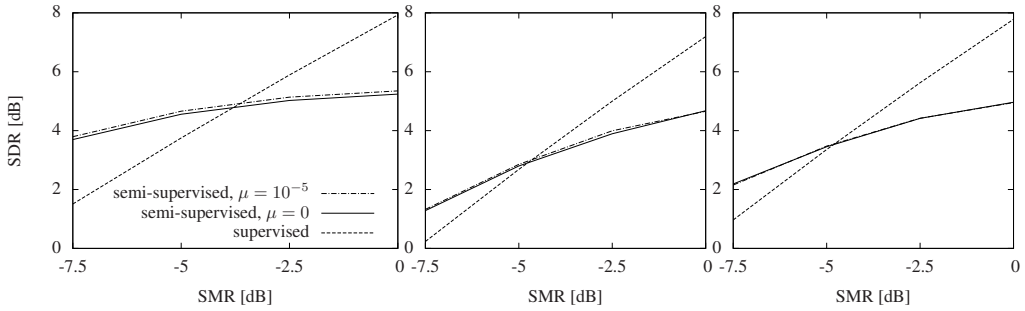
(a) SMR = -5 dB



(b) SMR = 0 dB

Figure 4.2: Average separation performance on TIMIT test set (1 680 sentences) overlaid with waltzes from the BRD database. Effect of DFT window size on non-sparse semi-supervised NMF separation (dashed lines) compared to supervised NMF (continuous lines).

genre affects the separation for three different styles: classical (waltz), latin and rock. The experiments are done using a DFT window size equal to 128 ms with sparse ($\mu = 10^{-5}, \lambda = 0$) and non-sparse ($\mu = \lambda = 0$) semi-supervised as well as supervised NMF. The results reveal that the improvement by using sparse instead of non-sparse semi-supervised NMF – i. e., enforcing harmonicity in the music spectra – is mostly visible for waltz music with its arguably high degree of harmonicity, while for rock music no gains can be observed.



(a) waltz (b) latin (c) rock

Figure 4.3: Separation of TIMIT test set (1 680 sentences): Average SDR results for speech overlaid by different music styles, for sparse semi-supervised ($\mu = 10^{-5}, \lambda = 0$), semi-supervised ($\mu = \lambda = 0$) and supervised NMF. DFT window size 128 ms.

Corroborating the results obtained for speech recognition accuracy in [157] on the acoustic level, all methods exhibit highest performance for waltz music while results consistently downgrade for the other styles: On average a loss of almost 2 dB SDR is observed for rock compared to waltz music, which can be attributed to the spectral overlap with 'noisy' instruments such as drums and distorted guitars, since rock music also decreases the SDR obtained by supervised NMF.

4.1.3 Conclusions

In this section, a large-scale study on performance of semi-supervised and supervised NMF algorithms for compensation of background music in speech has demonstrated the effectiveness of semi-supervised NMF particularly in highly noisy environments. Comparing the semi-supervised method to an upper benchmark for supervised NMF assuming the characteristics of the music are known, it is notable that in highly 'noisy' conditions, the semi-supervised method suppresses the music to a larger extent than the supervised benchmark, in terms of SIR; however, at higher speech-to-music ratios a decrease in overall quality (SDR) has to be accepted. This can be attributed to the relative simple modeling of speech by predefined spectral vectors, which can cause information loss in the reconstructed speech signal since subtleties of speech not modeled by the predefined dictionary are captured by the iteratively updated vectors designated to contain the background music. By enforcing sparsity constraints on the spectra and on the activations, the performance of semi-supervised NMF could be improved slightly, but the gain strongly depends on the genre and according complexity of the music signal. Overall, the performance of semi-supervised NMF seems to depend strongly on 'parameter tuning', such as the DFT window size. This finding has been corroborated by the author in subsequent studies (cf. [223, 224], Section 5.1.2).

Future work on speech/music separation could focus on integration of automatic genre recognition on the separated music signal in a two-stage separation algorithm: This enables the usage of optimal NMF parameterizations for different genres. Furthermore, the relatively simple speech models and factorization constraints in this study should be extended by explicit models of temporal dependencies, such as in [132, 134].

4.2 Synopsis: Supervised training for speech enhancement

Having compared supervised and semi-supervised NMF, in this section, a comparative evaluation of supervised training methods and discriminative training for real-time speech enhancement will be provided, as was presented by the author and his colleagues in [215]. The methods investigated will comprise machine learning based methods allowing real-time separation. These include supervised NMF[1] and deep (non-recurrent as well as recurrent) neural networks. Furthermore, the influence of discriminative training as introduced in Section 3.6 on these methods will be evaluated in a fair comparison. As such, this section provides a synopsis on the performance of the machine learning methods introduced in this thesis in the speech enhancement task. Non-stationary noise will be considered which not only comprises music as in the previous section, but also interfering speakers and domestic noise, as featured in the CHiME-2013 corpus (cf. Section 2.2.1.1).

4.2.1 Experiments

As input features for speech enhancement, spectral features are extracted according to Section 3.1, with various magnitude warping exponents α, using the square root of the Hann window, a window size of $W = 400$ points (25 ms) and a frame shift of 160 points ($\Delta\tau = 10$ ms). Optionally, Mel spectra are calculated.

Supervised NMF is implemented according to Section 3.2.2, i.e., using sparse regularization. As shown in [216], sparse NMF outperforms exemplar-based NMF by a large margin on the CHiME-2013 corpus. A larger dictionary than in the previous section is required since by the CHiME-2013 setup, the evaluation has to be speaker-independent. The SNMF hyper-parameters are set as $\alpha = 1$ (using magnitude spectra), $T_L = 8$ (frame stacking with 8 frames of left context), $T_R = 0$ (no right context, in order to allow for real-time operation), $K = 25$ (number of iterations), $R^{(s)} = 1\,000$, $R = 2\,000$ (number of speech dictionary atoms and total

[1] Note that semi-supervised NMF does also allow for real-time separation, yet with additional overhead [96], and is more susceptible to parameter tuning than supervised NMF (cf. the previous section as well as [96, 224]) – hence, it will not be considered in this section.

Table 4.1: Average SDR for various topologies (# of hidden layers × # of hidden units per layer) of DNN and LSTM-DRNN on the CHiME-2013 development set.

SDR [dB]	Input SNR [dB]						
	-6	-3	0	3	6	9	Avg.
Noisy	-3.73	-1.05	1.18	2.86	4.53	6.19	1.66
DNN 1×1024	4.48	6.90	8.96	10.38	12.11	13.95	9.46
DNN 2×1024	4.76	7.17	9.15	10.62	12.38	14.27	9.72
DNN 3×1024	5.77	8.00	9.92	11.24	12.99	14.84	**10.46**
DNN 4×1024	5.70	7.92	9.91	11.26	13.02	14.83	10.44
DNN 2×1536	4.61	7.06	9.13	10.60	12.39	14.28	9.68
LSTM-DRNN 1×256	7.30	9.31	11.14	12.38	14.15	15.93	11.70
LSTM-DRNN 2×256	7.94	9.89	11.68	12.92	14.60	16.35	**12.23**
LSTM-DRNN 3×256	7.64	9.69	11.52	12.70	14.46	16.18	12.03
Oracle (IRM)	*13.91*	*15.26*	*16.52*	*17.38*	*18.91*	*20.49*	*17.08*

dictionary size) and $\mu = 5$ (sparsity weight), based on limited parameter tuning on the CHiME-2013 development set [216].

Likewise, for feedforward DNNs $T_L = 8$ and $T_R = 0$ are used, whereas the parameter α, as well as the DNN and DRNN topologies (number of hidden layers and units per layer) are validated on the CHiME-2013 development set (cf. below).

In neural network training, the weight matrices are estimated by supervised training according to (3.103) (mask approximation objective) and (3.104) (discriminative objective). The time-frequency masks $\mathbf{y}_t^*$ and source spectra $\mathbf{s}_t$ used as training targets are derived from the parallel noise-free and multi-condition training sets of the CHiME-2013 data (cf. Section 2.2.1.1). Accordingly, discriminative NMF training is performed following the algorithm from Section 3.6, using the least-squares (D_2) objective (3.109) in analogy to the DNN training, yet 'down-sampling' the data by a factor of ten to cope with memory requirements. The sparse (non-discriminative) NMF baseline is implemented by training speech and noise dictionaries according to the objective (3.27), using the same data (sub-)set as for discriminative NMF.

In D(R)NN training, inputs are logarithmized to compress the dynamic range, which is considered useful in speech processing. Furthermore, the input features are globally mean and variance normalized on the training set, this kind of normalization allowing for on-line processing at run time. The D(R)NN hidden layer(s) have hyperbolic tangent activation functions, while the output layer has a sigmoid activation function, such that each output activation is forced to the range $[0, 1]$. All weights are randomly initialized with Gaussian random numbers ($\mu = 0$, $\sigma = 0.1$). For DNN training, 'discriminative' pre-training is used [246], i.e., building the DNN layer by layer by backpropagation (as opposed to generative pre-training).

The DNNs and DRNNs are trained through stochastic gradient descent with

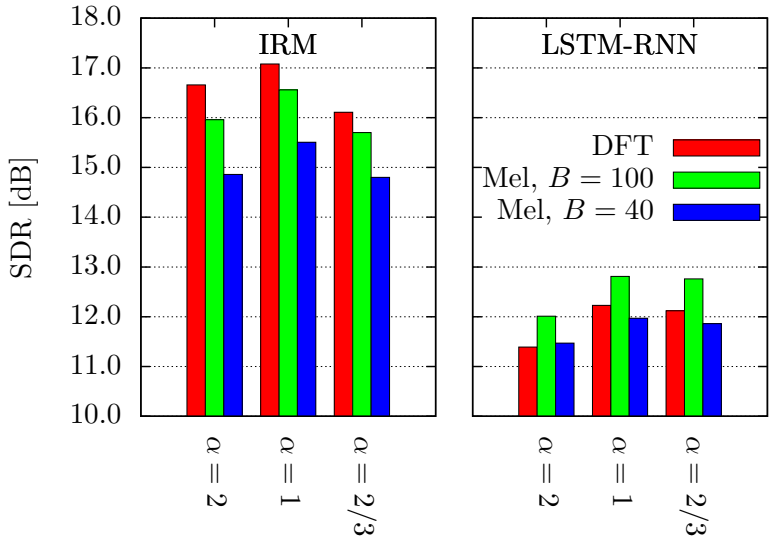


Figure 4.4: SDR on the CHiME-2013 development set with oracle masking (ideal ratio mask, IRM) as well as LSTM-DRNN based mask estimation for various spectral warping exponents α used in computation of DFT and Mel spectra ($B = 40$, $B = 100$).

an initial learning rate of 10^{-5} and a momentum of 0.9. Weights are updated after mini-batches of $|\mathcal{B}| = 25$ feature sequences. Since in DRNN training, there is no parallelism across time steps that can be exploited for GPU training (cf. Section 3.5.5), to increase the efficiency of DRNN training, the utterances are 'chopped' into sequences of at most $T = 100$ timesteps (but not shorter than $T = 50$).

Two common strategies are used to minimize over-optimization on the training set. First, Gaussian noise ($\mu = 0$, $\sigma = 0.1$) is added to the inputs in the training phase. Second, an early stopping strategy is used where the objective function is evaluated on the development set after each training epoch and the best network is selected accordingly. Training is stopped as soon as no improvement on the development set is observed for ten training epochs, or after 100 epochs have elapsed in total.

The open-source DNN and LSTM-DRNN training software CURRENNT co-authored by the author of this thesis (cf. Section 3.5.5) has been used for the experiments to enable reproducibility, in conjunction with the publicly available evaluation database.

4.2.2 Results

4.2.2.1 Neural network topologies

Table 4.1 shows the source separation performance using various network architectures and dimensions. Best DNN results are obtained with 3 layers and 1024 units per layer (10.46 dB SDR), whereas for 4 layers the performance saturates. 1.0 dB SDR is gained by increasing the depth from 1 to 3 layers, whereas increasing the breadth of the network to 1536 units does not seem to help. LSTM-DRNN can achieve up to 12.23 dB SDR with a much smaller model size (3×1024 DNN: 4.1 M trainable parameters, 2×256 LSTM-DRNN: 1.0 M), indicating a clear benefit of explicitly modeling temporal dependencies. Interestingly, the benefit of adding depth to LSTM-DRNN (besides their inherent depth in time) seems to be comparatively minor for the de-noising task, leading to competitive results even with a single layer (11.70 dB).

4.2.2.2 Influence of feature representation

Having 'tuned' the network topology, different feature representations are evaluated: DFT vs. Mel spectra, and the choice of the spectral warping exponent α. Figure 4.4 first shows the influence of α on the oracle masking performance as well as the results obtained with supervised training of mask estimation with LSTM-DRNNs. As is expected, in the oracle case the full-resolution mask delivers the best SDR. Regarding compression, $\alpha = 1$ (magnitude spectrum) works best. Conversely, when the estimated mask is used, best results are obtained with Mel masks ($B = 100$), and the full-resolution mask works only slightly better than the low-resolution ($B = 40$) Mel mask. Since for $B = 100$, the lower Mel bins correspond to single DFT bins while the higher Mel bins comprise multiple DFT bins, this indicates difficulties in precisely estimating the mask for the higher frequencies, which could be due to insufficient training data. Furthermore, while 'auditory' spectra ($\alpha = 2/3$) deliver clearly the worst performance in oracle masking, they are on par with magnitude spectra in case of mask estimation. Apparently, using compression with $\alpha = 2/3$ eases the optimization of the cost function enough to compensate for the lower attainable performance in oracle masking. Overall, the performance differences stemming from the feature representation are astonishing. In the DFT power spectrum domain, 11.39 dB average SDR are obtained, compared to 12.81 dB in the Mel magnitude domain with $B = 100$.

4.2.2.3 Discriminative training

Figure 4.5 shows the impact of using discriminative objective functions for $\alpha = 1$. Interestingly, when training LSTM-DRNNs using the the discriminative objective E^{DT} (3.104), (3.106) ('DT' in Figure 4.5), worse performance is obtained than with the standard mask approximation objective E^{MA} (3.58) ('MA' in Figure 4.5). In

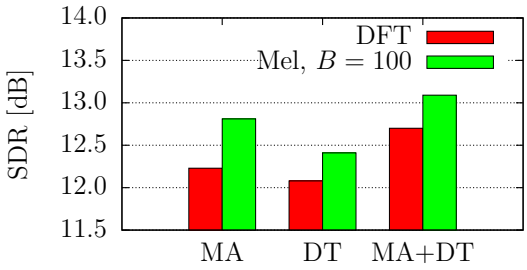


Figure 4.5: SDR on the CHiME-2013 development set with LSTM-DRNN time-frequency masking, trained with the mask approximation (MA) and discriminative training (DT) objectives, and discriminative retraining of LSTM-DRNNs trained with MA (MA+DT). Mel ($B = 100$) and DFT magnitudes ($\alpha = 1$).

Table 4.2: Source separation performance for selected systems on CHiME-2013 test set ($\alpha = 1$). Mel: $B = 100$.

SDR [dB]	Mel	DT	Input SNR [dB]						
			-6	-3	0	3	6	9	Avg.
Noisy			-2.27	-0.58	1.66	3.40	5.20	6.60	2.34
NMF [216]	–	–	5.48	7.53	9.19	10.88	12.89	14.61	10.10
NMF [216]	–	√	6.61	8.40	9.97	11.47	13.51	15.17	10.86
DNN	–	–	6.89	8.82	10.53	12.25	14.13	15.98	11.43
DNN	–	√	7.89	9.64	11.25	12.84	14.74	16.61	12.16
DNN	√	√	8.36	10.00	11.65	13.17	15.02	16.83	12.50
LSTM-DRNN	√	√	10.14	11.60	13.15	14.48	16.19	17.90	**13.91**
Oracle (IRM)	–	–	*14.53*	*15.64*	*16.95*	*18.09*	*19.65*	*21.24*	*17.68*

this case, convergence of the cost function was found to be sub-optimal. However, when starting from the solution obtained by training with E^{MA} until convergence, the results are significantly improved over MA ('MA + DT' in Figure 4.5). Yet, the results in the DFT domain using MA + DT are still below the results with Mel domain MA. Furthermore, if MA + DT is applied in the Mel domain, best results can be obtained (13.09 dB average SDR on the CHiME-2013 development set).

4.2.2.4 CHiME-2013 test set evaluation

The evaluation is concluded with a comparison of selected speech enhancement systems on the CHiME-2013 test set, cf. Table 4.2. The topologies for DNN and LSTM-RNNs as tuned on the development set are used (2×256 LSTM-DRNN and 3×1024 DNN, cf. Table 4.1). Comparing the results obtained with full-resolution magnitude spectra, it is observed that the DNN significantly outperforms NMF (according to a t-test comparing SDR measurements at the various input SNRs).

Note that without discriminative training, the comparison is not entirely fair, since NMF does not use parallel training data in this case. Discriminative training leads to a similar performance improvement for both. As on the development data, switching to the Mel magnitude domain ($B = 100$) further improves the results for the DNN. The gains by using the LSTM-DRNN network architecture are complementary, and 1.4 dB performance improvement are achieved with the LSTM-DRNN over a strong DNN baseline using Mel magnitudes and discriminative training, leading to the best result of 13.91 dB average SDR. While this corresponds to 11.6 dB gain over the noisy baseline, there is still a large gap to the performance attainable by oracle masking (17.68 dB).

4.2.3 Conclusions

By a comparative evaluation on the CHiME-2013 test set, it could be shown that regression-based approaches to speech source separation are effective, and that the discriminative training criterion based on optimal speech reconstruction in the DFT domain, as introduced in Section 3.6, can improve the performance of model-based approaches (here, NMF) as well as training-based approaches (here, D(R)NN) to speech enhancement. The overall best performance in real-time speech enhancement on the CHiME-2013 database was achieved by DRNNs operating in a reduced feature space (Mel domain). It is interesting that DRNNs outperform DNNs by a large margin in the present study, whereas this was not the case in earlier work by Huang et al. [87]; this can be attributed to avoiding the vanishing temporal gradient in conventional DRNN training – as used by Huang et al. [87] – by the LSTM principle. Furthermore, it is notable that the choice of feature representation has such a strong effect on the results, but this is in accordance with earlier studies showing that DNN acoustic models cannot compensate even for simple rotations of the input features [131]. Thus, in future work it will be investigated if the advantage of the Mel-domain feature reduction vanishes with more training data, which could allow for learning larger models supporting the separation of harmonics in the higher frequencies. Another goal will be to transfer the concepts of recurrent models to (discriminative) NMF.

5

Noise-robust front-ends for automatic speech recognition

It is not the voice that commands the story; it is the ear. – Italo Calvino

Having introduced the application of NMF and DNNs to speech separation, where the goal was to obtain noise-free time-domain signals, for example in human-*human* communication as by mobile telephony, the focus will now be laid on improving the results of automatic speech recognition, i.e., human-*machine* communication, by using NMF and DNN front-ends. The combination with DNN back-ends (acoustic models) will be addressed in Chapter 6.

5.1 Speech separation for robust ASR

5.1.1 Case study: The 1st CHiME Challenge

This section describes the main findings of the author and his colleagues obtained on the data of the first CHiME Challenge (CHiME-2011, cf. Section 2.2.1.1), as laid out in more detail in [211, 222, 242]. The key idea is to use NMF not only for speech enhancement, but also directly for noise-robust speech recognition, as is described below. The NMF recognition approach is based on the ideas simultaneously discovered by Gemmeke and Virtanen [55] as well as the author and his colleagues [172]. Building on these foundations, a system is proposed that combines NMF-based speech enhancement and speech recognition methods, treating them as separate systems despite their overlap in methodology. The research objectives on the CHiME-2011 Challenge data in this section are to investigate the benefits of (a) combining these NMF methods, and (b) of adding these components to a strong ASR baseline using a multi-stream BLSTM-DRNN acoustic model, multi-condition training, and model adaptation.

5.1.1.1 Word prediction by sparse NMF

Supervised sparse NMF as introduced in Section 3.2.3 can be exploited directly for multi-source recognition. This is done by assigning labels to each dictionary atom r for a source l of interest (typically speech). The labels can correspond to, e.g., phoneme or word identities, or corresponding HMM states. In case of exemplar-based NMF (3.33), these can be determined by forced alignment, since each dictionary atom $\mathbf{w}_r^l$ corresponds to a short-term spectrum from the training set, and a sequence of spectra in case of frame stacking or NMD. Assuming L classes, R^l dictionary atoms for the source of interest, and P short-term spectra per atom (frame stacking or NMD), the labels can be formally defined as a $L \times R^l$ matrix $\mathbf{Y}^l = (y_{c,r}^l)$ with class occupancy probabilities

$$y_{c,r}^l = \Pr(c \mid r) = \frac{\sum_{t=\tau_r}^{\tau_r+P-1} \delta(c_t, c)}{P}, \tag{5.1}$$

where c and r are the indices of the class and atom, c_t is the forced alignment label at time frame t in the training data, τ_r is the start frame of the atom r, and δ is the Kronecker delta.

At test time, the activation weights $\mathbf{H}^l$ obtained in supervised NMF on a mixture spectrogram $\mathbf{M}$ are normalized column-wise, yielding a matrix $\widetilde{\mathbf{H}}^l = (\widetilde{h}_{r,t}^l)$ with $\sum_r \widetilde{h}_{r,t}^l = 1$. Thus, for each time frame t pseudo-probabilities $\widetilde{h}_{r,t}^l = \Pr(r \mid \mathbf{m}_t)$ are computed. Then,

$$\mathbf{N}^l = (n_{c,t}^l) = \mathbf{Y}^l \widetilde{\mathbf{H}}^l \tag{5.2}$$

contains the class pseudo-posteriors for each time frame in the test signal, $n_{c,t}^l = \Pr(c \mid \mathbf{m}_t)$. This technique is known as *non-negative sparse classification* (NSC) [89]. Standalone recognition results on the CHiME-2011 database in a hybrid ASR system using NSC (in analogy to the hybrid DNN-HMM approach outlined in Section 3.7.2.2) are given by Hurmalainen et al. [89]. The non-negative modeling of the wanted and the interfering spectral patterns with corresponding activations provides inherent source separation capabilities and hence allows for high robustness in speech recognition, particularly in low SNRs [57, 89]. For each time frame t, the most likely label n_t is obtained as

$$n_t = \arg\max_c n_{c,t}^l \tag{5.3}$$

In the experiments described in this section, the resulting label sequence $(n_t)_{t=1,\ldots,T}$ is used as a feature stream in a multi-stream HMM decoder, in analogy to the multi-stream DNN-HMM approach outlined in Section 3.7.2.3.

5.1.1.2 Experiments

The architecture of the proposed system using a multi-stream HMM acoustic model with BLSTM-DRNN and NSC feature streams is shown in Figure 5.1. The triple-

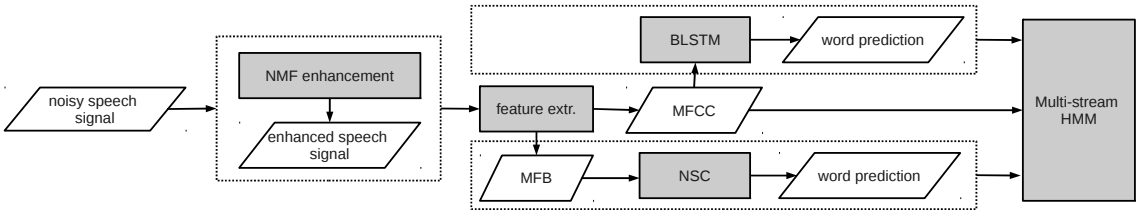


Figure 5.1: Block diagram of the proposed system [222]: The central component is a multi-stream HMM acoustic model fusing MFCCs with optional word predictions by non-negative sparse classification (NSC) operating on Mel frequency bands (MFB), and/or the BLSTM-RNN (processing MFCC features). The MFCC as well as MFB feature extraction can optionally by performed on an enhanced speech signal, applying convolutive NMF as pre-processing.

stream system using n_t in conjunction with a BLSTM-DRNN prediction b_t has the emission probability

$$p(\mathbf{x}_t, b_t, n_t|s_t) = p(\mathbf{x}_t|s_t)^{\mu} \times p(b_t|s_t)^{\gamma} \times p(n_t|s_t)^{\nu} \tag{5.4}$$

in the HMM state s_t.

As a preprocessing step in the front-end, the multi-stream architecture uses speech enhancement by convolutive NMF (cf. Section 3.5.1) as in [211]. Convolutive NMF dictionaries are estimated in advance for both speech and noise, resulting in a supervised NMF scheme as in [211]. For each of the 51 words $w = 1, \ldots, 51$ and each of the 34 speakers $i = 1, \ldots, 34$ in the CHiME-2011 training data, the corresponding segments of the noise-free CHiME-2011 training utterances are extracted according to an HMM forced alignment and concatenated into a single spectrogram which is reduced to a dictionary atom $\mathbf{W}^{(s,i,w)}(p)$ by convolutive NMF using the generalized KL divergence D_2 as cost function, using the updates (3.66) and (3.67) with $\beta = 2$ and $P = 13$. From the 51 speaker-specific word spectrograms for speaker i, speaker-dependent dictionaries $\mathbf{W}^{(s,i)}(p) = [\mathbf{W}^{(s,i,1)}(p) \cdots \mathbf{W}^{(s,i,51)}(p)]$ are formed. A general noise dictionary is obtained by sub-sampling the 4 hours of training noise provided with the CHiME-2011 corpus and applying convolutive NMF with $R^{(n)} = 51$ components, resulting in a dictionary $\mathbf{W}^{(n)}(p)$.

The features for convolutive NMF are STFT magnitudes with 64 ms frame size and 16 ms frame shift, and hence use longer windows than those commonly used in speech recognition. This has been proven beneficial for the quality of NMF-enhanced signals [185, 220], cf. also Section 4.1.

At test time, since the speaker identity i is assumed to be known in accordance with the CHiME-2011 evaluation protocol [7], the speaker-specific dictionary $\mathbf{W}^{(s,i)}$ and the noise dictionary $\mathbf{W}^{(n)}$ are concatenated to form the supervised convolutive NMF dictionary. After obtaining the NMF activations, a speech estimate is computed using (3.69), and a time domain signal is resynthesized for further processing in the

multi-stream recognizer.

The temporal resolution used for NSC-based speech recognition corresponds to a typical ASR setup – 25 ms frame size and 10 ms shift. As spectral features, 26 Mel scale magnitude bands ($B = 26$) are used. This resolution is believed to capture most of the information needed for direct classification, while keeping the computational complexity manageable [57, 89]. Exemplar windows spanning $P = 20$ frames are used in a frame stacking approach, based on earlier results on CHiME-2011 data [89]. On these data, using convolutive NMF instead of frame stacking did not significantly improve results [88]. The other factorization options, including weighting of features, sparsity penalty values and the number of iterations were exactly set as in [89]. The dictionaries $\mathbf{W}^{(s)}$ and $\mathbf{W}^{(n)}$ used in the experiments consist of 5 000 speaker-dependent speech exemplars and 5 000 noise exemplars, which are extracted from the CHiME-2011 training data. Here, the label matrix $\mathbf{Y}^{(s)}$ is chosen so as to correspond to word identities, i.e., $L = 51$, that are obtained by forced alignment using the CHiME-2011 baseline HMM system [21].

The BLSTM network applied for generating the estimates b_t for the multi-stream system is trained on framewise word targets obtained via HMM-based forced alignment of the clean training set. Similar to the network configuration used in [238], the BLSTM network consists of three hidden LSTM layers with sizes of 78, 150, and 51 hidden units per input direction. Subsampling layers of size 39 and 75 are used for information reduction between layers. The remaining training configurations are the same as in [238]. To create speaker-dependent networks, the BLSTM word predictor is adapted by performing additional training epochs using only the training utterances of a single speaker. For each speaker, a network is generated by initializing with the weights of the speaker independent networks and training until no further improvement on the development data of the chosen speaker can be observed. By using speaker-dependent BLSTMs, performance on the CHiME test set could be improved by about 3 % absolute with respect to speaker-independent BLSTM networks [211]. The stream weights are chosen for the double-stream MFCC-BLSTM case based on limited parameter tuning on the development set as in [211]: $\mu = 1.3, \gamma = 0.7, \nu = 0$. For the MFCC-NSC stream combination the weights are $\mu = 1, \beta = 0, \nu = 1$, and for the triple-stream model the weights are $\mu = \gamma = \nu = 1$.

The proposed system is evaluated against the baseline provided by the 2011 CHiME Challenge [21] organizers. As a basic technique for increased robustness, mean-only MAP adaptation with $\tau = 5$ (3.142) is used to estimate speaker-dependent GMMs modeling the MFCC stream, and MCT is used for both the MFCC GMMs and the weights within the BLSTM layers. The MCT data is generated by mixing all 17 000 training utterances with random segments of the training noise in the CHiME corpus. Thus, the complete MCT (clean and noisy) data consists of 34 000 utterances. Note that it is intentional that the noise or speech levels are not scaled to obtain specific SNRs for training, as the SNR conditions in the test data are assumed to be unknown.

Table 5.1: Keyword recognition accuracies [%] on the CHiME corpus using multi-stream HMMs with MFCC, BLSTM, and/or non-negative sparse classification (NSC) feature streams. –: not present ($\lambda_i = 0$), $\surd$: present, $\surd$+: computed from NMF enhanced signal.

Streams			**Devel Mean**	**Test SNR** [dB]						**Test Mean**
MFCC	BLSTM	NSC		-6	-3	0	3	6	9	
CHiME Challenge Baseline										
$\surd$	–	–	56.3	30.3	35.4	49.5	62.9	75.0	82.4	55.9
Speaker adaptation / multi-condition training										
$\surd$	–	–	74.6	54.5	61.1	72.8	81.7	86.8	91.3	74.7
$\surd$+	–	–	82.7	75.6	79.2	84.1	87.7	88.3	90.6	84.2
$\surd$	$\surd$	–	86.5	72.8	79.0	85.4	90.8	93.8	95.8	86.3
$\surd$+	$\surd$+	–	90.1	82.9	87.2	90.3	93.7	93.9	94.8	90.5
$\surd$	–	$\surd$	85.3	67.2	75.1	85.0	89.8	92.0	93.4	83.7
$\surd$+	–	$\surd$	89.3	79.1	82.8	88.7	91.2	92.7	93.5	88.0
$\surd$+	–	$\surd$+	88.2	80.4	83.2	87.5	89.9	90.3	92.8	87.4
$\surd$	$\surd$	$\surd$	91.0	76.9	82.9	88.8	92.3	95.3	96.4	88.8
$\surd$+	$\surd$+	$\surd$	**92.6**	84.8	88.3	92.1	93.9	95.7	96.4	**91.9**
$\surd$+	$\surd$+	$\surd$+	92.1	84.5	87.9	91.0	93.5	95.0	95.6	91.3

The HMM topologies of the proposed system correspond to the baseline [21]. Left-to-right word-level HMMs with 4–10 states and seven Gaussian mixtures per state are employed. 39-dimensional cepstral mean normalized MFCC features (MFCCs 1–12 with energy, deltas and delta-deltas) as in the baseline are used.

5.1.1.3 Results

Experimental results on the development and test set of the CHiME-2011 corpus are shown in Table 5.1. In accordance with the CHiME-2011 Challenge setup, the evaluation measure is keyword accuracy, which is the accuracy (2.16) obtained on the 25 letters (A-Z without W) and 10 digits in the ASR task. A noticeable improvement of almost 19 % absolute in keyword accuracy (on test) is gained by using MCT and mean-only MAP adaptation for the GM modeling of the MFCC stream, as detailed in [211]. When using only the MFCC stream in a MAP-adapted HMM with MCT, NMF enhancement delivers a gain of about 10 % absolute as reported in [211]. Still, this improvement is mostly visible for lower SNRs, while at 9 dB SNR there is a slight degradation. Considering a noise-robust back-end including additional BLSTM modeling without any front-end enhancement yields 86.3 % average accuracy;

while the improvement by the BLSTM is slightly smaller than the one by NMF enhancement at low SNRs, significant gains are now observed at the highest SNR level (4.5 % absolute at 9 dB, $p < .001$ according to a z-test). Using NMF enhancement in combination with BLSTM modeling gives an average accuracy of 90.5 %, again boosting the performance at low SNRs while inducing a slight degradation at 9 dB SNR: It appears that at 9 dB the BLSTM alone delivers predictions so robust that NMF separation artifacts outweigh the benefit of additional noise suppression.

Modeling the NSC word prediction in analogy to the BLSTM in a double-stream HMM, 83.7 % average accuracy are obtained without prior speech enhancement. Most notably, this accuracy is boosted to 87.4 % when using NMF enhancement in addition to NSC, indicating that the NMF and NSC approaches both contribute to robustness: It can be argued that although NMF is the basis of both, the input representations and dictionaries are considerably different (cf. Sections 5.1.1.1 and 5.1.1.2) – in fact, NSC was shown to produce better recognition of noisy speech than recognition of enhanced signals reconstructed from the same sparse representation [57]. Interestingly, even higher performance (88.0 %) is observed when using enhancement only for the MFCC stream: Enhancement degrades performance of the MFCC-NSC model starting from 0 dB SNR. This can probably be attributed to a mismatch of the NSC dictionaries, which are built from unprocessed speech and noise data, and the characteristics of the separated signal with separation artifacts and remaining interferences.

Finally, the overall best results are obtained by a triple-stream HMM fusing the enhanced MFCC stream with BLSTM and NSC word predictions, reaching 91.9 % keyword accuracy on the test set. Again, using NMF enhancement prior to NSC does not further improve performance; yet again, without NMF enhancement at all, performance is considerably lower (88.8 %). Notably, it can be seen that the triple-stream approach significantly ($p < .005$) outperforms both double-stream approaches, providing evidence for complementarity between the BLSTM and NSC streams.

5.1.1.4 Conclusions

A highly effective system exploiting NMF for speech enhancement as well as directly for speech recognition was introduced. In combination with a BLSTM-RNN multi-stream HMM acoustic model 91.9 % average keyword accuracy were achieved on the CHiME-2011 data set containing highly non-stationary noise at SNRs from -6 to 9 dB. This was the best result reported on these data in 2012 [222]. Using a similar system, yet with larger dictionaries for NMF enhancement, the author and his colleagues were later able to obtain the best results on the CHiME-2013-SV corpus [52] (93.9 % accuracy). The results presented in this section suggest that NMF enhancement and recognition are complementary; it remains to investigate whether this is due to different parameterizations (features, dictionaries) or fundamental

methodological differences. Furthermore, since the triple-stream approach delivers best results, it can be argued that robustness by flexible context modeling in the BLSTM is complementary to explicit noise modeling in NSC. This result, and the consistent gains by NMD speech enhancement, suggest that explicit source separation is complementary to state-of-the-art acoustic models. In [52], the author and his colleagues also demonstrated how to generalize the triple-stream system presented in this section to large vocabulary ASR.

5.1.2 Case study: Noise-robust conversational ASR

The study on noise-robust *conversational* ASR presented in this section [223] was motivated by the fact that earlier studies on source separation for noise-robust ASR, such as the one described above, were restricted to small vocabulary, prompted speech. ASR in many realistic scenarios, including hands-free natural human-computer interaction and multimedia retrieval, has to deal with spontaneous speech on the one hand, and interfering audio sources on the other hand. To overcome the shortcomings of previous studies, the evaluation is carried out on the CHiME-2011-Buckeye database created by the author and his colleagues (cf. Section 2.2.1.2). This database enforces a stricly speaker independent evaluation for speech enhancement as well as ASR.

To address the variability of spontaneous speech as well as non-stationary noise, a system is proposed which is structurally similar to the CHiME-2011 system presented above and in [211], yet uses a Tandem BLSTM-DRNN HMM acoustic model instead of the double-stream HMM. This is because context-sensitive Tandem BLSTM-DRNN acoustic models (cf. Section 3.7.2.1) have been shown to lead to remarkable performance gains in conversational speech recognition [235], outperforming the multi-stream HMM approach. Furthermore, semi-supervised phoneme-based NMF is employed, which is arguably more suited to the larger vocabulary task at hand and provides unsupervised adaptation to background noise, as was shown for the case of music noise in Section 4.1.

5.1.2.1 Experiments

For speech enhancement on the development and test set, semi-supervised sparse NMF (cf. Section 3.2.5) was used. Spectrograms of the mixture signals were calculated by short-time Fourier Transform (STFT) using Hann windows of 25 ms length at 10 ms frame shift. A shorter window size and frame shift than in the author's and his colleagues' previous study on the small vocabulary CHiME Challenge ASR task (cf. [211] and the previous section) were chosen to cope with higher variability of spontaneous conversational speech.

In accordance with the medium vocabulary and speaker independent ASR task, a phoneme-dependent yet speaker-independent speech dictionary was constructed

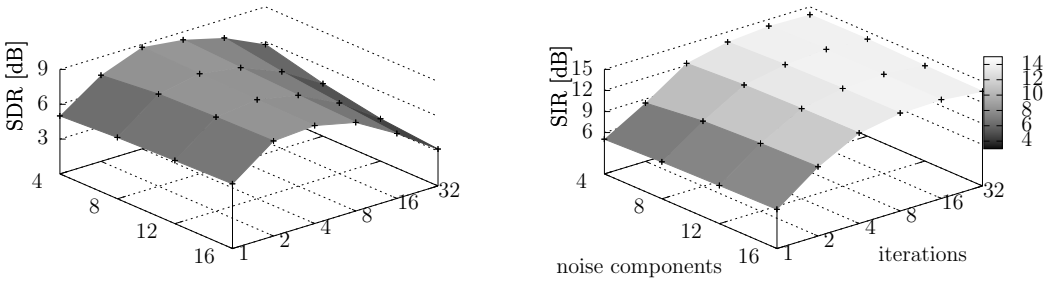


Figure 5.2: Signal-to-distortion ratio (SDR, top) and signal-to-interference ratio (SIR, bottom) on the CHiME-2011-Buckeye development set (average across 6 SNRs from -6 to 9 dB), by number of noise components and number of iterations in semi-supervised sparse NMF ($\mu = 0.1$).

for NMF, instead of a word- and speaker-dependent speech dictionary as used for speech enhancement in the above. To this end, for each phoneme, the corresponding spectrograms were extracted from the Buckeye training set according to a forced alignment with the recognizer described in [235]. These concatenated phoneme spectrograms were reduced to a single dictionary atom by a 1-component NMF. The column-wise concatenation of these atoms builds the matrix $\mathbf{W}^{(s)}$. Thus, the number of speech components $R^{(s)}$ in semi-supervised NMF was equivalent to the number of phonemes (39). The advantage of such phoneme-dependent speech dictionaries over unsupervisedly learnt ones has been shown in [166].

The number of noise components $R^{(n)}$ as well as the sparsity constant μ and the number of NMF iterations K were optimized in a limited three-dimensional grid search on a subset of the development set which consisted of 10 randomly selected utterances of each speaker at 6 SNRs (parameter ranges: $R^{(n)} \in \{4, 8, 12, 16\}$, $\mu \in \{0, 0.01, 0.1, 1\}$, $K \in \{1, 2, 4, 8, 16, 32\}$). This is in contrast to the experiments with semi-supervised NMF in Section 4.1 which kept these parameters fixed. The separation performance was measured in terms of SDR, SIR and SAR (cf. [200] and Section 2.1.1.2). The overall best SDR (8.8 dB on average from -6 to 9 dB SNR) was obtained for 4 noise components, 4 NMF iterations and $\mu = 0.1$, which is a gain of more than 4 dB over the noisy data (average SDR = 4.5 dB). As can be seen from Figure 5.2, higher numbers of iterations tend to decrease SDR especially for a high number of noise components. More precisely, additional iterations increase noise suppression in terms of SIR at the expense of introducing artifacts (decreasing SAR); this can be explained by (over-)fitting of the noise components to parts of the observed speech, due to the mismatch with the speaker-independent speech dictionary. Conversely, increasing $R^{(n)}$ is only slightly advantageous for the SDR in the case of 1 or 2 iterations. This is somewhat expected, as the number of noise sources present in a single speech turn is limited, and thus over-fitting occurs if the noise components

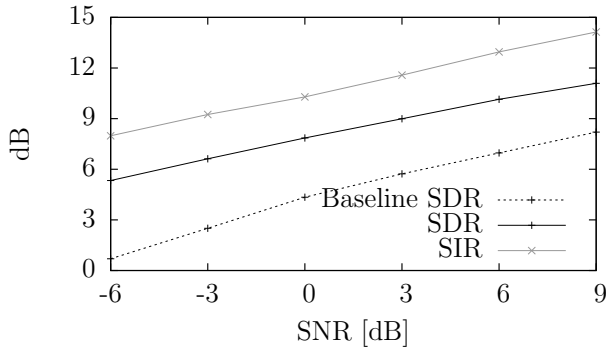


Figure 5.3: Separation performance on the CHiME-2011-Buckeye test set: Baseline SDR and SDR / SIR after applying sparse semi-supervised NMF ($K = 4$, $R^{(n)} = 4$, $\mu = 0.1$) for various input SNRs.

have too many parameters. Overall, the results of the parameter tuning corroborate previous findings of Joder et al. [96] for enhancement of conversational speech in a larger-scale and speaker-independent evaluation.

As the acoustic model, a Bottleneck-(BN-)based Tandem approach incorporating a BLSTM-DRNN was used (cf. Section 3.7.2.1). The network consisted of three hidden layers (per input direction) and was trained on framewise phoneme targets obtained via HMM based forced alignment of the clean Buckeye training set. All network and training parameters, including the size of the hidden layers, learning rate, etc. were set exactly as in [235]. Only the first 39 principal components of the PCA-transformed BN-BLSTM feature vector were used as final features for tandem ASR. In the HMM system applied for processing the tandem and BN-BLSTM features each context-dependent phoneme is represented by three emitting states (left-to-right HMMs) with 16 Gaussian mixtures. Tied-state cross-word triphone models with shared state transition probabilities were applied. The clean acoustic models as well as a back-off bigram language model were trained on the training set of the Buckeye corpus. Multi-condition acoustic models were trained accordingly, on the training set of the CHiME-2011-Buckeye corpus.

5.1.2.2 Results

The separation by sparse semi-supervised NMF, with parameters optimized on the development set as described in Section 5.1.2.1, is evaluated on the test set in Figure 5.3. A constant and significant gain over the noisy SDR baseline is observed; however, the SDR gain decreases with increasing input SNR, ranging from 4.6 dB (-6 dB SNR) down to 2.9 dB (9 dB SNR). Furthermore, NMF boosts the SIR by 7.3 dB at -6 dB SNR and by 5.9 dB at 9 dB SNR.

Considering the word accuracies of ASR (Table 5.2), drastic decreases are observed in noisy conditions compared to clean data; however, by using NMF, a consistent

Table 5.2: Word accuracies [%] on Buckeye test set at SNRs from -6 to 9 dB, on average across these SNRs, and for clean speech. $\mathbf{x}_t$: acoustic feature vector (MFCC or tandem, (3.149)). MCT: multi-condition training.

$\mathbf{x}_t$	**NMF**	**MCT**	**SNR**							**clean**
			-6 dB	-3 dB	0 dB	3 dB	6 dB	9 dB	avg	
MFCC	–	–	21.21	23.11	25.40	27.85	30.85	34.48	27.15	50.97
MFCC	–	√	25.25	27.36	30.09	31.59	34.20	37.00	30.92	43.84
MFCC	√	–	23.06	25.32	27.17	29.65	32.56	36.48	29.04	50.54
MFCC	√	√	26.51	28.82	30.85	32.85	35.13	37.95	32.02	43.83
Tandem	–	–	22.73	25.08	28.13	30.51	35.16	39.04	30.11	58.21
Tandem	–	√	34.93	37.58	40.04	41.71	44.60	46.87	40.96	51.12
Tandem	√	–	24.47	26.79	29.75	32.18	36.53	40.74	31.74	57.94
Tandem	√	√	35.74	38.45	40.49	42.45	45.27	47.29	41.62	50.91

gain of around 2 % absolute WER across all considered SNRs is achieved over the noisy baseline. The latter is in contrast to the study on the CHiME-2011 database (cf. [211] and the previous section), where a downgrade had to be accepted at high SNRs when using NMF. This can be attributed to the optimization of the NMF parameters on SDR which effectively leads to using much less NMF iterations than in [211] (4 instead of 100).

In contrast to NMF, using MCT improves the performance of the MFCC front-end particularly in highly noisy conditions, but a severe downgrade of 7 % absolute is observed for clean speech; clean speech, in turn, seems to be largely unaffected by applying NMF. By combining NMF and MCT, the results on noisy speech can be further improved, but the downgrade for clean speech remains. This downgrade along with the low accuracies in noisy conditions indicate the difficulty of modeling highly variable speech and noise at the same time.

The BN-BLSTM features deliver consistently higher word accuracies than the MFCC front-end, both with and without NMF; the gain by using the BN-BLSTM approach instead of MFCC features increases with the SNR and the largest improvement (7 % absolute, up to 58.21 %) is found for clean speech. Finally, the overall best result across noisy speech (35.74 % to 45.27 %, mean = 41.62 %) and clean speech (50.91 %) is observed when combining BN-BLSTM with NMF and MCT. Overall, it seems that the BN-BLSTM acoustic model can profit much more from training with noisy data than the GMM-HMM with MFCC features. This is in accordance with more recent findings on DNN-based acoustic modeling of noisy speech [247].

5.1.2.3 Conclusions

A large scale study on speaker independent recognition of spontaneous speech in various levels of interfering non-stationary noise has been carried out. Significant gains could be achieved by a combination of NMF and BN-BLSTM, and optimization of NMF on SDR could avoid a downgrade in high SNRs and clean conditions. Still, the word accuracies indicate that this task remains highly demanding, especially due to the interfering speakers occuring in the CHiME-2011 noise used for evaluation; the latter condition is especially challenging for monaural separation algorithms.

Furthermore, it is found that the drastic gains in SDR by NMF only correspond to slight gains in ASR accuracy, especially in the case of the noise-robust BN-BLSTM frontend. This is in contrast to the results on 'command and control' speech in the CHiME-2011 Challenge task [211] (cf. also the previous section), where NMF improved by a larger margin over the BLSTM baseline. This observation could be attributed to insufficient power of simple linear, low dimensional NMF models in the case of spontaneous speech, in contrast to non-linear BLSTM-DRNN acoustic models with many more trainable parameters. As an extension, it could be promising to exploit larger NMF models, including exemplar-based, sparse, and discriminative NMF. Besides, the mismatch of ASR accuracies and source separation metrics deserve further investigation, in order to enable optimization of source separation for ASR without costly task based evaluations.

5.2 Feature enhancement: A CHiME-2013 benchmark

Robustness of ASR systems can be addressed at different stages of the recognition process [171], and successful systems usually employ a combination of them [7]. In this thesis, so far, speech and noise separation by NMF in the front-end, as well as back-end model adaptation and acoustic modeling using DNNs have been considered to increase ASR robustness in the case of single-channel signals with non-stationary noise. Yet, 'in between' one can also address noise-robust features – a popular expert crafted feature extraction scheme is RASTA-PLP [79] – or feature enhancement, defining a mapping from noisy to noise free speech features. One advantage of considering feature enhancement is that enhanced features can – ideally – be used in a recognizer without back-end modification, allowing to treat the back-end as a 'black box' as long as the feature extraction procedure is known. An example for a data-based, non-parametric technique for feature enhancement is histogram equalization [26, 239]. The subject of this section will be to investigate feature enhancement based on supervised training of DRNNs, as was presented by the author of this thesis and his colleagues in [230]. In this case, the input features represent sequences of mixture features $g(\mathbf{m}_1, \ldots, \mathbf{m}_T)$, where $\mathbf{m}_t$, $t = 1, \ldots, T$, is a short-term spectrum of a noisy

and reverberated mixture and g is an acoustic feature extractor (e.g., computation of MFCCs with delta coefficients), and as training targets corresponding sequences of isolated speech features $g(\mathbf{s}_1, \ldots, \mathbf{s}_T)$ are used. Thus, the following cost function is minimized in training:

$$E^{\mathsf{FE}}(\mathbf{W}) = \sum_t |h_{\mathbf{W}}(g(\mathbf{m}_t)) - g(\mathbf{s}_t)|^2, \tag{5.5}$$

where $h_{\mathbf{W}}$ is a vector-valued regressor such as a DNN with trainable parameters $\mathbf{W}$ and $|\cdot|$ indicates the Euclidean distance. In this objective function, no assumptions are made on the relation between clean and corrupted speech features, such as assuming an additive mixing process, as was done for spectral domain enhancement by time-frequency masking (3.104). As a result, the above objective is very generic and can be applied to any type of speech feature and any type of (non-linear) corruption. In particular, this framework also comprises the case of feature dereverberation, in which case time-frequency masking is hardly applicable due to the wanted and interfering signals being highly correlated. However, depending on g, the relation between clean and corrupted speech features can be highly complex and hard to learn. It has been shown that DNNs can exploit non-linear relationships between noisy and clean features in the context of acoustic modeling [183], but it is not clear whether this also holds for feature enhancement.

Thus, a main research question to investigate in this section is the choice of g for optimal performance. In particular, feature enhancement in the logarithmic Mel frequency domain and in the cepstral domain are compared. Furthermore, it is investigated whether measures of the network regression performance are correlated to ASR performance, which involves much more complicated likelihood functions than the error functions typically used in network training (cf. Section 3.3). Finally, also the effects of using multi-condition training with reverberated and noisy speech, feature transformations, and discriminative back-end training are taken into account separately. This serves to verify the hypothesis that a feature enhancement step improves over state-of-the-art multi-condition trained ASR models.

Regarding the model for h, DRNNs are the method of choice, due to the success in other speech processing tasks (cf. above). Feature enhancement by RNNs has been considered before [121, 145]. In particular, BLSTM-DRNNs have been employed by Wöllmer et al. [236] for feature enhancement in highly non-stationary noise, by mapping noisy cepstral features to clean speech cepstral features, and have been shown to outperform traditional RNNs on this task. In [225], the author of this thesis and his colleagues successfully applied the BLSTM feature enhancement methodology to both ASR tasks (CHiME-2013 and CHiME-2013-SV) of the CHiME-2013 Challenge (cf. [202] and 2.2.1.1). There, the BLSTM approach outperformed a similar approach using conventional RNNs on the CHiME-2013-SV task [122]. In this section a larger scale evaluation of BLSTM-DRNNs and other types of neural networks – including feedforward DNNs – is presented on the CHiME-2013 (medium vocabulary) task.

5.2.1 Experiments

5.2.1.1 Feature enhancement front-end

The original contribution of the author and his colleagues to the CHiME-2013 Challenge [225], and a related contribution using standard RNNs [122] considered MFCCs as input and output of the feature enhancement networks. Using MFCCs is mainly an ad-hoc solution motivated by their use in a GMM-HMM speech recognition back-end with diagonal covariances. However, it is now well known that DNN-based speech recognition can be improved by directly using logarithmic Mel filterbank outputs (Log-FB) [67, 82]. Hence, it is interesting to compare MFCC and Log-FB in the context of feature enhancement. Note that considering Log-FB as training targets resembles multi-task regularization of the network due to the correlation in the targets, which is expected to help generalization.

26 Log-FB covering the frequency range from 20–8 000 Hz are used. Delta coefficients (3.138) are added both to the input and output features; using them as targets is similar in spirit to the proposal by [182] to use multi-frame information as training targets in neural network based speech recognition, which again serves to improve generalization. As additional feature in input and output, root-mean-square (RMS) energy is used along with its deltas. For the MFCC features, acceleration coefficients (second order deltas) are also added, and Cepstral Mean Normalization (CMN) is performed to compensate short-term channel responses. Thus, in the MFCC case, the network input and output exactly correspond to the ASR front-end used in the HTK CHiME baseline [202]. In the Log-FB case, the outputs can be converted to MFCCs by simply applying a Discrete Cosine Transformation (DCT) and cepstral liftering with the parameters listed in [245]. Log-FB features are investigated with and without log spectral subtraction, which is the Log-FB domain equivalent of CMN [54]. For reproducibility, feature extraction is done using the HTK [245], using the `MFCC_E_D_A_Z`, `FBANK_E_D` and `FBANK_E_D_Z` types of features with the default parameters [245].

Prior to feature extraction, the stereophonic signals from the CHiME-2013 corpus are down-mixed to monophonic audio by averaging channels, corresponding to simple delay-and-sum beam-forming. This is useful for the specific setup of the CHiME-2013 corpus, where the speaker is positioned at a frontal position with respect to the microphone, and is hence exploited also in the baseline system by Vincent et al. [202].

All features are globally mean and variance normalized. To this end, the global means and variances of the noise-free and the noisy training set feature vectors are computed, and mean and variance normalization of the network training targets and the network inputs are performed accordingly. This normalization was found to be very important for performance; in particular, it ensures that features with large variance due to noise do not 'mask' important information in features with lower

variance such as delta coefficients.

5.2.1.2 Network training

Feature enhancement networks are trained by minimizing (5.5), substituting the noisy training set for the features $\mathbf{m}_t$ and a noise-free training set for $\mathbf{s}_t$ in (5.5). In a first set of experiments, a de-noising network is trained, i.e., the network learns a mapping of noisy to noise-free features within the same acoustic environment. As a consequence, the output features will still be reverberated, and they will be used for decoding with a model trained on reverberated data. This is done to enable a comparison to previous results with BLSTM-DRNN feature enhancement on the CHiME-2013 corpus [225]. Additionally, learning mappings from noisy and reverberated to dry (noise-free, close-talk microphone) speech features will be considered, i.e., the network is forced to also learn the removal of convolutive noise in the ASR feature domain. In the latter case, deep learning with pre-training is also considered, cf. below. While the CHiME-2013 corpus also contains noise context for each utterance, this is not exploited here, i.e., only the end-pointed speech segments are used (so-called 'isolated utterances' in the CHiME-2013 corpus).

The networks are trained through on-line gradient descent (batch size $|\mathcal{B}| = 1$) with a learning rate of 10^{-5} and a momentum of 0.9. Prior to training, all weights are randomly initialized with Gaussian random numbers (mean 0, standard deviation 0.1). The on-line gradient descent algorithm applies weight changes after processing each utterance, using a random order of utterances in each training epoch to alleviate overfitting. Using on-line learning was found to drastically speed up convergence and increase generalization compared to (full) batch learning. Zero mean Gaussian noise with standard deviation 0.1 is added to the input activations in the training phase, and an early stopping strategy is used in order to further help generalization. The latter is implemented as follows: The cost function (5.5) is evaluated on the development set after every fifth epoch. Training is aborted as soon as no improvement in (5.5) on the development set has been observed during 30 epochs. The network that achieved the lowest cost on the development set (across all six SNRs) is chosen as the final network.

Most of the applied BLSTM networks have three hidden layers consisting of $2F$, 128, and $2F$ LSTM cells as described above, where F is the input and output feature dimension (39 for MFCC, 54 for Log-FB). Each memory block contains one memory cell. This topology was empirically determined on a similar speech feature enhancement task [236]. In case that noisy features are mapped to clean features, networks with four hidden layers incorporating $2F$, 128, $2F$, and $2F$ LSTM cells are considered in addition. The rationale behind this is that mapping to clean features is a more complex task than just removing noise, which also involves de-reverberation. Besides training the four hidden layers without additional constraints, it is also aimed at enforcing structure by pre-training of the first three hidden layers. In particular, a

fourth hidden layer is added to the three-layer network which has been trained to map noisy and reverberated to noise-free reverberated features, and then run additional training epochs using the same inputs, but clean features as targets. For the sake of consistency, the training parameters are set based on previous experience with RNN-based enhancement of conversational speech in noise [236]. For reproducibility, the open-source CURRENNT software written by the author and his colleague is used for BLSTM training (cf. Section 3.5.5). CURRENNT is delivered with a subset of the CHiME-2013-SV feature enhancement task as demonstration.

5.2.1.3 Baseline networks

To verify the effectiveness of BLSTM networks for feature enhancement, simpler network architectures are considered as baselines: bidirectional RNNs (BRNNs) and feedforward DNNs. In the case of DNNs, a certain degree of context-sensitivity is introduced by frame stacking (cf. Section 3.5). The sliding windows are of size 9 ($T_L = T_R = 4$). Consequently, the DNN training targets are positioned at the center frames of the sliding windows.

As DNN topologies, 'symmetric' hidden layers (3×256 units) are investigated as well as a structure that reduces information layer by layer (486, 256, and 108 hidden units), matching the size of the first hiddden layer to the input layer and the size of the third layer to two times the size of the output layer. BRNNs have the same size as BLSTM-RNNs (108, 128, 108 hidden units). Since BLSTM-RNNs have many more parameters than DNNs or BRNNs of the same hidden layer size, a smaller BLSTM net (81, 96, and 81 hidden units) is also investigated whose number of parameters compares to the simpler architectures. For a fair comparison, both BRNNs and DNNs were trained using the same stochastic gradient descent algorithm as BLSTM-RNNs, using random initialization. The learning rate for DNNs and BRNNs was tuned on the development set, and it was found that best performance with DNNs was obtained with 10^{-7}, as opposed to 10^{-5} for the BLSTM-RNNs, requiring more training epochs until convergence. BRNNs required setting a particularly low learning rate (here, 10^{-8}) in order for the training to converge. The latter is in accordance with the findings of Sutskever et al. [191].

5.2.1.4 Obtaining ASR features

As detailed above, the first step of obtaining enhanced ASR features is presenting the frame-wise noisy features (MFCC or Log-FB) $\mathbf{m}_t$ to the trained network and computing the denoised features $\mathbf{y}_t = h(\mathbf{m}_t)$ as the output activations. In principle, cepstral mean normalized MFCC features with deltas output by a network can be used 'as is' in the speech recognizer. However, due to the normalization of the training targets, $\mathbf{y}_t$ will be (approximately) mean and variance normalized, which does not match the features used to train the baseline models. Thus, to be able to

use the enhanced features in a 'plug-and-play' fashion, i.e., without any recognizer modification, the global mean and variance normalization is reverted after obtaining the enhanced MFCC features, to ensure compatibility with the means and variances of the trained recognition models. More specifically, each enhanced feature vector is multiplied element-wise with the corresponding variances of the noise-free training set, and the mean feature vector of the noise-free training set is added. For the Log-FB features, deltas output by the network are thrown away, the MVN is reverted as above, and cepstral mean normalized MFCC features with delta and acceleration coefficients are computed from the Log-FB features output by the network.

5.2.1.5 Speech recognition back-ends

In the following, the speech recognition back-ends used for evaluating the feature enhancement procedure are described.

5.2.1.5.1 Baseline models The performance of the enhanced features is first evaluated using the baseline models provided by the Challenge organizers, as well as in models which are re-trained with enhanced features. The baseline system is implemented using HTK [245] based on the WSJ-0 'recipe' by Vertanen [199]. From these models, a 'reverberated' baseline model is generated by EM-ML estimation on the reverberated training set. Four EM-ML iterations are used. The 'noisy' baseline model is created by four additional EM-ML iterations using the training set with convolutive noise. From these 'noisy' models, 're-trained' models are derived simply by repeating the MCT step using features that have been processed by the enhancement networks. This is done to investigate to which extent distortions by enhancement can be compensated by model re-training. Furthermore, it is expected that feature de-noising and de-reverberation results in lower feature variance, requiring model adaptation. In contrast, using the baseline models without modification serves to estimate the 'compatibility' of enhanced features with their clean counterparts used to train the ASR models. From an application point of view, it corresponds to a 'plug-and-play' configuration – in other words, a scenario where the recognizer back-end is a 'black box' and only the feature extraction front-end is known.

5.2.1.5.2 Discriminatively trained models with feature transformations The training procedure used to generate the CHiME baseline models does not use many state-of-the-art ASR techniques for GMM-HMM (cf. Section 3.7.1), such as feature transformations and discriminative training. Thus, it is of crucial interest to investigate whether the performance of state-of-the-art ASR, such as the back-end used by [192] for their (winning) contribution to the CHiME-2013 Challenge, can also be improved by the proposed feature enhancement technique. This system is implemented with the Kaldi speech recognition toolkit [152].

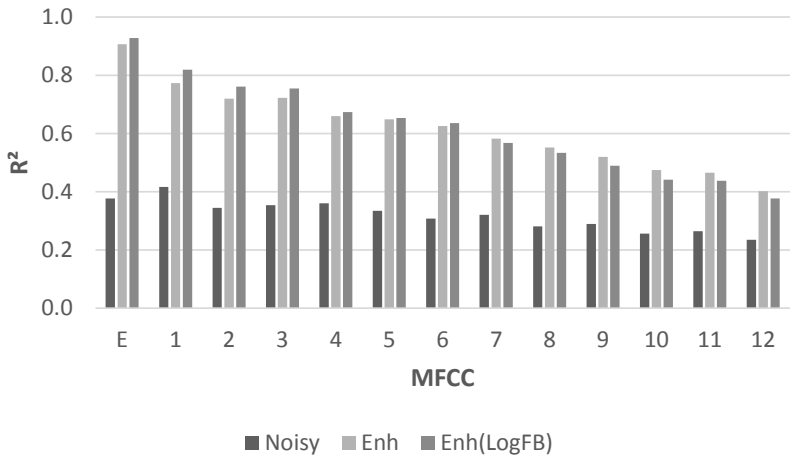
LDA-STC (Sections 3.7.1.1 and 3.7.1.2) is employed for feature transformation, using a sliding window of 9 frames ($T_L = T_R = 4$) of 13 MFCCs (0–12) for LDA and keeping the 40 first dimensions of the transformed 117-dimensional vector. The LDA classes are obtained by aligning the 2 500 tri-phone HMM states. Finally, since in the CHiME-2013 Challenge setup the speaker identities are assumed to be known, fMLLR speaker adaptation (cf. Section 3.7.1.4) is applied, and canonical models are obtained by Speaker Adaptive Training (SAT) [4], which corresponds to re-training the models using the fMLLR transformed features of the training utterances. For unsupervised adaptation at test time, a tight-beam decoding is performed on all test utterances of a single speaker to obtain a first pass transcription, which is used to re-estimate the fMLLR transform, before doing a final decoding.

Parameterization and training of acoustic models follows [192] and works as follows: 40 phonemes (including silence) are integrated in context-dependent triphone models with 2 500 states and a total number of 15 000 Gaussians. First, models are trained with clean training data applying the EM-ML principle. Next, EM-ML training is continued with reverberated training data, using the alignments and triphone tree structures from the clean models. Then, isolated noisy training data are used for training. Another set of EM-ML training iterations is then performed after applying the described feature transformations, using the noisy training data. Subsequently, features are transformed using LDA-STC and model re-estimation is done. Afterwards, fMLLR transforms are estimated for SAT, leading to another set of model re-estimation iterations. Based on the ML trained acoustic models, discriminative training is performed with the noisy training data, using bMMI (3.141) with a boosting factor of $b = 0.1$.

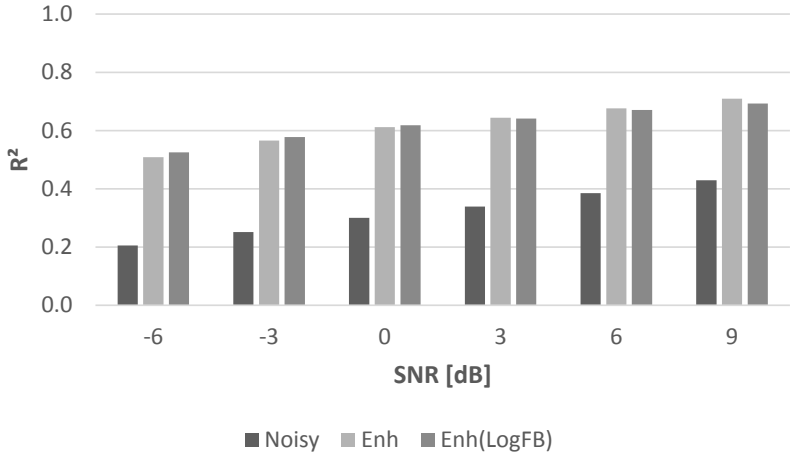
5.2.2 Results

5.2.2.1 Regression performance

Before turning to task-based ASR evaluation, let us first investigate the feature enhancement performance in terms of regression error. The determination coefficient R^2 (squared Pearson correlation coefficient) of the noise free features is computed with (i) the unprocessed noisy MFCC features, (ii) the MFCC features output by the MFCC enhancement network, and (iii) the MFCC features computed from the output of the Log-FB enhancement network. The correlation of Log-FB outputs with Log-FB 'ground truth' was not considered because the two types of enhancement are compared in the context of ASR using MFCC features. From the results displayed in Figure 5.4, it can be seen that BLSTM feature enhancement always improves over the noisy baseline. Furthermore, lower order MFCCs are predicted with slightly higher precision by the Log-FB enhancement network while the MFCC enhancement network is better at predicting higher order MFCCs. This is somewhat expected since the error function averages over frequency bands in the first case and over



(a) R^2 by coefficient across input SNRs



(b) R^2 by input SNR across MFCC 1–12 and energy

Figure 5.4: Evaluation of BLSTM feature de-noising: R^2 of enhanced and noisy MFCC features (1–12) and RMS energy (E) with noise-free MFCC features on the CHiME-2013 development set. Enh (MFCC): BLSTM output = enhanced MFCC; Enh (Log-FB): MFCCs generated from enhanced Log-FB features output by BLSTM.

MFCCs in the second case – thus, lower quefrencies are given more weight in the error calculation for the Log-FB enhancement network. However, lower order MFCCs seem to be easier to enhance than higher order MFCCs regardless of the actual type of features used in the network. Especially for high MFCCs at higher SNRs, a drop in performance is observed by Log-FB instead of direct MFCC enhancement (e.g., MFCC 12 at 9 dB SNR, Log-FB: $R^2 = .46$, MFCC: $R^2 = .51$). Conversely, e.g., enhancement of the MFCC 1 at -6 dB SNR works considerably better when

Table 5.3: ASR evaluation of BLSTM feature de-noising (MFCC, Log-FB). Word error rates (% WER) on the CHiME-2013 development set using the baseline GMM-HMM recognizer trained by EM-ML on the reverberated noise-free training set (si_tr_s). SSub: (log) spectral subtraction.

WER [%]	**SNR** [dB]						**Mean**
FE Domain	-6	-3	0	3	6	9	
—	86.25	82.79	76.08	71.35	63.04	55.87	72.56
MFCC	69.57	62.23	53.83	48.51	43.18	37.15	52.41
Log-FB	**62.59**	55.84	**47.80**	43.82	**38.25**	**34.68**	**47.16**
Log-FB + SSub	63.57	**55.38**	48.02	**43.76**	38.75	35.48	47.49

using Log-FB as features in the enhancement network (Log-FB: $R^2 = .73$, MFCC $R^2 = .67$). Overall, these results are quite promising since it is expected that higher performance on the lower order MFCCs achieved by Log-FB domain enhancement would result in ASR performance gains. This hypothesis will be verified below.

5.2.2.2 ASR performance

Let us begin the ASR evaluation of BLSTM enhanced features by considering BLSTM de-noising, i.e., learning mappings between noisy and noise-free features within the same acoustic environment. As acoustic models, the 'reverberated' CHiME baseline models [202] are used. Evaluation is done on the CHiME-2013 development set (test set results will be given below for selected systems). The resulting WERs are shown in Table 5.3. It can be seen that by enhancing the MFCCs directly, one obtains an improvement of 20 % absolute (28 % relative) in terms of WER. Using Log-FB outputs as net input and target, WER is further decreased by 5 % absolute (10 % relative), reaching 47.16 % average WER across the six SNRs. Using log spectral subtraction (SSub) on the filterbank outputs cannot further improve results. Thus, it seems that the mapping from noisy to clean features can best be learnt in the 'raw' log spectral domain.

Regarding the performance of BLSTM in comparison to simpler network architectures, i.e., bidirectional RNN and feedforward networks with input frame stacking, it is found that BLSTM significantly outperforms both BDRNN and DNN (Table 5.4). This corroborates earlier results with neural network based feature enhancement [236]. Comparing the number of parameters of the networks, it can be seen that the superiority of BLSTM is not simply due to increasing model complexity in terms of weights. In particular, the BLSTM network with 81, 96, and 81 units per layer performs almost equally to the larger network considered above, while DNNs with the same number of parameters perform significantly worse (58.48 % WER with

Table 5.4: ASR evaluation of alternative network topologies in Log-FB domain enhancement. DNNs using nine frames of input context to enhance center frame. # Wts: number of weights in network. Word error rates (% WER) on the CHiME-2013 development set using the baseline GMM-HMM recognizer trained by EM-ML on the reverberated noise-free training set.

WER [%]			**SNR** [dB]						**Mean**
Network	Layers	# Wts	-6	-3	0	3	6	9	
BLSTM-DRNN	81-96-81	305 k	63.36	55.22	48.71	44.28	38.05	34.39	**47.34**
BDRNN	108-128-108	159 k	76.44	69.68	60.90	58.49	52.32	47.62	60.91
DNN	256-256-256	270 k	74.78	68.27	59.79	54.91	49.54	43.60	58.48
DNN	384-384-384	503 k	76.26	69.68	60.96	56.99	50.07	45.79	59.96
DNN	486-256-108	395 k	76.37	68.86	60.88	55.93	49.82	46.87	59.79

the DNN with 3 × 256 units having 270 k weights, vs. 47.34 % with the BLSTM having 305 k weights). Further increasing the DNN size to 384 units per layer, or adjusting the hidden layer size to the size of the adjacent input and output layers (486-256-108 topology) does not improve performance. Generally, the fact that larger networks do not improve performance could be attributed to the limited amount of training data in the CHiME Challenge. Furthermore, it is observed that BRNNs perform slightly worse than DNNs with stacked inputs, pointing at the difficulty of training conventional RNNs through standard gradient descent. The fact that LSTM modeling outperforms feature frame stacking in DNNs is in accordance with the results reported by Wöllmer et al. [240] and recently by Geiger et al. [51] as well as Sak et al. [163] for neural network based ASR.

Next, in Table 5.5, the performance of BLSTM de-noised and de-reverberated features is considered in the close-talk recognizer provided by Vertanen [199]. The baseline WER of this recognizer applied to the CHiME development set is very high (89.43 % on average and 82.07 % even at 9 dB SNR). However, a drastic drop in WER occurs when applying feature enhancement in the MFCC domain (50.79 % WER, using the same network topology as above). Again, when using Log-FB outputs as enhancement domain, a further improvement down to 46.97 % WER is obtained – using the same network topology as for de-noising. When simply using a fourth layer, results are much worse (51.52 % WER), pointing at overfitting due to the increased number of parameters. When the above-mentioned deep training technique for mapping to clean features is used, 47.76 % WER are obtained, which is, however, below the result with simple training of a three-layer network. Switching to the zero mean log spectral domain, direct training of three- or four-layer networks does not reach the performance obtained with deep training. In the result, the lowest average WER that is attained with the unmodified close-talk recognizer is at 46.15 %, which is

Table 5.5: ASR evaluation of BLSTM feature de-noising and de-reverberation. Word error rates (% WER) on the CHiME-2013 development set using the baseline GMM-HMM recognizer trained by EM-ML on the close-talk microphone WSJ-0 training set (si_tr_s). SSub: (log) spectral subtraction. 3+1 layers: 4 layer network with pre-training of 3 layers (see text).

WER [%]	Layers	**SNR** [dB]						**Mean**
FE Domain		-6	-3	0	3	6	9	
Baseline (no enhancement)								
—	—	94.08	92.97	91.51	89.92	86.03	82.07	89.43
With BLSTM feature enhancement								
MFCC	3	70.10	61.16	52.34	47.66	38.97	34.51	50.79
Log-FB	3	65.34	57.58	48.18	41.78	36.5	32.44	46.97
Log-FB	4	69.97	62.49	52.71	47.28	41.23	35.41	51.52
Log-FB	3+1	66.53	59.27	48.6	43.08	36.49	32.57	47.76
Log-FB + SSub	3	65.92	58.53	48.34	42.19	36.65	31.36	47.17
Log-FB + SSub	4	65.48	57.47	47.62	42.49	35.97	31.70	46.79
Log-FB + SSub	3+1	**65.07**	**56.85**	**47.03**	**41.51**	**35.56**	**30.90**	**46.15**

a 48 % relative reduction with respect to using unenhanced features. In comparison, a four-layer network achieves 46.79 % average WER and a three-layer network 47.17 % WER. These rates are significantly worse (true average WER differences $\geq$ .45 and $\geq$.65 with 95 % confidence, according to a one-tailed t-test, treating WER per SNR as independent observations). Comparing the results to those obtained with the reverberated ASR models and BLSTM de-noising without de-reverberation, it is found that the latter works better at lower SNRs and performs worse at higher SNRs. This can be attributed to higher variances of the reverberated GMMs.

In the following, let us further investigate the relation between back-end refinement and front-end enhancement. The most obvious back-end adaptation is to consider multi-condition training using noisy data, as is done in the CHiME 'noisy' baseline acoustic models. As front-end enhancement, Log-FB de-noising is investigated – which yields best results with the reverberated models – and Log-FB de-noising and de-reverberation with log spectral subtraction – which gives best results with the clean models. Results are shown in Table 5.6.

Without any front-end enhancement, the CHiME multi-condition baseline yields an average WER of 58.27 % on the development set, improving by over 40 % absolute with respect to the clean models. With BLSTM de-noising, an additional improvement of 8 % absolute WER is observed. If the multi-condition models are re-trained using the BLSTM de-noised training set, average WER is decreased to 43.38 %. The gain by re-training is especially visible at higher SNRs. When using BLSTM de-noising

Table 5.6: ASR evaluation of BLSTM enhanced features in multi-condition trained models: EM-ML trained CHiME Challenge baseline models and discriminatively trained models using feature transformations (see text). Multi-condition training using unenhanced (= no re-training) and enhanced (= re-training) noisy and reverberated CHiME training set. Feature enhancement (FE) type: de-noising with Log-FB front-end (Table 5.5) or de-noising + de-reverberation with Log-FB + SSub front-end (Table 5.5). Evaluation on the CHiME-2013 development set.

WER [%]	**Retraining**	**SNR** [dB]						**Mean**
FE type		-6	-3	0	3	6	9	
EM-ML trained recognizer [202]								
—	—	73.17	67.43	59.89	55.71	49.07	44.34	58.27
de-noising	–	62.68	56.89	51.36	47.90	43.02	40.94	50.47
de-noising	√	**57.74**	**51.14**	**43.85**	39.10	35.54	32.88	**43.38**
+de-rev.	–	65.49	60.22	54.79	50.10	47.87	43.92	53.73
+de-rev.	√	62.34	53.39	45.17	**38.93**	**34.17**	**29.63**	43.94
EM-ML trained recognizer with feature transformations [192]								
—	—	59.63	49.97	40.60	34.72	29.56	25.52	40.00
de-noising	–	**46.35**	**37.94**	**31.20**	27.45	**23.60**	**21.12**	**31.28**
de-noising	√	49.45	41.08	33.18	29.31	25.14	22.05	33.37
+de-rev.	–	48.77	39.21	33.04	**27.26**	24.52	21.26	32.34
+de-rev.	√	55.60	46.79	38.69	31.60	27.57	22.78	37.17
Boosted MMI trained recognizer with feature transformations [192]								
—	—	56.47	47.12	38.47	31.86	27.50	23.32	37.46
de-noising	–	47.91	40.30	33.09	28.22	25.11	22.70	32.89
de-noising	√	**43.71**	**35.12**	**27.66**	24.90	21.55	18.56	**28.58**
+de-rev.	–	57.86	50.48	44.36	39.81	36.63	33.49	43.77
+de-rev.	√	47.31	37.65	30.15	**24.30**	**20.83**	**18.00**	29.71

and de-reverberation, additional improvements are obtained in the re-trained multi-condition models at higher SNRs ($\geq 3\,\text{dB}$), at the expense of reduced accuracy at lower SNRs. This is in line with the observations made above without noisy training.

The system proposed by [192] exploiting LDA, MLLT and SAT with fMLLR adaptation achieves better results without front-end enhancement than the best CHiME baseline system with front-end enhancement (40.00 % average WER)[1]. However, using BLSTM de-noising, another 8.7 % absolute accuracy improvement (31.28 % WER) is gained 'on top'. Interestingly, results are found to be best in a 'plug-and-play' setup where the feature transformations are estimated on noisy data instead of enhanced

[1]Note that this result is much better (6 % absolute WER difference) than the corresponding result reported by [192], because for a fair comparison beam-forming is used as in the CHiME baseline.

data – thus, there seems to be a larger mismatch in the enhanced features than in the noisy features across training and development set. This could be due to the networks being trained speaker-independently – in the future, enhancement on the SAT transformed features could be investigated.

Finally, it is observed that BLSTM feature enhancement is also complementary to discriminative training using boosted MMI. Boosted MMI and feature transformations without front-end enhancement yield 37.46 % WER, while the best combination (BLSTM de-noising, feature transformations, boosted MMI training using enhanced noisy data) results in 28.58 % average WER on the development set. Notably, when using boosted MMI training using unenhanced data and evaluating using de-noised and de-reverberated data, results are vastly degraded (43.77 % WER) while reasonable results are obtained with re-training (29.71 % WER) – this probably indicates in a large mismatch of the phoneme errors on unenhanced and enhanced data, leading to over-fitting.

Since fMLLR adaptation in the form used by the system of Tachioka et al. [192] requires the utterances of each speaker to be processed at once, it is not suitable for real-time applications such as dialog systems; it is thus of interest to also consider results without adaptation (and hence without SAT). In this case, best performances (not shown in Table 5.6) are obtained using the de-noising (not de-reverberation) front-end, with recognizer re-training, leading to 35.87 % (instead of 33.37 %) average WER in the EM-ML and 30.82 % (instead of 28.58 %) WER in the discriminatively trained recognizer (without de-noising and without SAT: 46.07 %, 45.13 %).

For the results reported so far (Tables 5.3, 5.5, and 5.6), a constant language model weight (LMW, cf. (3.126)) of 15 has been used for a fair comparison of results. However, it was found that since 'cleaner' features yielded generally higher acoustic likelihoods, the language model weight should be increased accordingly. An optimal weight $\mu^* \in \{9, 10, 11, \ldots, 20\}$ was determined on the development set. Results are displayed in Figure 5.5. It can clearly be seen that the 'cleaner' the features, the higher the language model weight has to be for optimal performance. In the boosted MMI system using feature transformations, $\mu^* = 11$ yields 34.18 % WER without BLSTM feature enhancement; 28.10 % WER are obtained at $\mu^* = 17$ with BLSTM de-noising; for BLSTM de-noising and de-reverberation, $\mu^* = 20$, resulting in 28.66 % WER. For comparison, let us note that the best DNN in the best back-end (LDA-MLLT, SAT, fMLLR adaptation, boosted MMI) achieved 33.22 % WER, which is significantly (more than 4 % absolute) better than the noisy baseline but clearly below the BLSTM result.

Let us now proceed to evaluate selected ASR systems (combinations of back-ends and BLSTM front-ends) on the official CHiME-2013 Challenge test set, and compare to other state-of-the-art approaches. Results are shown in Table 5.7. The best system without back-end modification (using close-talk acoustic models) yields 42.06 % average WER across SNRs from -6 to 9 dB. This is much better than the result using noise compensation only in the back-end by MCT, in the same HMM

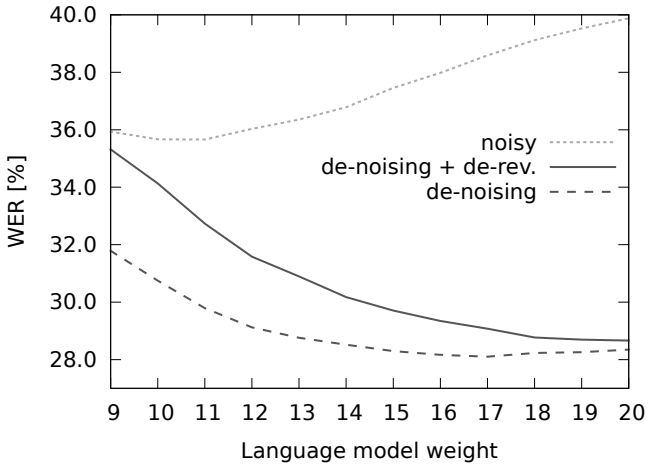


Figure 5.5: Influence of the language model weight on the WER on the CHiME-2013 development set for noisy features, BLSTM de-noising and de-reverberation.

framework (55.01 %, [202]). It also outperforms a state-of-the-art approach for feature enhancement in the linear Mel frequency domain using non-negative matrix factorization (NMF) [52], which gives 48.07 % WER.

Combining BLSTM feature enhancement and MCT results in 39.24 % WER, which is a noticeable improvement but also indicates the limits of the basic HMM recognizer framework. Still, this result is better than a previous result with multi-stream HMM fusion of MCT EM-ML trained MFCC-GMMs and a BLSTM phone recognizer ([52], 41.76 % WER). In this work, a deep BLSTM was used as a secondary acoustic model providing frame-wise phoneme probabilities, instead of performing front-end enhancement.

Using the BLSTM front-end, but changing the back-end to a state-of-the-art system exploiting feature transformations and discriminative training [192], 22.16 and 22.78 % WER are obtained in combination with BLSTM de-noising and de-noising / de-reverberation. This is a 17 % relative improvement over the BLSTM feature enhancement results presented in the CHiME-2013 Challenge (26.73 % WER, cf. [225]).

The BLSTM result also outperforms DNN feature enhancement by 4 % absolute; in turn, DNN enhancement in the front-end seems to perform slightly better than binary masking [192]. Comparing to recent results by Geiger et al. [53], the results by BLSTM feature enhancement are slightly inferior to BLSTM acoustic modeling in a multi-stream HMM framework (21.5 % WER). Still, the best result reported here is better than the one by Weng et al. [209] for standard RNN acoustic modeling (22.8 % WER).

Table 5.7: Final CHiME-2013 test set evaluation of ASR systems with BLSTM feature enhancement and comparison to related approaches. [1]: Only three significant digits given in [53].

WER [%]						
SNR [dB]						**Mean**
-6	-3	0	3	6	9	
Systems using BLSTM feature enhancement						
BLSTM de-noising + de-reverberation / base WSJ-0						
61.55	50.64	43.84	37.04	31.94	27.33	42.06
BLSTM de-noising / CHiME multi-condition baseline						
53.18	44.97	40.65	34.32	32.39	29.93	39.24
BLSTM de-noising / feat. transf. + MMI						
35.55	27.11	22.40	17.45	16.14	14.29	**22.16**
BLSTM de-noising + de-rev. / feat. transf. + MMI						
37.79	28.71	23.37	18.70	15.41	12.70	22.78
Other systems for CHiME 2013 track 2 task						
CHiME multi-condition baseline [202]						
70.43	63.09	58.42	51.06	45.32	41.73	55.01
NMF / CHiME multi-condition baseline [52]						
61.85	55.58	50.94	43.51	39.14	37.40	48.07
Binary masking / feat. transf. + MMI [192]						
44.12	35.46	28.12	21.20	17.43	14.83	26.86
DNN feature enhancement / feat. transf. + MMI						
42.11	33.08	26.17	21.43	18.08	16.05	26.15
BLSTM-HMM double-stream recognizer[1] *[53]*						
37.1	27.2	22.5	16.7	13.9	11.8	**21.5**

5.2.3 Conclusions

The efficacy of data-based feature enhancement using deep recurrent neural networks for ASR in non-stationary convolutive noise has been demonstrated. Reasonable results have been achieved even with unmodified close-talk acoustic models, which otherwise fail at decoding the CHiME utterances. Best results on the CHiME-2013 task have been achieved by combining enhancement with feature transformations and discriminative HMM training. Furthermore, the enhancement of ASR features by the proposed method has been shown to be complementary to that kind of state-of-the-art multi-condition training. The improvements by the proposed BLSTM feature enhancement method are all the more noticeable since it does not directly exploit phonetic information. However, this absence of phonetic information could contribute to the observed inferior performance in comparison with BLSTM acoustic

modeling. Thus, effective combination strategies with BLSTM acoustic modeling will be an important topic of future research.

Another caveat is that in contrast to other enhancement techniques such as factorial models [159, 222], the present approach requires frame-by-frame correspondences between distorted and clean training features, similar to the supervised training of speech enhancement presented in the previous sections. Hence, the most straightforward approach to generate training data is to algorithmically apply distortions to clean data, as done in the CHiME Challenges and previous evaluations such as the AURORA-4 database. Still, realistic training data is not trivial to obtain (it could be done, e.g., by loudspeaker playback and recording in various settings involving real noise and reverberation). A more promising approach might be to use semi-supervised learning, initialized by large amounts of systematically generated training data using combinations of speech and noise corpora, and continuing using real noisy and reverberated speech for which no 'clean' counterpart exists.

One important issue in training-based methods such as deep learning, as compared to 'blind' de-noising and de-reverberation approaches, is generalization to unseen test scenarios. In Section 6.1, an approach will be presented that exploits both blind and training-based approaches for ASR feature enhancement, and in Section 6.2 a combination of microphone array processing and BLSTM feature enhancement will be exploited. Including noise context, i.e., training on noisy streams instead of end-pointed but corrupted speech data, might also help generalization – there is already evidence that LSTM networks are very well suited to voice activity detection in noise [44].

5.3 From feature to speech enhancement

Let us now conclude the discussion of speech enhancement and ASR feature de-noising by uniting the two in a single framework.

Again, time-frequency masking is considered as the general framework to recover a source l of interest from mixture signal features $\mathbf{m}_t$. Explicit modeling of both the source l and the interference $\bar{l}$ is assumed: Spectral representations $\widetilde{\mathbf{s}}_t^l$ and $\widetilde{\mathbf{s}}_t^{\bar{l}}$ are computed from the mixture. Then, the source estimate $\hat{\mathbf{s}}^l$ is computed through

$$\hat{\mathbf{s}}_t^l = \frac{\widetilde{\mathbf{s}}_t^l}{\widetilde{\mathbf{s}}_t^l + \widetilde{\mathbf{s}}_t^{\bar{l}}} \otimes \mathbf{m}_t. \tag{5.6}$$

For example, traditional spectral subtraction can be cast into this framework by unsupervised estimation of $\widetilde{\mathbf{s}}_t^{\bar{l}}$ (e.g., by minimum statistics [124]), then setting

$$\widetilde{\mathbf{s}}_t^l = \mathbf{m}_t - \widetilde{\mathbf{s}}_t^{\bar{l}}. \tag{5.7}$$

Of course, $\widetilde{\mathbf{s}}_t^l$ and $\widetilde{\mathbf{s}}^{\bar{l}_t}$ can be estimated as non-negative combinations of spectral dictionary atoms, resulting in the NMF filter equation (3.26). Alternatively, the

feature enhancement approach from the previous section, which outputs a speech feature estimate given a mixture feature estimate, can be used. Given a trained feature enhancement regressor $h_{\mathbf{W}^l}$,

$$\widetilde{\mathbf{s}}_t^l = \overline{g}(h_{\mathbf{W}^l}(g(\mathbf{m}_t)) \tag{5.8}$$

is obtained, where $\overline{g}$ is the 'inverse' feature extraction procedure. Furthermore, the feature 'enhancement' approach can be used reversely, in order to obtain an interference feature estimate from the mixture feature estimate:

$$\widetilde{\mathbf{s}}_t^{\bar{l}} = \overline{g}(h_{\mathbf{W}^{\bar{l}}}(g(\mathbf{m}_t)). \tag{5.9}$$

Note that in the general case the feature extraction is not invertible. However, if Log-FB features are used as in the previous section, the logarithmic transform can simply be reverted by exponentiation, and a Mel-domain mask is computed according to (5.6) using $\overline{g}(x) = \exp(x)$. Then, (3.61) is used to obtain full-resolution spectral estimates of the source. It is also possible to run a 'hybrid' approach where the interference is estimated in the full-resolution DFT domain using an unsupervised method, and

$$\widetilde{\mathbf{s}}_t^l = \widetilde{\mathbf{B}}^\intercal \exp(h_{\mathbf{W}^l}(g(\mathbf{m}_t)), \tag{5.10}$$

where $\widetilde{\mathbf{B}}^\intercal$ is the transpose of the Mel matrix $\mathbf{B}$ where filterbank coefficients are normalized to sum to unity. This is similar to the proposal of Lu et al. [119] uniting a DNN approach for speech feature estimation with an unsupervised noise estimation approach. These setups will be investigated below.

In this section, DNN- and DRNN-based models are chosen as feature estimator(s) h, in analogy to the speech enhancement experiments reported in Section 4.2. A focus is laid on real-time capability, and hence, unidirectional LSTM-DRNNs; yet, performance is also compared with bidirectional (BLSTM)-DRNNs as were used in Section 5.2.

5.3.1 Experiments

Experiments are carried out on the NAVIC database of conversational speech in multiple noise and reverberation conditions (cf. Section 2.2.1.2.1). The task considered is speech enhancement, with the only wanted source s^l corresponding to speech. A multi-condition training setup is used where no knowledge of RIR, SNR, or noise type is assumed during enhancement.

5.3.1.1 Network training and evaluation

As input features for the neural networks, logarithmic Mel scale spectrograms $\mathbf{M} \in \mathbb{R}_+^{B \times T}$ with $B = 40$ frequency bands equally spaced on the Mel frequency scale are used. The scope of the evaluation is constrained to the de-noising, not the

de-reverberation task – that is, the output of the networks will comprise convolutive noise. Independent networks with three hidden layers are used to predict either speech or noise features. Both DRNNs and DNNs are considered for speech feature estimation. DNNs and LSTM-DRNNs have 256 units per layer while BLSTM-DRNNs have 128 units per direction. Sub-sampling layers with 64 units are inserted after each LSTM layer. Networks are trained on the NAVIC training instances with SNR $< \infty$; for speech feature prediction, the corresponding reverberated, yet noise-free features are used as training targets (SNR $= +\infty$). An analogous procedure is followed for noise feature prediction. To prevent over-fitting at high SNRs in training, Gaussian noise with zero mean and standard deviation 0.1 is added to the inputs. Input and target features are standardized to zero mean and unit variance on the training set, and delta regression coefficients (3.138) of the feature contours are added. To further alleviate over-fitting, early stopping on a held-out validation set as well as random shuffling of training sequences are used. The cost function at the output layer is the squared Euclidean distance (3.41).

In all experiments, DNNs and LSTM-DRNNs are trained and evaluated on GPUs using the CURRENNT software (cf. Section 3.5.5). Mini-batch gradient descent with batch size $|\mathcal{B}| = 15$ is used in the experiments. One BLSTM-DRNN training epoch on the 32 k sequences, 5.8 M time steps AVIC training set (cf. above) takes around 20 minutes on a consumer grade GPU. Training a DNN for an epoch takes only 50 seconds due to an increased level of parallelization across timesteps (using 50 parallel sequences). Depending on the task to learn, networks took around 35–100 epochs to converge. Except for the number of units and the DNN learning rate (reduced to 10^{-6} to ensure convergence), all chosen hyper-parameters (such as learning rate) correspond to the regression example delivered with CURRENNT for straightfoward reproducibility, which is based on experiments with the CHiME-2013 feature enhancement task (cf. above and [225]). Decoding one of the 60 test sets (44 minutes of speech) in batch processing takes less than a minute on a consumer grade GPU.

5.3.1.2 Source separation evaluation

After resynthesizing time-domain signals from the filtered magnitude spectra $\hat{\mathbf{s}}_t^l$ (5.6) by means of windowing and overlap-add, using the mixture phase, SDR, SIR, and SAR are computed (cf. Section 2.1.1.2 and [200]). As baselines, unprocessed noisy signals are considered. Furthermore, different combinations of minimum statistics (unsupervised) and neural network based speech and noise estimates are evaluated. As implementation of minimum statistics spectral subtraction, the freely available Voicebox toolkit for MATLAB is used[2]. The default parameters in the toolkit are changed by setting the sliding window length for minimum statistics estimation

[2] `http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html`

Table 5.8: Noisy AVIC (NAVIC) corpus: Evaluation of speech enhancement by unsupervised spectral subtraction using minimum statistics (MinStat) or supervised training of noise estimators ((B)LSTM-RNN); clean speech estimation using supervised training of DNN, LSTM-RNN or BLSTM-RNN.

model		[dB]		
speech	noise	SDR	SIR	SAR
Noisy baseline (no processing)				
—		13.2	13.2	∞
On-line Enhancement				
–	MinStat	8.3	**17.1**	10.3
DNN	MinStat	11.4	15.4	16.2
LSTM-RNN	MinStat	12.1	15.7	16.4
LSTM-RNN	LSTM-RNN	**14.6**	17.0	**19.7**
Off-line Enhancement				
BLSTM-RNN	BLSTM-RNN	14.8	16.6	20.8

to 0.256 s (16 windows), and disabling over-subtraction by setting the maximum subtraction factor to 1, which led to a few dB SDR gain in a preliminary experiment.

5.3.2 Results

Table 5.8 shows the average SDR, SIR, and SAR obtained on the NAVIC test set, across all acoustic conditions, input SNRs (0 to 20 dB), and noise types. Using minimum statistics spectral subtraction, about 4 dB absolute in interference reduction (SIR) are gained at the cost of artifacts, which lower the SDR by almost 5 dB absolute with respect to the noisy baseline. Using a DNN or LSTM-RNN based speech estimate along with minimum statistics noise estimation significantly increases the SDR (by 3 and 4 dB absolute) due to an increase in SAR by about 6 dB absolute, with respect to the minimum statistics baseline. However, around 1.5 dB absolute are lost in interference reduction. Using a LSTM-RNN based noise estimate in addition further boosts the SDR to 14.6 dB and SAR to 19.7 dB while providing similar interference reduction (SIR = 17 dB) as minimum statistics spectral subtraction. Considering bidirectional LSTM-RNNs, a slight improvement in SDR (+0.2 dB) can be gained at the expense of real-time capability.

Figure 5.6 shows the results by input SNR in more detail. It can be seen that the SDR of the minimum statistics spectral subtraction saturates at around 11 dB – which is due to the introduction of artifacts, i.e., lower SAR – while SIR increases consistently with input SNR. However, at low SNRs (0 and 5 dB), LSTM-RNNs outperform minimum statistics also in terms of SIR.

Finally, in Figure 5.7 two examples of speech corrupted by city noise (clicking

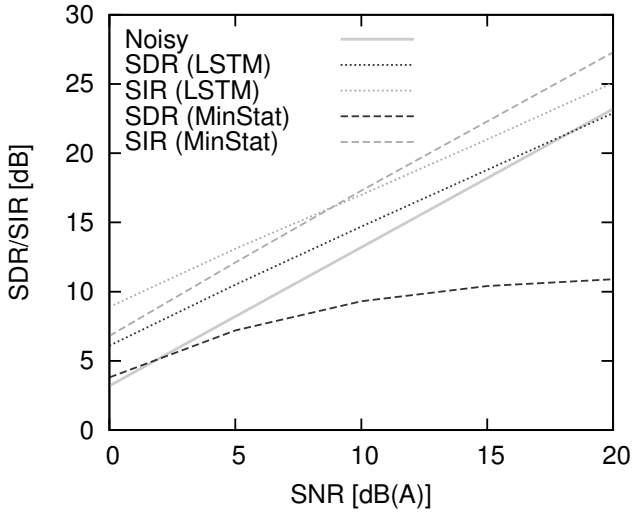


Figure 5.6: NAVIC corpus: SDR and SIR by input SNR, averaged across acoustic conditions; LSTM-RNN speech/noise model or minimum statistics (MinStat) noise model.

noise caused by a bicycle) at high SNR (20 dB), as well speech corrupted by music noise (rock music with distorted guitars and drums, SNR = 5 dB) are shown. In the former case, the spectro-temporal structure of the original speech is very well reconstructed by the LSTM-RNN approach while most of the broadband transient interference is reduced. Minimum statistics does not remove all of the interference while partially 'destroying' speech components, resulting in some musical noise. The bottom row shows that also music noise can be compensated by the LSTM-RNN approach to some degree; while some harmonic interferences from the music remain in the lower frequency bands, there is significantly less musical noise than with minimum statistics.

Informal listening tests confirm that the LSTM speech enhancement approach presented in this section produces naturally sounding speech, and remaining interferences also sound natural.

In order to compare to the results obtained by time-frequency mask training, an additional experiment was performed using the mask approximation strategy from Section 4.2. Accordingly, the hyper-parameter selection was preserved from the experiments on the CHiME-2013 development set: $B = 100$ Mel bands and $\alpha = 1$ were used, and a LSTM-DRNN with two hidden layers with 256 units each. This configuration achieved an SDR of 14.6 dB on the NAVIC test set, which is slightly below the performance reported above (14.8 dB). From this result, it can be concluded that the speech enhancement approach derived from feature enhancement is highly effective. However, it has to be noted that the network for the mask approxmation has only 918 K parameters, while in the experiments above, two networks with 1.06 M

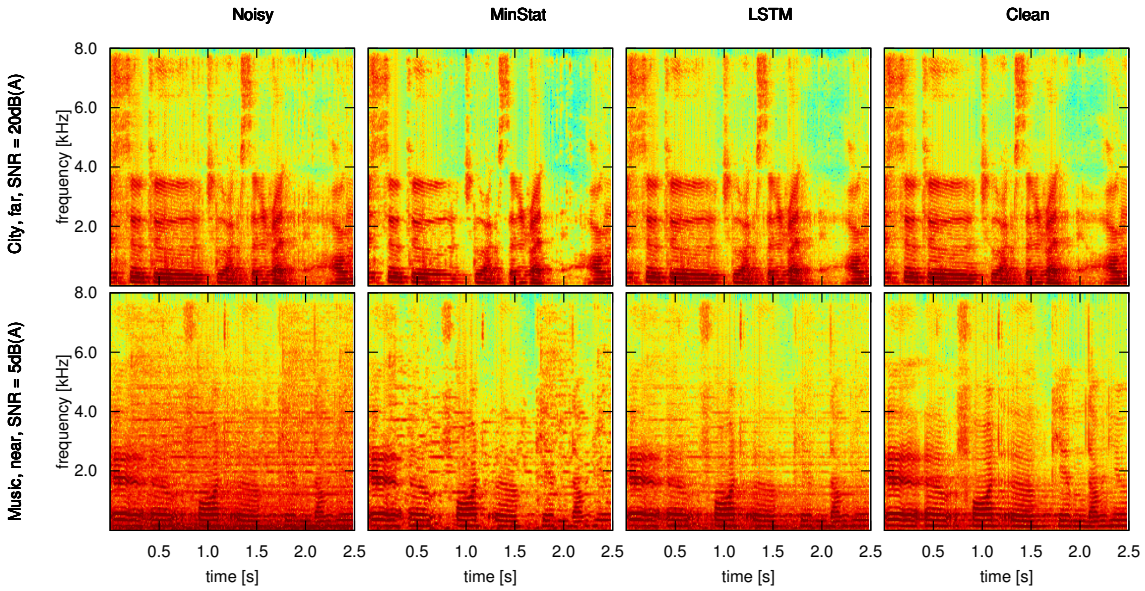


Figure 5.7: De-noising examples: Test utterances (top: #4, bottom: #8) from the NAVIC corpus, after processing with spectral subtraction using a minimum statistics (MinStat) noise estimate, or LSTM-RNN speech and noise estimates, and the noise-free ('clean') version.

weights each are used, i.e., 2.12 M trainable parameters in total. It can be conjectured that the mask approximation formulation yields a very efficient representation of the relation of speech and noise features, compared to the 'double' feature enhancement approach. However, this will have to be verified in future experiments using various network topologies, such as multi-task networks that estimate speech and noise features simultaneously.

There is also evidence that although the LSTMs are trained on simulated parallel data, the approach generalizes to real-world recordings. An initial experiment was conducted with mobile phone recordings, using a Huawei P2 phone (with disabled speech enhancement) in a busy street scene near TUM. An example recording (original and enhanced by LSTM) is shown in Figure 5.8, where a male speaker (25 years old) reads a sentence from a scientific paper. It can be seen in the spectrograms that despite the poor quality of the recording as such, the noise can be removed successfully. Particularly notable is the bell sound around the 8 s position, caused by a bike driving by, which is attenuated by the LSTM filtering despite its transient nature. These claims can be verified by the interested reader by listening to the example, which is provided at the above mentioned location.

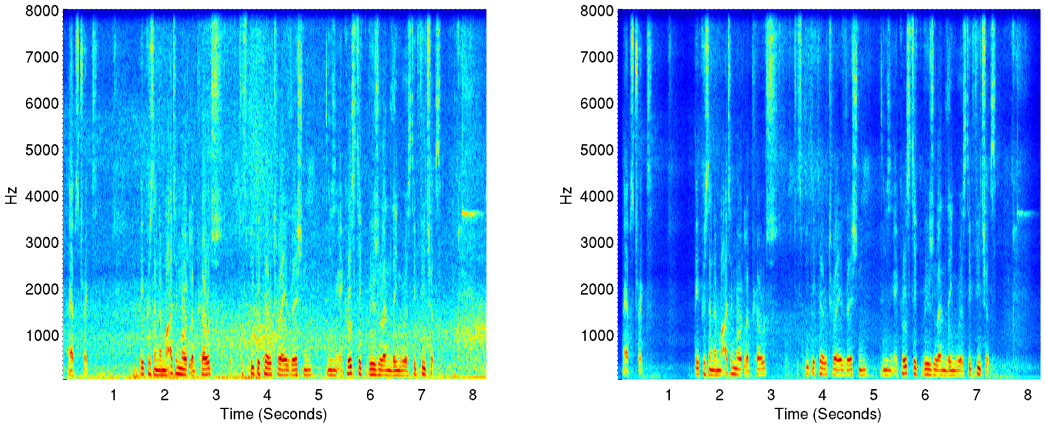


Figure 5.8: Application of LSTM speech enhancement trained on NAVIC corpus to real mobile phone recording in street noise: noisy (left) and processed (right).

5.3.3 Conclusions

In this section, a generic framework for time-frequency masking was derived from the previously presented ASR feature enhancement approach, and was shown to be highly effective for conversational speech enhancement in non-stationary noise and various acoustic conditions. Applying supervised training of LSTM-DRNN feature enhancement, unsupervised speech enhancement was outperformed by a large margin in terms of SDR, and performance was found to be similar to a dedicated speech enhancement approach. The proposed method introduces very little artifacts while providing good interference reduction, and seems to generalize to real-world recordings. In the future, the performance of the approach might be further improved by considering negative SNRs in training, i.e., data where noise is dominant, to ease the training of noise feature estimators. Furthermore, the subjective quality of all the speech enhancement algorithms presented in this and the previous chapter could be enhanced by applying unsupervised techniques for musical noise reduction such as the one proposed by Esch and Vary [39].

6

Feature de-reverberation for automatic speech recognition

When there is an original sound in the world, it makes a hundred echoes.
– John A. Shedd

After focusing on the removal of non-stationary additive noise in the previous section, let us now shift the attention to convolutive noise, i.e., reflected sounds caused by reverberant environments. Conversely, in this chapter, additive noise will be limited to stationary ambient noise. This scenario corresponds to relevant applications such as meeting transcription [116].

6.1 Early fusion in feature enhancement

So far, robustness of ASR in this thesis has been realized by data-based methods such as multi-condition training, acoustic modeling techniques such as DNN, supervised training of signal and feature enhancement, and combinations of these. In the context of reverberation, Ishii et al. [93] were among the first to study DNN-based feature enhancement for de-reverberation.

However, a problem with such data-based approaches is generalization to acoustic environments which are not known at training time. Traditionally, supposing a physically motivated model of reverberation, 'blind' or 'model-based' techniques can be used to estimate physical parameters of the room acoustics, such as reverberation time [158], or to compensate the influence of the transfer function of the room on ASR features [54, 103, 179, 180]. A survey on these techniques is given by Habets [72]. Furthermore, model-based ASR adaptation techniques (cf. Section 3.7.1.4) allow to blindly estimate transformations of the ASR features suited to the current acoustic environment [49, 153]. They can also account for speech modifications by de-reverberation [28].

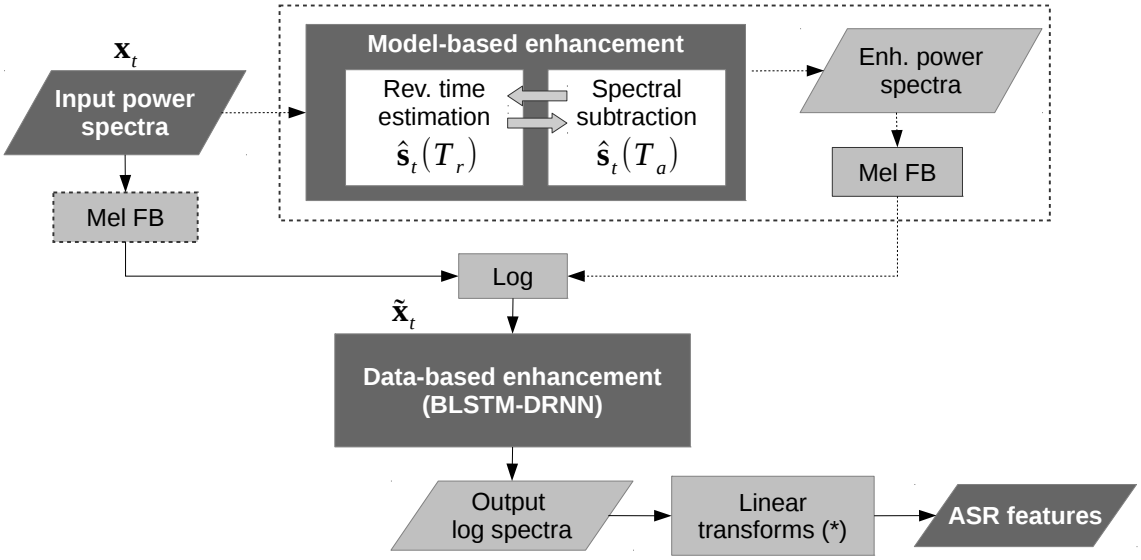


Figure 6.1: Flowchart of the proposed method for reverberated speech recognition. Dashed lines depict optional processing steps. FB: filterbank. (*) Linear transformations: DCT (to obtain MFCC), LDA, MLLT, CMLLR – see text.

The research objectives in this study, which was presented by the author and his colleagues in [232], are three-fold. The first goal is to show that supervised training of feature enhancement in analogy to the previous section generalizes to real reverberated speech in unseen acoustic conditions. Second, it is proposed to perform early fusion of unprocessed spectral features with spectral features obtained from a model-based de-reverberation algorithm, and it is shown that this further improves performance, in contrast to naïve cascading. Third, it is demonstrated that blind ASR adaptation provides complementary performance gains to all these system combinations – that is, the proposed combined de-reverberation improves over state-of-the-art ASR techniques.

As in the previous section, feature enhancement is done by LSTM-DRNNs which provides a flexible amount of temporal context to the network, as is required for de-reverberation in multiple acoustic environments. The method is evaluated on the 2014 REVERB Challenge data (cf. Section 2.2.1.3) which features both simulated reverberated and noisy data as well as real recordings from a meeting room.

6.1.1 Experiments

The proposed feature enhancement algorithm with early fusion is depicted as a flowchart in Figure 6.1. Below, the steps in the algorithm will be explained in more detail.

6.1.1.1 Spectral subtraction with RT estimation

As a physical model based single-channel de-reverberation method, the algorithm proposed by Tachioka et al. [193] is employed, which performs spectral subtraction based de-reverberation with reverberation time (RT) estimation. If the RT is much longer than the frame size, the band energies of the reflected and direct sounds can be simply superposed. Therefore, an observed power spectrum $\mathbf{x}_t$ is modeled as a weighted sum of the (dry) speech power spectrum $\mathbf{s}_t$ to be estimated and a stationary noise spectrum $\mathbf{n}$ as

$$\mathbf{x}_t = \sum_{j=0}^{t} h_j \mathbf{s}_{t-j} + \mathbf{n}, \tag{6.1}$$

where h_j are time-varying weights for delays j. $\mathbf{n}$ can be estimated by averaging $\mathbf{x}_t$ for a small number of frames where speech is absent. The power spectrum of the dry speech source can be estimated from the observed power spectra as

$$\hat{\mathbf{s}}_{t-j} = \zeta(T_r)\mathbf{x}_{t-j} - \mathbf{n}, \tag{6.2}$$

where T_r is the RT in the given environment and ζ is the average proportion of direct sounds in the frequency bins. ζ is a decreasing function of T_r because the amount of reflected sounds increases with longer T_r. Assuming that h_0 is unity, and given an RT estimate T_a, a clean speech estimate $\hat{\mathbf{s}}_t(T_a)$ is derived from (6.1) and (6.2):

$$\hat{\mathbf{s}}_t(T_a) = \mathbf{x}_t - \sum_{j=1}^{t} h_j(T_a) \left[\zeta(T_a)\mathbf{x}_{t-j} - \mathbf{n}\right] - \mathbf{n}. \tag{6.3}$$

To estimate the weights h_j, a two-stage reverberation model is assumed. Early reverberation is complicated, but it is ignored here, because it only influences short-term spectra, which can be compensated by the acoustic model. On the contrary, the ASR performance is mainly degraded due to late reverberation. There, the sound energy density decays exponentially with time according to Polack's statistical model [72]. The threshold between early and late reverberation shall be denoted by ξ. Hence, the weights h_j are determined as

$$h_j(T_a) = \begin{cases} 0 & (1 \le j \le \xi)\,, \\ \gamma \exp(-\frac{6 \ln 10}{T_a} \Delta\tau j) & (\xi < j)\,, \end{cases} \tag{6.4}$$

where $\Delta\tau$ is the frame shift and γ is a subtraction parameter to be set. With (6.4) and assuming constant $\zeta(T_a)$, the result of (6.3) is similar to spectral subtraction [16]. If the estimated speech power spectrum $\hat{\mathbf{s}}$ is less than $\epsilon\mathbf{x}$, it is substituted by $\epsilon\mathbf{x}$, where ϵ is a flooring constant. The flooring probability ϕ is defined as the ratio of the number of floored DFT bins in $\epsilon\mathbf{x}$ to the number of total DFT bins.

Two observations are exploited to estimate T_r from flooring probabilities ϕ. First, when varying the RT estimate T_a in (6.3), ϕ increases monotonically with T_a for

constant ζ, because the weights h_j increase with T_a. This is modeled as a linear relationship with inclination Δ_ϕ. Second, ϕ increases with T_r, since the assumption of constant ζ leads to an overestimation of the subtrahend in (6.3) for longer T_r, and hence, oversubtraction. Therefore, T_r has a linear relationship with Δ_ϕ, which can be modeled as

$$T_r = a\Delta_\phi - b \tag{6.5}$$

with two empirically detemined constants a and b. Thus, to compute T_r, first the ratio $\phi(T_a)$ is calculated for various T_a in steps of 0.05 s. From this, Δ_ϕ is obtained by least-squares regression, and T_r and $\hat{\mathbf{s}}(T_r)$ are computed according to (6.5) and (6.3).

6.1.1.2 BLSTM-DRNN feature enhancement

In addition to blind de-reverberation, spectral enhancement is performed based on supervised DNN training. To highlight the connection of this feature enhancement method and the one proposed earlier by Ishii et al. [93], the term 'de-noising auto-encoder' (DAE) is used interchangably with DRNN. To model the context needed for compensating late reverberation, LSTM-DRNNs are the method of choice, which deliver state-of-the-art performance in ASR also in real reverberated and noisy speech [241]. As input features and training targets, Log-FB features with 24 Mel frequency bands are used. In addition, log power spectra, for comparison with blind de-reverberation on similar features, are considered. In the following, the network input will be denoted by $\tilde{\mathbf{x}}_t$.

Delta coefficients are added to the filterbank features to capture dynamics at the feature level, which gives a slight performance gain. Since the level of the RealData utterances was found to be very low (below -30 dB) in general, these were amplified to a peak level of -24 dB. This is important because the features $\tilde{\mathbf{x}}_t$ are scale-sensitive.

Since feature enhancement is investigated in combination with utterance-based ASR adaptation in the present study, future context within a sequence can be exploited. Here, this is done by considering BLSTM-DRNNs.

6.1.1.3 Integration

To integrate spectral subtraction and RT estimation with the BLSTM-DRNN feature enhancement, one can simply use the output of the former as input for the latter, i.e., $\tilde{\mathbf{x}}_t = \log(\mathbf{B}\hat{\mathbf{s}}_t(T_r))$, where $\mathbf{B}$ is the Mel matrix. Alternatively, one can use early (feature level) fusion of unprocessed and de-reverberated speech, i.e., $\tilde{\mathbf{x}}_t = \log[\mathbf{B}\mathbf{x}_t; \mathbf{B}\hat{\mathbf{s}}_t(T_r)]$. This is similar in spirit to the proposal by Seltzer et al. [183] to use a model-based estimate of additive stationary noise in noise-robust neural network training; however, here the noise is of convolutive nature. By providing de-reverberated features, an initial solution for the output features is given to the network, and by having access to the original features, the network can potentially

compensate some distortions by the blind enhancement algorithm. Finally, data-based fusion of $\hat{\mathbf{s}}_t(T_a)$ for different T_a is investigated, instead of using a heuristic to compute T_r and $\hat{\mathbf{s}}_t(T_r)$ from multiple $\hat{\mathbf{s}}_t(T_a)$. In particular, the input features $\tilde{\mathbf{x}}_t = \log[\mathbf{B}\hat{\mathbf{s}}_t(0.3\,s); \mathbf{B}\hat{\mathbf{s}}_t(0.5\,s); \mathbf{B}\hat{\mathbf{s}}_t(0.7\,s); \mathbf{B}\mathbf{x}_t]$ are used.

Note that enhanced ASR features are directly generated from the BLSTM-DRNN outputs by applying the DCT (to obtain MFCCs) and other linear transforms (cf. below).

6.1.1.4 ASR baseline

The ASR baseline used for the present study is based on the Kaldi speech recognition toolkit [152] and is an improved version of the ASR baseline provided by the REVERB Challenge organizers, which is implemented with HTK [245]. A clean triphone recognizer is trained on the WSJCAM0 training set, while a multi-condition triphone recognizer is trained by repeating the HMM training steps using the REVERB multi-condition training set. The standard 5 k WSJ language model is used, whose weight is set to 15.

In this paper, two major improvements are implemented compared to the HTK baseline. These are motivated by their success on the noise-robust ASR task in the CHiME Challenge (cf. Section 5.2). First, while the HTK baseline employs standard MFCCs plus delta coefficients, here LDA (cf. Section 3.7.1.1) is used with context size $T_L = T_R = 4$, keeping the 40 first components. During model training on LDA features, after every other iteration (2–10) STC matrices are estimated (cf. Section 3.7.1.2). In case of MCT, LDA and STC matrices are estimated on the REVERB multi-condition training data.

Second, while the HTK baseline performs adaptation by fMLLR on all test utterances of a specific test condition, here basis fMLLR is used for robust per-utterance adaptation [153]. The bases are estimated on the training set of each recognizer (clean, multi-condition). On the SimData and RealData sets, the WERs of the proposed ASR baseline are 19.42 % and 41.4 % (HTK multi-condition + CMLLR baseline: 25.16 and 47.2 %). For reference, let us note that the WER of the clean recognizer on the clean WSJCAM0 development set is 8.26 % (HTK clean baseline: 10.94 %). To perform ASR on pre-processed data (by de-reverberation methods), the clean and MCT recognizers are evaluated with and without adaptation to the processed data, as well as re-trained recognizers obtained by performing the MCT step (including the estimation of the LDA and MLLT transforms and the CMLLR basis) with the pre-processed multi-condition training set.

6.1.1.5 De-reverberation parameterization

For both de-reverberation methods, short-time spectra of 25 ms frames at 10 ms frame shift are extracted. For the blind de-reverberation method, parameters are set

Table 6.1: REVERB development set results obtained with a clean recognizer and BLSTM-DRNN feature enhancement trained with different input features and training targets. Significant digits reflect the different sizes of the SimData and RealData sets. $\mathbf{x}_t$ denotes a short-term power spectrum.

WER [%] Input	Target	SimData	RealData
Power spectral domain enhancement			
$\log \mathbf{x}_t$	$\log \mathbf{s}_t$	24.99	75.4
Mel filterbank domain enhancement			
$\log \mathbf{B}\mathbf{x}_t$	$\log \mathbf{B}\mathbf{s}_t$	21.22	56.5
$\log \mathbf{B}\hat{\mathbf{s}}_t(T_r)$	$\log \mathbf{B}\mathbf{s}_t$	22.97	56.9
Mel filterbank domain enhancement; feature level fusion			
$\log[\mathbf{B}\mathbf{x}_t; \mathbf{B}\hat{\mathbf{s}}_t(T_r)]$	$\log \mathbf{B}\mathbf{s}_t$	20.02	61.8
$\log[\mathbf{B}\mathbf{x}_t; \mathbf{B}\hat{\mathbf{s}}_t(T_a)]$ $T_a \in \{0.3, 0.5, 0.7\}$s	$\log \mathbf{B}\mathbf{s}_t$	**19.06**	**52.5**

as follows: $\xi = 9$, $\gamma/\zeta = 5$, $\epsilon = 0.05$, $a = 0.005$, and $b = 0.6$. BLSTM-DRNN weights are estimated on the task to map the multi-condition training set of the REVERB Challenge to the clean WSJCAM0 training set, in a frame-by-frame manner. The networks are trained through stochastic on-line gradient descent with a learning rate of 10^{-7} (10^{-6} for power spectrum features) and a momentum of 0.9. Weights are updated after mini-batches of $|\mathcal{B}| = 50$ utterances (feature sequences). Input and output features are mean and variance normalized on the training set. All weights are randomly initialized with Gaussian random numbers ($\mu = 0$, $\sigma = 0.1$). Zero mean Gaussian noise ($\sigma = 0.1$) is added to the inputs in the training phase, and an early stopping strategy is used in order to further help generalization. The experiments are done using the GPU enabled BLSTM-RNN training software CURRENNT written by the author and his colleague, which is publicly available and described in Section 3.5.5. The network topology used in this study is motivated from the feature enhancement experiments on the CHiME-2013 data (cf. Section 5.2). Networks have three hidden layers each consisting of 128 LSTM units for each direction, but without using sub-sampling layers, which were found to decrease performance on the REVERB data.

6.1.2 Results

As evaluation measure for de-reverberation, the WER of the ASR back-end is considered. To test WER differences across systems for statistical significance, the Wilcoxon signed rank test of speaker WER (cf. Section 2.1.3.2) is used at a significance level of $\alpha = .05$.

First, the results obtained by different system architectures for the BLSTM-DRNN front-end are investigated in a clean recognizer without model adaptation. Results are shown in Table 6.1. Considering straightforward mappings from reverberated to clean log spectral features, it is observed that filterbank features perform significantly better than power spectrum features, both on SimData and RealData, while decreasing computational complexity. This might be due to less overfitting in a smaller and less correlated feature space. While a naïve cascade of blind de-reverberation and BLSTM-DRNN does not improve performance (25.12 / 62.6 %), early fusion of reverberated and de-reverberated features gives a significant performance gain of 1.2 % absolute on SimData (20.02 %). Dispensing with the rule-based fusion of de-reverberated spectra with various T_a in favor of a data-based approach yields another gain of 1.0 % absolute on SimData. Only the latter combination approach is better on RealData than the standard feature enhancement approach.

Second, the filterbank domain BLSTM-DRNN ('LSTM') and the BLSTM-DRNN using multiple $\hat{\mathbf{y}}_t$ as input ('Fusion') are compared in recognizers with and without MCT and adaptation. As baselines, unprocessed features ('None') or spectral subtraction based de-reverberation ('SSub') are considered. Results are shown in Table 6.2. As expected, MCT is highly effective even without pre-processing; combined with adaptation, remarkable WERs of 19.42 and 41.4 % are obtained on SimData and RealData (clean: 48.22 / 91.7 %). The effectiveness of MCT can also be attributed to the estimation of LDA-STC on noisy data (using MCT in a recognizer without LDA-STC, 27.48 / 52.8 % WER are reached). As it seems, it is hard to compensate the distortions in reverberated speech with only adaptation (36.93 % WER). A notable trend is that the blind de-reverberation method is only effective for ASR if the recognizer is re-trained using the de-reverberated training set. In this case, it provides an additional WER reduction by 2.2 % relative (19.42 to 18.99 %).

In contrast, data-based de-reverberation (LSTM) gives good results in the clean recognizer without any back-end modification (21.22 % WER). This result is significantly better than MCT (23.41 %) and indicates a good match between the clean and the enhanced features. On the RealData set, LSTM outperforms spectral subtraction when used with the clean recognizer, which indicates good generalization to unseen conditions despite its data-based nature. On SimData, the clean recognizer with the Fusion front-end and adaptation significantly outperforms the MCT recognizer with adaptation (17.43 % vs. 19.42 %). When using the MCT recognizer in combination with pre-processing but without adaptation, performance decreases – this can be explained by a mismatch of reverberated and de-reverberated features, and it can be observed for both LSTM and SSub front-ends. Using recognizer re-training with LSTM enhanced data only slightly improves performance (18.68 to 17.85 % WER), and not at all for Fusion (17.43 to 17.62 %); this might be because the enhanced training set becomes 'too close' to the clean features and remaining distortions on the development set are non-linear. Conversely, the gains by combining

Table 6.2: ASR results on the REVERB Challenge development set, obtained using clean, multi-condition trained (MCT) and re-trained MCT recognizers, with and without basis CMLLR adaptation. SSub: Model-based de-reverberation by spectral subtraction (Section 6.1.1.1). LSTM: Feature enhancement by bidirectional LSTM-DRNN (Section 6.1.1.2). Fusion: Early fusion of original and processed spectral features in the BLSTM-DRNN (see Section 6.1.1.3).

WER [%]	SimData				RealData			
	Processing				*Processing*			
Recognizer	None	SSub	LSTM	Fusion	None	SSub	LSTM	Fusion
Clean	48.22	57.49	21.22	19.06	91.7	84.2	56.5	52.5
+adaptation	36.93	38.94	18.68	**17.43**	80.1	71.7	48.9	44.0
MCT	23.41	46.98	26.36	26.59	47.8	55.3	43.2	39.5
+adaptation	19.42	24.39	18.04	17.80	41.4	42.0	37.7	**36.5**
Re-trained MCT	–	21.39	20.13	18.94	–	46.2	50.1	44.4
+adaptation	–	18.99	17.85	17.69	–	40.5	44.3	40.4

input features become less pronounced with recognizer re-training and adaptation.

On RealData, no significant gains are obtained in general by early fusion. However, the Fusion front-end in the MCT recognizer with adaptation significantly outperforms the baseline MFCC and spectral subtraction front-ends in the same system (36.5 vs. 41.4 / 42.0 % WER). Interestingly, recognizer re-training significantly decreases performance of the LSTM front-end on RealData, which indicates an even stronger mismatch between enhanced training and test features than it is the case on SimData. In contrast, the performance of the spectral subtraction method in a MCT recognizer with adaptation is slightly improved (40.5 % vs. 42.0 %) by re-training.

6.1.3 Conclusions

An effective combination of model- and data-based de-reverberation by spectral subtraction and BLSTM-DRNN feature enhancement for reverberant ASR has been introduced. Results on the 2014 REVERB Challenge data indicate significant gains with respect to traditional multi-condition training and adaptation. Large improvements can be obtained even with a clean recognizer back-end; furthermore, in unseen acoustic conditions the data-based method achieves notable performance compared to the model-based method. Furthermore, for better integration with the ASR back-end, improved cost functions can be investigated for DRNN training, taking account parameters of the ASR back-end instead of just optimizing distances in the spectral domain. To alleviate the need for suited multi-condition training data and to improve generalization, weakly supervised training of feature enhancement using physical models of reverberation will be investigated in addition. Finally, the

spectral feature fusion scheme could be extended to multi-channel input, similar to a recent study by Liu et al. [117] which proposes such fusion for DNN acoustic modeling. In the next section, a cascading approach will be presented for combining multi-channel and single-channel front-ends. Furthermore, discriminative methods, which have proven effective for reverberated speech [192], will be investigated in the next section.

6.2 Combination with multi-channel front-end

In this section, further investigations on the automatic recognition of reverberated speech, as presented by the author and his colleagues in [231], will be summarized. In line with the objectives of this thesis, the main goal of the following study is to further improve the ASR back-end in order to show that front-end enhancement is complementary to state-of-the-art ASR. To this end, discriminative GMM-HMM training, DRNN-based acoustic models and improved language models are investigated. Furthermore, it will be shown that the BLSTM-DRNN feature enhancement method introduced in the previous section can be effectively used to further enhance speech that has been processed by a multi-channel beam-forming method. Again, the evaluation is carried out on the REVERB data (cf. Section 2.2.1.3).

6.2.1 Experiments

Figure 6.2 shows a schematic overview of the proposed ASR techniques. Single- or multi-channel audio is transformed to the time-frequency domain. In case that multiple channels are available, the direct sound is enhanced by estimating the direction of arrival (cross-spectrum phase analysis, CSP) and subsequent delay-and-sum (DS) beamforming. The resulting complex spectrum is converted to a power spectrum and passed through a Mel filterbank. The logarithmic filterbank (Log-FB) outputs are passed to a DRNN for single-channel enhancement, as in the previous section. As before, ASR features are generated directly from the enhanced Log-FB features, by applying feature transformations including DCT, unsupervised adaptation, etc. (cf. below). These ASR features are modeled by a GMM acoustic model (AM), whose likelihoods are combined with the language model (LM) for decoding. Alternatively, a DRNN AM can be used on top of enhanced Log-FB features. In this case, the GMM and DRNN AMs are fused by a multi-stream HMM.

6.2.1.1 Beamforming after DoA estimation

To enhance the direct sound from the speech source in multi-channel mixture signals, a frequency domain delay-and-sum beamformer is applied [98]. Given C microphones, the complex STFT spectra $\mathbf{z}_t(c)$, $c = 1, \ldots, C$ are summed to the enhanced complex

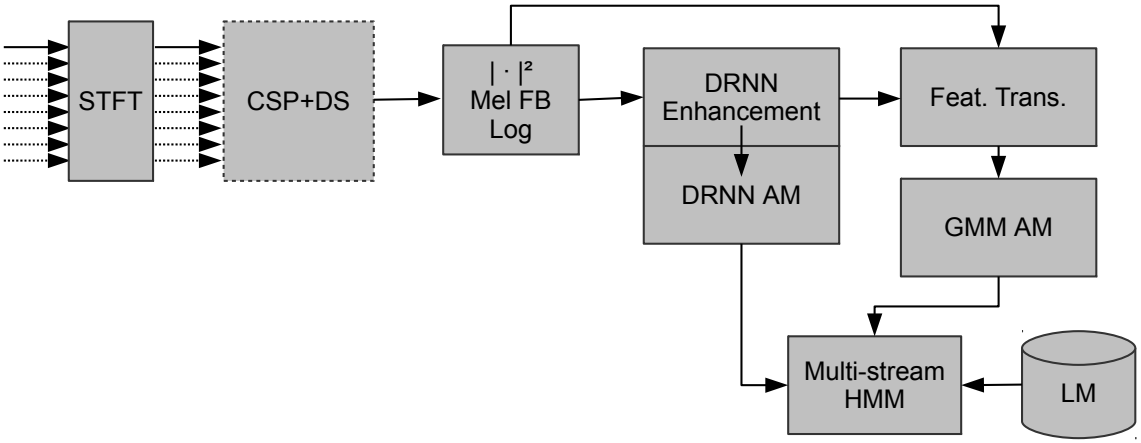


Figure 6.2: Combination of multi- and single-channel front-end enhancement for de-reverberation: Flowchart of the proposed system, using beamforming (CSP+DS) and DRNN enhancement in the front-end, and GMM and/or DRNN acoustic models (AM) in the back-end.

spectrum $\hat{\mathbf{z}}_t$,

$$\hat{\mathbf{z}}_t = \sum_c \mathbf{z}_t(c) \otimes \exp(-\jmath\boldsymbol{\omega}\tau_{1,c}), \tag{6.6}$$

where $\boldsymbol{\omega}$ denotes the angular frequencies and $\jmath$ the imaginary unit. The arrival time delay of the c-th microphone from the first microphone $\tau_{1,c}$ is related to the direction of arrival (DoA) and is estimated by the cross-spectrum phase (CSP) analysis, which uses a cross-power spectrum between two microphones [106] as

$$\tau_{1,c} = \arg\max \mathcal{F}^{-1} \left[\frac{\mathbf{z}_t(1) \otimes \mathbf{z}_t(c)^*}{|\mathbf{z}_t(1)||\mathbf{z}_t(c)|} \right], \tag{6.7}$$

where $\mathcal{F}$ is the STFT operation and * denotes the complex conjugate. To improve the performance of the original CSP method, the post-processing steps described by [194] are added. For the purpose of further processing, the power spectrum $\mathbf{x}_t = |\hat{\mathbf{z}}_t|^2$ is computed.

6.2.1.2 DRNN feature enhancement

The DRNN feature enhancement strategy is similar to Section 6.1, the main differences lying in the feature representation and pre-processing. The training task for single-channel DRNN feature enhancement is to map the reference channel features of the multi-condition set of the REVERB Challenge to the features of the clean WSJCAM0 training set, in a frame-by-frame manner. For the eight-channel case, the beamformer is applied on the multi-condition set, and the resulting features form the training inputs. Log-FB features with $B = 23$ Mel bands are used as input and output features,

and log spectral subtraction (as in Section 5.2) is performed per utterance, so that cepstral mean normalized MFCCs can be obtained simply by applying a discrete cosine transformation (DCT) to the DRNN output $\mathbf{y}_t$. In practice, performing another CMN after DCT turned out to give better performance, because the actual network outputs tend not to have exact zero mean. Using normalized Log-FB, the peak level scaling of the utterances used in the previous system (Section 6.1) can also be dispensed with. Input and output features are globally variance normalized on the training set. Thus, all feature transformations at test time are suitable for utterance-based processing.

6.2.1.3 GMM-HMM back-end

As in Section 6.1, the GMM-HMM acoustic model uses an LDA-STC front-end where nine consecutive frames of 13 MFCC features (coefficients 0–12) are reduced to 40 components. STC transforms are estimated after every other iteration of model training up to iteration 10. To use MCT with front-end enhancement (beam-forming and DRNN), the multi-condition set is processed by the same enhancement steps. Using the original multi-condition training set in combination with enhancement delivered inferior results, sometimes even falling below the clean training result.

With respect to the ASR baseline presented in Section 6.1, three improvements are implemented. First, after conventional ML training of the GMM parameters, discriminative training by the bMMI criterion (3.141) is employed. Second, instead of the standard MAP approach (3.123), MBR decoding (3.127) is performed. Third, a tri-gram language model is included in the experiments. Experiments are performed using the standard 5 k WSJ bi-gram and tri-gram language models. In most of the experiments, the language model weight is fixed at $\mathsf{LMW} = 15$. The open-source Kaldi toolkit is used to implement this improved GMM-HMM back-end.

6.2.1.4 Multi-stream DRNN-GMM-HMM acoustic model

As an extension to the GMM-HMM back-end, DRNN acoustic modeling in a multi-stream HMM approach (cf. Section 3.7.2.3) is considered. To integrate feature enhancement with the multi-stream HMM approach, instead of cascading both as in previous work by the author and his colleagues [53], here the fact is exploited that feature enhancement is performed by a DRNN, and thus the enhancement layers can be stacked with the recognition layers, which allows backpropagation of the recognition error to the enhancement layers. More precisely, given a feature enhancement DRNN with K hidden layers, first the the weight matrices $\mathbf{W}^{(K+2)}, \ldots, \mathbf{W}^{(K+K'+2)}$ of a DRNN with $K + K' + 1$ hidden layers are trained, i.e., the output layer of the feature enhancement DRNN becomes the $K + 1$-th hidden layer of the stacked DRNN. The error function is the cross-entropy between phoneme posteriors and phoneme labels (3.43). After convergence, *all* weight matrices $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(K+K'+2)}$

are re-trained until convergence, using the same error function. Thus, the feature enhancement network is re-trained discriminatively so as to enable good phoneme classification instead of optimizing the squared error in the enhanced features. The 'stacking' procedure also resembles pre-training of the first K hidden layers in a de-noising auto-encoder scheme [204].

In the proposed ASR system, the GMM stream emission probabilities $p(\mathbf{x}_t|s_t)$ are always calculated using features before DRNN enhancement. In particular, the GMM model parameters exactly correspond to those obtained using multi-condition discriminative training (on beamformed data in the 8-channel case). This improved performance over using enhancement in both streams, probably because it makes both streams carry more complementary information. For the multi-stream HMM back-end, the stream weight of the acoustic feature vector $\mathbf{x}_t$ is set to 1.2.

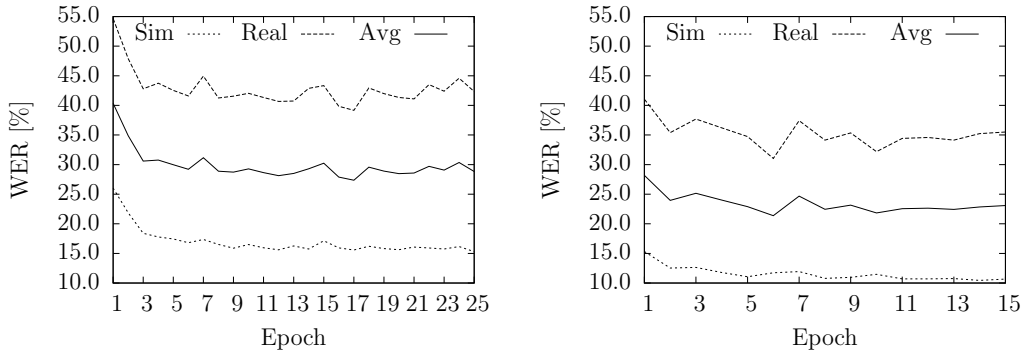
6.2.1.5 Network training

The network topology used in this study was determined based on limited tuning on the REVERB development set. In the case of beamformed input, networks have two hidden layers each consisting of 128 LSTM units for each direction ($N = 2$). For single-channel input, an additional hidden layer is used ($N = 3$), reflecting the fact that the single-channel enhancement task is more complex. All weights are randomly initialized with Gaussian random numbers ($\mu = 0$, $\sigma = 0.1$).

The DRNNs are trained through stochastic on-line gradient descent with a learning rate of 10^{-6} and a momentum of 0.9. Weights are updated after mini-batches of $|\mathcal{B}| = 50$ utterances (feature sequences). Zero mean Gaussian noise ($\sigma = 0.1$) is added to the inputs in the training phase. An early stopping strategy is used to minimize overfitting, by evaluating an error function on the development set for each training epoch and selecting the best network accordingly. In this section, the guidelines of the REVERB Challenge [104] are followed exactly, in order to enable a fair comparison to the state-of-the-art. Evaluating the the sum of squared errors function (3.41) on the development set would not comply with these guidelines, as it would require using the clean development data. Thus, in contrast to the evaluation in Section 6.1, now the ASR performance in terms of WER is used directly.

The networks are trained for a maximum of 50 epochs, and the WER obtained with the enhanced features and the clean trained GMM-HMM acoustic model using the LDA-STC front-end is measured on the SimData and RealData development sets for every training epoch. The best network in terms of the average of the SimData and RealData WERs is used as the final network. It was found that the optimal performance is obtained after only 6 epochs for the beamformed input, and after 17 epochs for the single-channel input. The corresponding WER curves are shown in Figures 6.3a and 6.3b.

For DRNN acoustic modeling, three hidden layers ($K' = 3$) with 50 LSTM units for each direction are used on top of the enhancement layers. A learning rate of 10^{-5}



(a) 1-channel input

(b) 8-channel (BF) input

Figure 6.3: DRNN feature enhancement training: WER curves on development set, clean training, LDA-STC, no adaptation, tri-gram LM. Baseline WER without DRNN: (a) 43.4 / 89.6 %; (b) 24.9 / 72.2 % (SimData / RealData).

and input noise with standard deviation $\sigma = 0.6$ is used. Phoneme alignments are obtained by the multi-condition trained GMM-HMM recognizer with the LDA-STC front-end on the Challenge multi-condition training set. Early stopping is done on the frame-wise phoneme error obtained on a held out part of the multi-condition training data, consisting of utterances from 10 speakers. As per the REVERB Challenge regulations [104], it is not allowed to use the phoneme error on the development data. Both for DRNN feature enhancement and acoustic model training, the open-source CURRENNT software by the author and his colleagues is used (cf. Section 3.5.5).

6.2.1.6 Search parameter optimization

For the final ASR system, the language model weight was tuned on the (unweighted) average WER on the development SimData and RealData sets, by rescoring the decoding lattices accordingly. This is done because front-end enhancement changes the acoustic model likelihoods in a way that the optimal language model weight differs from the standard front-end (cf. Section 5.2). The best development set language model weight (without adaptation) is also used for obtaining the first-pass hypothesis in fMLLR transformation estimation. Additionally, the width of the beam search both in first-pass decoding and lattice generation is increased to avoid search errors which might cause performance drops.

6.2.1.7 System combination

System combination is performed to investigate whether the DRNN feature enhancement system and the DRNN acoustic model system are complementary. For system combination, the Recognizer Output Voting Error Reduction (ROVER) scheme

Table 6.3: Results on the REVERB development set with improved GMM-HMM back-end using MCT, on SimData and RealData. Single-channel MFCC front-end, without pre-processing, LDA-STC transformation. bg/tg: bi-gram/tri-gram language model.

fMLLR	DT (bMMI)	LM	MBR	WER [%] Sim	WER [%] Real
–	–	bg	–	23.41	47.80
√	–	bg	–	19.42	41.42
–	√	bg	–	17.34	46.48
√	√	bg	–	15.53	40.60
√	√	tg	–	12.28	31.05
√	√	tg	√	**12.05**	**30.73**

implemented in NIST's scoring toolkit[1] is used for reproducibility. First, 1-best hypotheses with word level posteriors ('confidences') are generated from the decoding lattices of each system. Then, alignment of the hypotheses is performed by dynamic programming. Finally, for each aligned segment a weighted majority vote is taken.

6.2.2 Results

6.2.2.1 Baseline ASR results

In a first step, the gains by discriminative GMM acoustic model training and tri-gram language modeling, as well as adaptation, are assessed (cf. Table 6.3). Performing MCT with the bMMI criterion (discriminative training) gives a boost on SimData (6 % absolute without adaptation, 4 % absolute with adaptation), but only a slight gain ($\approx 1\,\%$) on RealData, probably because of the mismatched condition. Next, choosing a tri-gram language model instead of a bi-gram one drastically improves performance by about 21 / 24 % relative on SimData / RealData. This shows the importance of adding domain knowledge to achieve increased robustness. Finally, using MBR decoding slightly improves WER both on SimData and RealData. All in all, the performance gain just by improving the ASR back-end are quite impressive, resulting in 52 % / 35 % relative reduction in WER compared to the multi-condition / fMLLR REVERB baseline [104].

6.2.2.2 Results with beamforming and/or spectral enhancement

Table 6.4 shows selected results with single- and multi-channel enhancement. First, a clean recognizer (with fMLLR adaptation) is used to show how well the front-end performs in a recognizer that has never seen noisy data in training (recall that

[1] `ftp://jaguar.ncsl.nist.gov/pub/sctk-1.2c.tgz`

Table 6.4: WER obtained by single- and multi-channel front-ends with and without feature enhancement, on the REVERB development set, using the GMM-HMM back-end, LDA-STC and basis fMLLR, tri-gram LM, MBR. # ch: number of channels; Enh: DRNN enhancement. Optimized: tuning of LM weight and beam width in decoding. [1] bMMI training using clean data.

WER [%]		**Back-End**							
Front-End		Clean trained		MCT		+ DT (bMMI)		Optimized	
# ch	Enh	SIM	REAL	SIM	REAL	SIM	REAL	SIM	REAL
1	–	33.21	77.76	14.92	35.20	12.05	30.73	11.22	30.77
1	√	13.99	35.03	13.51	32.69	10.77	28.30	10.44	26.30
8	–	16.42	54.49	**9.77**	26.34	7.94	23.82	**7.49**	23.91
8	√	**9.72**	**26.49**	9.94	**24.25**	**7.91**	**22.04**	7.67	**21.39**
Oracle		*5.96*	*10.12*	–	–	*5.01*[1]	*10.12*[1]	*5.06*[1]	*9.91*[1]

for the clean recognizer, the fMLLR basis is computed on clean data as well). It can be seen that DRNN enhancement on the single-channel input gives reasonable results with the clean recognizer, significantly outperforming beamforming on its own. However, combination of both in a straightforward cascade gives by far the best result, improving by 71 % / 66 % relative on SIMDATA and REALDATA over the baseline without front-end processing.

When the back-end is trained with multi-condition data, DRNN enhancement improves for all cases except the 8-channel SIMDATA. Furthermore, if the search parameters are optimized, the performance difference between features with and without DRNN enhancement becomes larger on REALDATA (both for 1-channel and 8-channel). This probably shows that for DRNN enhanced features, there are some search errors in the baseline ASR due to a different dynamic range of acoustic model likelihoods. Furthermore, it was found that they gave generally higher acoustic likelihoods, which requires adjusting the LM weight.

All in all, while the performance gains by enhancement are notable, they are still far from the performance obtained in clean conditions, especially on REALDATA. It has to be noted, though, that the ASR task of REALDATA itself seems much harder than the SIMDATA task, when eliminating reverberation and noise as confounding factors – the WER in clean conditions is about twice as high. This could be due to a mismatch in accent and/or speaking style, since the MC-WSJ-AV corpus was recorded in a different site and dialect region than the WSJCAM0 corpus, and a different recording protocol was used (speakers standing in a meeting room, rather than sitting in a sound-proof booth).

Table 6.5: REVERB development (Dev.) and evaluation set results (SimData) for selected 1-channel and 8-channel systems, as well as system combination. Evaluation set results are given per room and microphone distance (near / far). All with MCT, LDA-STC, basis fMLLR, bMMI, tri-gram LM, MBR, optimized search parameters. BF: beamforming. Enh: DRNN enhancement.

WER [%]	**Dev.**	**Evaluation set**						
	Avg	Room 1		Room 2		Room 3		Avg
		near	far	near	far	near	far	
1-channel systems								
REVERB baseline	25.16	16.23	18.71	20.50	32.47	24.76	38.88	25.26
GMM-HMM	11.22	6.37	7.67	8.76	16.22	10.66	20.20	11.65
GMM-HMM (Enh)	**10.44**	6.39	7.52	8.41	14.15	9.47	15.30	**10.21**
8-channel systems (BF)								
GMM-HMM (I)	7.49	5.39	5.93	6.38	9.71	6.87	12.47	7.79
DRNN+GMM-HMM (II)	6.48	5.32	5.76	6.19	9.00	6.65	10.78	7.28
GMM-HMM (Enh) (III)	7.67	5.49	6.12	6.80	9.69	7.13	11.28	7.75
8-channel systems (BF) – System combination								
ROVER I+II	6.58	5.00	5.44	6.04	8.95	6.70	11.04	7.20
ROVER II+III	**6.44**	5.08	5.66	5.95	8.58	6.72	10.10	**7.02**
ROVER I+III	7.15	5.29	5.98	6.24	9.34	6.98	11.19	7.50
ROVER I+II+III	6.75	5.30	5.88	6.06	9.00	6.89	11.07	7.37
Oracle enhancement								
GMM-HMM	*5.06*	*5.34*		*5.55*		*6.07*		*5.65*

6.2.2.3 Test set evaluation and system combination

Tables 6.5 and 6.6 shows the detailed results on the evaluation set (Table 6.5: SimData / Table 6.6: RealData) obtained by the 1-channel and 8-channel systems, with and without DRNN enhancement. In the 1-channel case, the best result is 10.21 / 26.73 % WER (SimData / RealData). Comparing the results obtained at the 'near' microphone distance with the 8-channel front-end to the oracle results on SimData, one can observe that the performance is already quite close (to be fair, one has to look at the GMM-HMM results). However, at the 'far' distance a significant difference remains. The same holds for the RealData set, which can be explained by the mismatched training / test conditions.

To investigate whether DRNN enhancement and acoustic modeling are complementary, let us first outline the results with the multi-stream DRNN+GMM-HMM recognizer. It obtains 6.48 / 7.28 % WER on SimData (development / evaluation set), which is the best single system result for SimData. However, the performance on RealData is lower than the one of the DRNN enhancement GMM-HMM system. This could be due to the different cost functions for enhancement and acoustic modeling, which apparently leads to a better modeling of the SimData which is

Table 6.6: REVERB development and evaluation set results (RealData) for selected 1-channel and 8-channel systems, as well as system combination. Evaluation set results are given per room and microphone distance (near / far). All with MCT, LDA-STC, basis fMLLR, bMMI, tri-gram LM, MBR, optimized search parameters. BF: beamforming. Enh: DRNN enhancement.

WER [%]	**Dev.**	**Evaluation set**		
	Avg.	near	far	Avg
1-channel systems				
REVERB baseline	47.23	50.74	47.57	49.16
GMM-HMM	30.77	31.84	30.93	31.39
GMM-HMM (Enh)	**26.30**	25.39	28.06	**26.73**
8-channel systems (BF)				
GMM-HMM (I)	23.91	20.25	23.16	21.71
DRNN+GMM-HMM (II)	22.07	19.74	23.63	21.69
GMM-HMM (Enh) (III)	21.39	17.66	22.52	20.09
8-channel systems (BF) – System combination				
ROVER I+II	22.60	19.26	22.01	20.64
ROVER II+III	**20.24**	16.96	22.25	**19.61**
ROVER I+III	21.20	17.57	21.10	*19.34*
ROVER I+II+III	21.62	17.76	22.62	20.19
Oracle enhancement				
GMM-HMM	*9.91*		*8.47*	

close to the training data while worsening the results on the mismatched RealData – recall that similar differences can be seen for the ML vs. bMMI objectives in GMM training, cf. Table 6.3. It is also in accordance with the fact that DRNN acoustic modeling worked better than feature enhancement on the CHiME data, which do not exhibit a strong training / test mismatch (cf. Section 5.2).

Let us now investigate the combination of two or three of the 8-channel systems. As it seems, the best combination on the development set is the DRNN acoustic model with the DRNN enhancement system. This combination achieves the best development set WER on SimData and RealData, indicating a certain degree of complementarity between these approaches although they use similar modeling techniques (both involving a DRNN and a GMM). This combination also achieves the best average WER on the evaluation set, reaching down to 7.02 and 19.61 % WER on SimData and RealData.

6.2.3 Conclusions

In this section, the complementarity of beam-forming and BLSTM-DRNN single-channel feature enhancement were shown, thereby uniting physical model based and training based approaches in an effective system. By late fusion, it was also

possible to combine performance gains from DRNN feature enhancement and DRNN acoustic modeling. However, using the former as front-end for the latter did not significantly improve the results. This can be explained by the fact that using feature enhancement layers is a kind of pre-training for the DRNN acoustic model weights, and pre-training does not always improve over a suitable random weight initialization [29].

The presented ASR system was submitted by the author and his colleagues as the MERL / MELCO / TUM system to the REVERB Challenge 2014 [231] and, according to the official results[2], achieved the second best performance on the RealData among the submissions of the participants exactly following the guidelines and not being affiliated with the organizers – second only to the one by Tachioka et al. [194], which, however, uses as many as sixteen acoustic models.

The proposed system architecture allows both multi-channel and single-channel processing. In particular, the proposed integration of (physical) model based multi-channel and data based single-channel processing has the advantage that the models do not have to be re-trained for different microphone array setups (as would likely be the case if the input features from the eight channels were just concatenated for DRNN enhancement).

The system has been designed to allow utterance based processing, but needs multiple recognition passes at this stage. It is thus suitable, e.g., for server-based ASR systems. Most of the system components could be used in an on-line ASR system as-is or with small modifications. The CSP+DS front-end is fully on-line capable. Low-latency enhancement could be done by using unidirectional DRNNs (possibly with a small lookahead); it remains to evaluate the performance of this setup.

From the results, it is evident that there is a fundamental limitation to the performance of training-based approaches in a mismatched condition setup, such as the RealData scenario. In a practical application one could perform semi-supervised training of the DRNN enhancement and acoustic model on 'field data' (in the Challenge scenario, this would be other data from the MC-WSJ-AV corpus, which was not allowed to be used). This, along with other methods to improve generalization, such as weight noise for DRNNs [64], will be a promising direction for future research.

[2] `http://reverb2014.dereverberation.com/result_asr.html`

7

Applications in Music Information Retrieval

If a composer could say what he had to say in words he would not bother trying to say it in music. – Gustav Mahler

Polyphonic music is a natural, practically relevant, and challenging test bed for multi-source recognition algorithms. This is because polyphonic music – similar to speech – is associated with symbolic information (the notes played, the conveyed emotion, the text sung by a lead vocalist, etc.), yet this information is naturally embedded in a complex mixture of audio signals (vocalist, instrumentalists, percussion and crowd noise, etc.) Thus, after having discussed a variety of speech applications, this last application chapter will be devoted to music processing.

In addition, the first two sections will go beyond traditional separation and transcription tasks, and deal with 'the voice behind the words', or computational paralinguistics [168], in polyphonic music. As highlighted by the author and his colleagues [221], the synergies between multi-source recognition in polyphonic music and environmentally robust speech processing are obvious: for example, there is a direct correspondence between multi-talker ASR and automatic music transcription.

7.1 Singer characterization

In the field of speech processing, a large body of literature exists on speaker characterization, cf., e.g., [6, 130, 214, 227]. Schuller and Batliner [174] introduced the taxonomy of speaker traits (long-term attributes) and states (short-term conditions). An analogous taxonomy can be established for *singer* characterization, that is, automatically recognizing meta data of the performing vocalist(s) in recorded music. Yet, singer characterization is currently still an under-researched topic in MIR, compared to speech processing. As some of the first studies in the field, gender, age, and race recognition have been introduced by the author and his colleagues [175, 213, 218].

In this section, the author's and his colleagues' experiments with gender and age recognition as previously presented in [213] are described. Singing style recognition, as a shorter-term phenomenon, is addressed in the following section (Section 7.2).

Speaker trait recognition is often used in dialog systems to improve service quality [19], yet another important area of application is forensics where it can deliver cues on the identities of unknown speakers [94]. Likewise, applications in music processing can be found in categorization and query of large databases with potentially unknown artists – that is, artists for whom not enough reliable training data is available for building singer identification models as, e.g., in [127]. Robustly extracting a variety of meta information can then allow the artist to be identified in a large collection of artist meta data, such as the Internet Movie Database (IMDB). In addition, exploiting gender information is known to be very useful for building models for other MIR tasks such as automatic lyrics transcription [126].

Current research in speech processing suggests that the automatic determination of age in full realism is challenging even in clean, spoken language, and even gender recognition is far away from perfection if a representative sample of the population is taken [176]. In comparison to speech, recognition of *singer* traits is expected to be an even more challenging task due to pitch variability, influence of voice training, and presence of multiple vocalists as well as instrumental accompaniment. The latter particulary motivates the consideration of the topic for this thesis. The goal is to show that by separating the leading vocals, the task of singer characterization is considerably simplified. In this respect, it is promising that an earlier study by the author and his colleagues [218] showed that gender identification of unseen artists in recorded popular music can be performed with over 90 % accuracy in full realism, by extracting the leading voice through an extension of non-negative matrix factorization (NMF) [36], and using BLSTM-RNN classification.

The following investigation builds on that study, extends the findings by combined percussive/harmonic and leading voice separation by NMF, and adds the recognition of age as a new dimension. For evaluation of automatic singer-independent classification, the UltraStar database as presented in Section 2.2.2.2 will be used.

7.1.1 Experiments

A major part of the experiments is devoted to finding the optimal preprocessing by source separation for recognition of vocalist gender and age. To this end, harmonic enhancement as in [77, 218] and extraction of the leading voice as in [36], as well as a combination of both, are investigated.

7.1.1.1 Enhancement of Harmonic Parts

Enhancement of harmonic parts is performed according to the generic, weakly supervised scheme outlined in Section 3.2.1. After decomposing the STFT magnitude

spectrogram into NMF components, an SVM is used to to discriminate between components corresponding to percussive or non-percussive signal parts. The classifier is trained on a manually labeled set of NMF components extracted from popular music as described in [170]. Features are extracted from both the NMF dictionaries and the activations. For straightforward reproducibility of the experiments, the default parameters of the publicly available[1] drum beat separation demonstrator of the source separation toolkit openBliSSART written by the author and his colleagues [212, 217] are used.

The STFT parameters are set as frame shift $\Delta\tau = 30\,\text{ms}$ and window size 60 ms, using the square root of the Hann window as proposed by Helen and Virtanen [77]. $K = 100$ NMF iterations and $R = 50$ NMF components are used. Semi-supervised NMF is used where 20 dictionary atoms for percussive components are initialized from typical drum spectra delivered with the openBliSSART demonstrator. To allow the algorithm to use different dictionaries for the individual sections of a song without increasing the dimensionality of the factorization, a blind segmentation into frame-synchronous non-overlapping chunks is performed before the separation, as described in [218].

7.1.1.2 Leading Voice Separation

The second method used to facilitate singer trait identification is the leading voice separation approach described in Section 3.2.5 and [35, 36]. To ensure best reproducibility of the results, an open-source implementation[2] of the algorithm with default parameters was used. The same chunking as for harmonic / percussive separation was applied.

7.1.1.3 Combined Source Separation Approaches

In a preliminary experiment it was found that when the leading voice separation is applied to popular music, part of the drum track may remain after separation. Hence, for this study, cascading of both separation techniques was considered: harmonic enhancement after leading voice separation (LV-HE), and vice versa (HE-LV). There, time domain signals are synthesized inbetween the two separation stages, in order to be able to use different NMF parameterizations for both algorithms.

7.1.1.4 Acoustic Features for Classification

The features used for classification exactly correspond to those used in [175] and were extracted for each beat using TUM's open-source toolkit openSMILE [42]. The short-time energy, zero-, and mean-crossing rate are considered, which are

[1] `http://openblissart.github.com/openBliSSART`
[2] `http://www.durrieu.ch/phd/software.html`

known to indicate vocal presence. In addition, several features are derived from the normalized autocorrelation sequence of the DFT coefficients, namely voicing probability, fundamental frequency (F0) and harmonics-to-noise ratio (HNR). F0 is the location of the highest peak of the autocorrelation sequence aside from the maximum at zero. HNR is computed by the value of this peak. These pitch and voice quality parameters have been successfully used in paralinguistic information assessment from speech [176]. Further, MFCCs 0–12 and their first-order delta regression coefficients are added, which are known to capture the characteristic qualities of individual voices for singer identification [127]. Thus, altogether the feature set comprises 46 time-varying features. The employed configuration of the openSMILE toolkit is provided for further reproducibility[3].

7.1.1.5 Classification by BLSTM-RNNs

As in [218], sequential vocalist gender classification with BLSTM-RNNs has been observed greatly superior to beat-by-beat classification by SVMs or Hidden Naive Bayes (90.77 % beat level accuracy on original signals vs. 72.78 % and 76.17 %), this classifier was chosen for the present study.

Individual BLSTM-RNNs are trained for each classification task. The tasks investigated comprise binary age and gender classification tasks (young / old, female / male, cf. Section 2.2.2.2), as well as ternary tasks where additionally the beats without any vocals must be identified. As in [218], the networks have one hidden layer with 80 LSTM memory cells for each direction. The size of the input layer is equal to the number of features (46), while the size of the output layer is equal to the number of classes to discriminate (2–3). A softmax output layer is used so that the outputs represent the posterior class probabilities. The songs in the test set are presented frame by frame (in correct temporal order) to the input layer, and each frame is assigned to the class with the highest pseudo-probability as indicated by the output layer.

For network training, supervised learning with early stopping was used as follows: The network weights were intialized randomly from a Gaussian distribution ($\mu = 0, \sigma = 0.1$). To improve generalization, the order of the input sequences was determined randomly, and Gaussian noise ($\mu = 0, \sigma = 0.3$) was added to the input activations. The network weights were iteratively updated using resilient propagation [161]. To prevent over-fitting, the performance (in terms of classification error) on the validation set was evaluated after each training iteration (epoch). Once no improvement over 20 epochs had been observed, the training was stopped and the network with the best performance on the validation set was used as the final network. Graves' open-source BLSTM-RNN implementation [65] is used.

[3] `http://www.openaudio.eu`

Table 7.1: Beat-wise BLSTM-RNN classification of UltraStar test set on 2- and 3-class tasks (V: presence of vocals; G: gender; A: age). Preprocessing: HE = harmonic enhancement (Section 7.1.1.1); LV = leading voice extraction (Section 7.1.1.2); LV-HE: HE after LV; HE-LV: LV after HE.

		Preprocessing									
[%]		–		**HE**		**LV**		**LV-HE**		**HE-LV**	
task	classes	**Rec**	**Acc**	**Rec**	**Acc**	**Rec**	**Acc**	**Rec**	**Acc**	**Rec**	**Acc**
V	0/1	74.55	74.50	73.82	73.84	**75.77**	75.81	75.40	75.41	75.09	75.11
G	0/m/f	63.75	68.54	65.65	68.91	**69.29**	71.31	67.90	70.41	68.52	70.44
	m/f	86.67	91.09	88.45	91.91	86.93	91.12	**89.61**	93.60	87.76	92.50
A	0/y/o	51.02	57.61	50.00	57.14	**53.50**	59.85	51.26	58.86	50.01	57.72
	y/o	55.30	55.60	**57.55**	56.56	53.93	53.63	55.97	54.89	54.69	54.17

7.1.2 Results

The primary measure for evaluating performance of automatic singer trait recognition is recall (Rec) (2.17) on beat level. Due to the general class imbalance (Table 2.4), recall better represents the discrimination power of the classifier than accuracy (Acc). Note that either random guessing or always picking the majority class would both achieve a recall of 33.33 % in ternary and 50.00 % in binary classification tasks.

7.1.2.1 Results on Beat Level

Results on beat level are shown in Table 7.1. In order to estimate the difficulty of the evaluated singer trait recognition tasks in full realism, first a BLSTM-RNN was evaluated on the task to recognize the presence of a singer (vocal activity recognition). It turns out that this can be done with over 75 % recall when using the leading voice extraction. Note that due to the underlying model (3.35), (3.36), (3.37), the separation algorithm does not discriminate human voices from other harmonic sources with similar filters; hence, the vocal activity recognition task remains non-trivial.

Best results on the 2-class gender recognition task are obtained by the combination of source separation algorithms (LV-HE, 89.61 % recall) while in the 3-class task, best recall is achieved by the LV algorithm alone (69.29 %), which significantly ($\alpha = .001$) outperforms the baseline without preprocessing. Notably, this recall also higher than it would be expected if the accuracies of vocal activity and 2-class gender recognition were independent, potentially showing the benefit of the 'multi-task' ternary modeling. Up to 57.55 % recall are achieved in *age* recognition when using HE; while this is clearly below typical results on spoken language, it is significantly above chance level (50 %) according to a z-test ($\alpha = .001$).

Table 7.2: Song-wise BLSTM-RNN predictions on UltraStar test set by beat-wise majority vote. Vote on 3-class tasks (ignoring beats classified as 0) or 2-class tasks. Preprocessing as in Table 7.1.

		Preprocessing									
[%]		–		**HE**		**LV**		**LV-HE**		**HE-LV**	
task	vote on	**Rec**	**Acc**	**Rec**	**Acc**	**Rec**	**Acc**	**Rec**	**Acc**	**Rec**	**Acc**
G	0/m/f	80.9	87.0	81.7	85.6	87.7	90.9	**91.3**	92.4	87.7	90.9
	m/f	86.9	90.1	89.0	90.9	87.7	90.9	**89.6**	93.9	89.6	93.9
A	0/y/o	55.2	54.5	54.6	54.1	56.0	54.1	**56.9**	57.4	50.9	51.6
	y/o	54.5	54.5	57.0	55.7	52.2	51.6	53.4	52.5	**58.9**	58.2

7.1.2.2 Results on Song Level

As a performance estimate for 'tagging' entire songs, for each scenario (front-end separation / classification task) the accuracies and recalls of a majority vote on beat level compared with the most frequent class label on beat level were calculated. Note that such measurements are rather heuristic in nature, since a song level ground truth cannot always be established due to phenomena such as alternating male / female vocalists. To briefly comment on the results (shown in detail in Table 7.2), song level gender can be recognized with up to 91.3 % recall, and age with 58.9 % recall. The song level gender estimation works even better from the 3-class than the 2-class beat level task. LV-HE preprocessing delivers overall best results.

7.1.2.3 Discussion

Compared to usual results obtained on spoken language [176], the performance of age recognition is rather low; the task seems to be especially challenging on the considered type of 'chart' popular music with a prevalence of singers in their twenties. At least, when using gender-dependent models for age, 61.63 % recall could be achieved for males; for females there is not enough training data.

A promising direction for further research may be to investigate different units of analysis, such as longer-term statistical functionals that are commonly used in paralinguistic information retrieval from speech [176], instead of recognition at the beat level. Still, this is not fully straightforward due to the feature variation, especially for pitch, which will necessitate methods for robust pitch estimation and transformation.

7.1.3 Conclusions

Inspired by previous successful studies on vocalist gender recognition, the concept of singer trait recognition was introduced and exemplified by age recognition in a large collection of recorded popular music. In the light of the research objectives of this

thesis, it is an highly interesting result that explicit separation of the leading voice according to a weakly supervised NMF algorithm could improve the performance of an RNN based vocal activity, gender, and age recognizer. While gender recognition is now close to perfection even on beat level (up to 93.60 % WA on unseen test data), it was also shown that even in chart music with a prevalence of singers from 20–30 years, age recognition could be performed significantly above chance level; still, when aiming at real-life applications new directions in research must be taken.

Future work should primarily focus on more variation in data by not only including chart music, but also classical, jazz and non-Western music. Furthermore, multi-task learning similar to the approach of Stadermann et al. [189] can be investigated to exploit interdependencies of singer traits and phonetic content, in order to improve generalization.

7.2 Singing style recognition

Having addressed the recognition of singer traits, let us now move to short-term singer states. As an example for singer states, the displayed emotion can be named, for example singing enthusiasm [25]. In this chapter, another short-term state, *singing style*, is introduced in analogy to speaking style (cf., e.g., [138, 248]). Its automatic recognition is exemplified by *vibrato*, as in the study previously presented by the author and his colleagues in [219].

Vibrato is usually defined characterized as a periodic oscillation of the fundamental frequency (pitch) of the voice at a rate of 4–8 Hz. Applications of automatic recognition of vibrato singing in recorded polyphonic music include singer identification, as different singers develop their own style of vibrato [139], as well as other MIR tasks such as structure and performance analysis. Furthermore, it can be useful for highly efficient audio coding, e. g., as an attribute for sound synthesis [33].

Few performance studies exist on fully automatic recognition of vibrato singing [3, 144]. Some of these [3, 144] are limited to monophonic recordings, which may be justified for applications such as coaching of singing students [3]. For retrieval applications in recorded music, however, one has to cope with additional sources from accompaniment. Salamon et al. [164] propose vibrato features for genre classification of polyphonic music, but without comparing the performance of vibrato recognition against a ground truth. Per definition, it can be assumed that automatic recognition of vibrato strongly depends on robustness of pitch extraction, which, however, is challenging in the condition of multi-source mono- or stereophonic recordings [120]. In fact, as shown below, the extraction of pitch is challenging in the presence of instrumental accompaniment, leading to unsatisfactory classification accuracy (61.1 %) if only the F0 frequency spectrum is used as features.

In line with the objectives of this thesis, the following investigation deals with robustness of vibrato classification against ‘noise’ by instrumental accompaniment.

Table 7.3: Feature extraction for vibrato recognition: Low-level descriptors (LLDs) and functionals.

LLD	Functionals
(Δ) Log. F0	*Extremes*
(Δ) RMS energy	Range
(Δ) Auditory spectrum	Position min., max.
	Dist. min./max. from arith. mean
	Moments
	Std. dev., skewness, kurtosis
	Temporal evolution
	Mean crossing rate (MCR)
	DCT coefficients 1–6
	Percentiles
	Inter-quartile ranges 1–2, 2–3, 1–3
	DFT coefficients 1–10 (log. F0)
	Arith. mean (of Δ)

Several strategies are evaluated to address robustness: First, besides the obvious frequency analysis of the F0 contour [33], other functionals of pitch itself, and also other 'low-level' features, are considered. Second, feature selection is investigated to keep only those features which are unaffected by noise. Third, the hypothesis is tested whether extraction of the leading voice by semi-supervised NMF (cf. Section 3.2.5) can contribute to the robustness of vibrato recognition.

A pecularity of this section is to characterize individual acoustic features by their robustness in fully automatic recognition. This is in contrast to the speech enhancement and automatic speech recognition applications, where the features considered were spectra, and unless noise is constrained to particularly low or high frequency ranges, no difference is expected in the noise robustness of these features. In contrast, for vibrato recognition, higher-level, musically motivated features are used (cf. below), and it will be shown that variants of these features exhibit strongly different properties in robustness.

7.2.1 Experiments

7.2.1.1 Feature Extraction

The feature set chosen in this study is a combination of generic features, similar to the ones in Section 7.1.1.4, and musically motivated, 'hand-crafted' features for the recognition of vibrato. An overview of the feature set is given in Table 7.3.

7.2.1.1.1 Low-Devel Descriptors As frame-wise low-level descriptors (LLDs), pitch (F0), root mean square (RMS) energy and auditory spectrum are extracted. All feature extraction is performed without manual post-processing such as correction of octave errors. Pitch detection is based on the Subharmonic Summation (SHS) algorithm [80] to identify pitch candidates in the frequency domain. The power spectrum is transformed to a logarithmic scale by spline interpolation and shifted spectra are added to the original spectrum to sum up the harmonics. The result is the so called SHS spectrum (SHSS). In theory there should be one prominent peak at F_0; however, in practice higher harmonics are also present. The N highest peaks in the SHSS are identified, and peak position and amplitude are adjusted by three point quadratic regression using the peak and its left and right neighbors to fit a parabola. A voicing probability is assigned for each candidate based on the (adjusted) peak's amplitude in the SHSS. The arithmetic mean (μ_s) of the bins in the SHSS is computed. For each pitch candidate i with a pitch candidate score s_{ci} (= peak amplitude) greater than μ_s the voicing probability p_{vi} is computed as $p_{vi} = 1.0 - \frac{\mu_s}{s_{ci}}$. Otherwise ($s_{ci} \leq \mu_s$), $p_{vi} = 0$. The final pitch contour as well as the final voicing decision is smoothed by dynamic programming where soft penalties for jumps and out-of-range values are applied. The algorithm is based on the Viterbi pitch smoothing as presented in [120], which was slightly modified for the SHS pitch values and voicing probabilities. This implementation of pitch extraction is available in TUM's open-source toolkit openSMILE [42].

To make the absolute amount of pitch variation independent of the fundamental frequency, the natural logarithm of the pitch is taken. Pitch and RMS energy are extracted from 50 ms frames of the audio signal windowed with a Gaussian function at 10 ms frame shift. The auditory spectrum is computed by reweighting the Mel frequency bands 1–26 obtained from a short-time Fourier transform (STFT) with 25 ms frame size and a Hamming window function, similarly to the procedure performed in extraction of PLP features [78].

7.2.1.1.2 Functionals To capture variation of the low-level descriptors, first order delta regression coefficients (Δ) are extracted according to Section 3.7.1.1, spanning five frames ($W = 2$). Furthermore, segment-wise functionals are computed from both the low-level descriptors and their Δ coefficients.

To capture pitch oscillations in the range relevant for vibrato, Discrete Fourier Transform (DFT) coefficients 1–10 are extracted from overlapping windows of 128 logarithmic F0 points which are centered to zero mean, corresponding to a window size of 1.28 s to achieve sufficient frequency resolution. Windows overlap by 64 points and are multiplied by a Hamming function before applying the Fourier transformation. Zero padding is applied for segments shorter than the length of a single window (1.28 s); for segments longer than a single window, incomplete windows at the end are discarded – it was often observed that the DFT coefficients from the previous

windows were deteriorated by the alteration of the frequency distribution in the last window due to zero padding. Finally, the mean across windows is taken for each DFT coefficient.

Other, more generic functionals applied to all kinds of LLDs include moments, range (absolute difference of minimum and maximum), distance of minimum and maximum from the arithmetic mean, standard deviation and higher order moments, mean crossing rate, Discrete Cosine Transform (DCT) coefficients 1–6 and finally inter-quartile ranges (IQR, absolute differences between quartiles). These functionals are often used in segment-based functional extraction from pitch and other LLDs for paralinguistic information retrieval [177]. Frames (erroneously) classified as unvoiced are excluded from calculation of functionals and Δ coefficients from the F0 contour, except for DFT coefficients to preserve the frequency of periodic oscillations – in that case, unvoiced frames are assumed to be equivalent to the mean of the F0 points in the corresponding window(s). Note that only those functionals are chosen which are independent of the absolute values of the LLDs, in order to capture signal variation instead of overall characteristics. Thus, for instance, the arithmetic mean is only computed from the delta coefficients, not from the LLDs themselves.

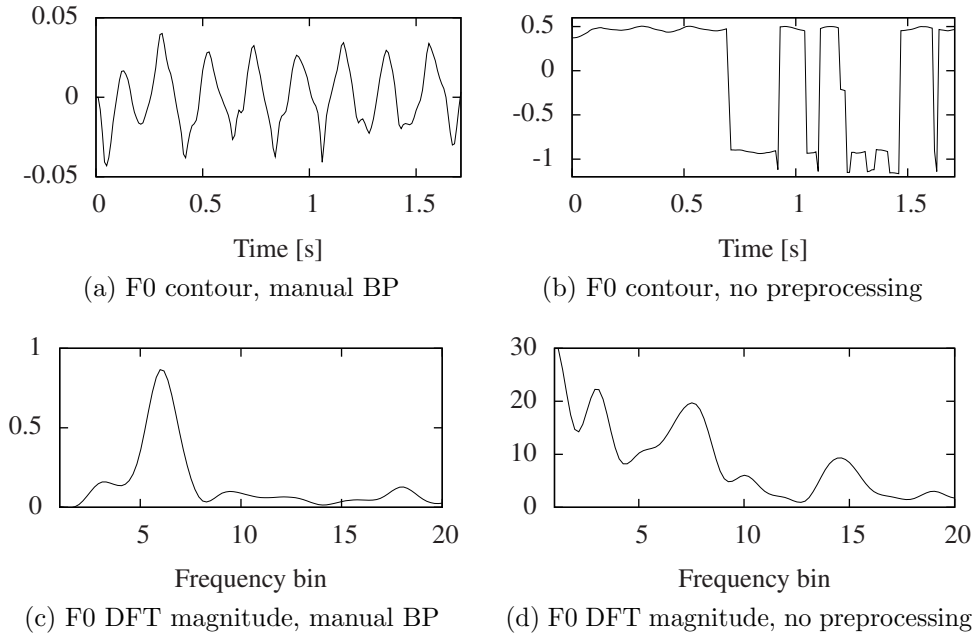
7.2.1.2 Automatic Classification Experiments

Preprocessing As a weakly supervised preprocessing method for extraction of the singer's voice in the presence of background music, the leading voice separation approach described in Section 3.2.5 and [36] is used. While this method also includes pitch tracking, it differs from traditional methods (cf. Section 7.2.1.1.1) by explicit modeling of accompaniment and the vocal tract of the singer. To ensure best reproducibility of the findings, an open-source implementation[4] of the algorithm with default parameters was used.

To get an upper performance bound by simulating near-perfect pitch extraction, a band-pass (BP) filter was applied for each segment. The pass-band was manually set to capture single harmonic(s) of the singing voice including pitch variations, so that robust automatic pitch determination is straightforward. These filters were applied to the DFT spectrum of each segment as a whole to achieve best frequency resolution.

Classification Algorithm For classification the SimpleLogistic algorithm [109] was used, as implemented in the Weka toolkit [74]. This classifier is particularly suitable for small to medium feature spaces as it is based on boosting of one-dimensional logistic regression functions. The number of boosting iterations was cross-validated on the training set, using the default parameters in the Weka toolkit. In order to optimize on a balanced recall of both the vibrato and non-vibrato classes,

[4] `http://www.durrieu.ch/phd/software.html`



(a) F0 contour, manual BP

(b) F0 contour, no preprocessing

(c) F0 DFT magnitude, manual BP

(d) F0 DFT magnitude, no preprocessing

Figure 7.1: Logarithmic F0 contour (normalized to zero mean) and its DFT coefficients for a segment of jazz music, without or with manual bandpass (BP) filtering to extract the singing voice.

a simple method of training instance up-sampling was applied as follows: For testing on each fold, all instances of the minority class (non-vibrato) in the two folds used for training were copied so as to achieve uniform a-priori class probabilities in the training set for the classifier.

7.2.2 Results

7.2.2.1 Classification Results

In Table 7.4, the recall (Rec) of the vibrato and non-vibrato classes is shown. Similar to the experiments with singer characterization, recall is preferred over accuracy as evaluation measure due to the class imbalance. It is observed that the joint feature space of F0 functionals provides highly robust classification, reaching 84.9 % recall without preprocessing. Still, this is significantly ($p < .05$ according to a one-tailed z-test) below the upper bound of 90.1 % achieved by manual BP filtering, indicating that accompaniment makes vibrato classification considerably more challenging. Surprisingly, vocal separation significantly degrades performance by over 8 % absolute compared to using no signal enhancement; this can be attributed to the fact that the algorithm was not designed to capture slight pitch variations, as the source STFT is assumed to be constant in the vocal model [36].

An analysis of different functional groups of F0 reveals that as expected, DFT

Table 7.4: Recall of vibrato / non-vibrato instances in singer-independent 3-fold stratified cross-validation using SimpleLogistic classification by feature set and functional groups. Preprocessing by vocal separation or manual band-pass (BP).

Recall [%]		Preprocessing		
LLD	Functionals	—	Voc.sep.	BP
(Δ) Log. F0	All	84.9	73.3	90.1
	DFT coeff.	61.1	66.1	**91.7**
	DCT / MCR	65.4	60.9	66.2
	Extremes	55.5	51.8	81.1
	Moments	56.4	53.5	79.1
	Percentiles	**86.9**	**78.0**	88.0
(Δ) Aud.spec.	All	67.0	59.7	61.9
(Δ) Energy	All	59.5	67.0	73.8
All	All	67.1	65.0	77.7
(Δ) F0 + Energy	All	79.8	75.1	82.7

coefficients allow highly robust classification on their own if pitch extraction is facilitated by manual bandpass filtering (91.7 % recall); this performance, however, is vastly degraded to 61.1 % recall without such preprocessing. Notably, vocal separation can partly alleviate this downgrade (66.1 % recall). An explanation for the performance drop is shown in Figure 7.1 for an example of jazz music. It is clearly visible that while there is a prominent peak in the 6th frequency bin (4.7 Hz) for 'ideal' F0 extraction, in the real-life setting without manual preprocessing the DFT coefficients are significantly deteriorated due to the pitch estimation errors. In the example, these are caused by chords played by a piano in the second half of the segment.

As to other functionals, extremes and moments both are robust (around 80 % recall) for bandpass filtering, yet performing not significantly ($p > .05$) above chance level (50 % recall) without preprocessing, probably due to sensitivity against outliers. Their performance apparently cannot be restored by vocal separation. In contrast, DCT and MCR perform similarly above chance level regardless of the preprocessing, but generally deliver unsatisfactory accuracy on their own (up to 66.2 % recall). Finally, and most importantly, percentile features, i. e., inter-quartile ranges enable highly robust classification (86.9 % recall) even without preprocessing; for bandpass filtering, their recall of 88.0 % is still remarkable, yet significantly below the one of DFT coefficients. This behavior will be further investigated below.

Before, briefly the performance of other LLDs than pitch is summarized. While both, auditory spectrum and RMS energy cannot compete with F0 in terms of recall, it is notable that both are observed highly above chance level ($\alpha = .005$). Interestingly, the auditory spectrum seems to carry valuable information especially

when using no preprocessing (67.0 % recall); this is possibly due to the strong interdependency with musical genre (cf. Table 2.3), which can be detected by spectral features. Furthermore, energy is considerably informative; this is *not* simply due to vibrato occurring in accented notes of higher loudness since the functionals are independent of the absolute energy (see above). Still, none of the 'alternative' LLDs seem to complement the information gained from the pitch, as classification with the union of feature sets (all or F0 + energy) cannot significantly surpass the performance of F0 functionals alone.

Overall, the effect of vocal separation is disappointing for the vibrato recognition task. While one could extend the algorithm to allow slight variations in the source model ($\mathbf{W}^E$, cf. (3.37)), the extent of variation to be allowed would depend on the presence of vibrato, resulting in a chicken-and-egg problem for the task of vibrato classification. Furthermore, given the stability of the pitch features without signal preprocessing, a large performance increase would be required to outweigh the high additional computational effort from a practitioner's point of view.

7.2.2.2 Feature Relevance

As a perspective on feature relevance independent of the classification algorithm, two-sided Welch two-sample t-tests (assuming inequal variance) are performed on the features derived from F0 and its delta regression coefficients in the whole data set. These tests indicate whether their mean in the vibrato segments is significantly different from the mean in the non-vibrato segments. This strongly differs from the paired-sample t-test used for recognition evaluation in Section 2.1.3.2, since now two populations are modeled instead of a single test set. Here, the t-statistic is

$$\frac{\mu_{V,f} - \mu_{\overline{V},f}}{\sqrt{\sigma^2_{V,f}/N_V + \sigma^2_{\overline{V},f}/N_{\overline{V}}}}, \tag{7.1}$$

where $\mu_{V,f}$ and $\mu_{\overline{V},f}$ are the sample means of feature f in the vibrato and non-vibrato segments, $\sigma_{V,f}$ and $\sigma_{\overline{V},f}$ are the corresponding sample standard deviations, and N_V and $N_{\overline{V}}$ the corresponding numbers of instances. The p-values are not corrected for repeated measurements since only a ranking of the features is desired, which would be unchanged by Bonferroni correction and the like. This evaluation is restricted to F0 functionals due to their vastly superior performance in general (cf. the previous section).

For manual BP filtered as well as for unprocessed signals, the ten most discriminative functionals of F0 and their delta regression coefficients by their absolute t-statistic are shown in Tables 7.5a and 7.5b. Evidently, inter-quartile ranges of Δ F0 are particulary informative in both cases. Inter-quartile ranges of F0 itself, however, are only informative for manual BP filtering. This indicates that deltas are robust against pitch estimation errors while F0 itself is not: Apparently, due

Table 7.5: Relevance of functionals of pitch (F0) and F0 delta regression coefficients (Δ F0): p-values and t-statistics obtained on the evaluation database. IQR: inter-quartile ranges.

(a) no preprocessing

LLD	Functional	p	t
Δ F0	IQR 1–2	$\ll .001$	12.9
Δ F0	IQR 1–3	$\ll .001$	12.8
Δ F0	IQR 2–3	$\ll .001$	10.7
Δ F0	DFT coeff. 1	$< .001$	4.2
F0	DCT coeff. 2	$< .001$	- 3.8
F0	MCR	$< .001$	3.6
F0	DFT coeff. 5	$< .005$	- 3.2
Δ F0	Position of max.	$< .005$	- 3.2
Δ F0	Position of min.	$< .01$	- 2.8
F0	DFT coeff. 4	$< .01$	- 2.6

(b) manual BP

LLD	Functional	p	t
Δ F0	IQR 1–3	$\ll .001$	19.9
Δ F0	IQR 2–3	$\ll .001$	19.7
Δ F0	IQR 1–2	$\ll .001$	18.9
F0	DFT coeff. 7	$\ll .001$	13.2
F0	DFT coeff. 8	$\ll .001$	13.0
F0	DFT coeff. 9	$\ll .001$	9.1
F0	DFT coeff. 6	$\ll .001$	8.9
F0	Max-mean-dist.	$\ll .001$	6.8
F0	DFT coeff. 10	$\ll .001$	6.7
F0	IQR 1–3	$\ll .001$	6.4

to the Viterbi smoothing, the musical accompaniment causes systematic errors in pitch estimation rather than random fluctuations. Furthermore, it is well known that generally, percentile-based features such as IQR are robust against outliers caused by measurement errors. Concerning DFT coefficients, it is obvious that they have strong discriminative power after applying manual BP filtering: The most relevant coefficients 6–10 exactly correspond to vibrato rates from 4.7 to 7.8 Hz. In accordance with the automatic classification experiments, they are less discriminative when not doing preprocessing; furthermore, although DFT coefficients 1 (of Δ F0), 4 and 5 occur in the 10 most relevant features, their relation to vibrato is not immediately obvious: Coefficient 1 corresponds to 0.77 Hz, and the t-statistic of coefficients 4 and 5 is negative. Thus, these functionals (as well as the first DCT coefficient and mean crossing rate) apparently provide an useful, yet generic assessment of temporal evolution.

7.2.3 Conclusions

In a singer-independent evaluation on a real-life database spanning different musical genres from pop to opera, different approaches to robust automatic recognition of vibrato in recorded polyphonic music were investigated. As LLDs, F0 contours and other low-level descriptors were considered, and features were generated using segment-wise functionals. It turned out that while the conventional approach of using the F0 discrete spectrum dramatically suffers from pitch estimation errors due to multiple present sources of accompaniment, percentiles of the Δ coefficients of the pitch contour are highly robust, reaching up to 86.9 % accuracy in real-life

conditions without any signal enhancement. Signal enhancement could not improve the accuracy of vibrato recognition, which is in contrast to the other applications in this thesis. Instead, a combination of multi-condition training – in this case, training with mixtures of singing and accompaniment – and feature selection was most successful.

Future work should focus on dynamic spotting of vibrato singing in recorded polyphonic music, integrating the proposed classification framework into a fully automatic system; this could be achieved by using note on-set detection as a first stage, or dynamic modeling of LLDs or functionals by DRNNs.

7.3 Polyphonic piano transcription

In this section, a discriminative method for the transcription of polyphonic piano music is presented, which was introduced by the author and his colleagues in [226].

Transcription of polyphonic music is one of the key applications in MIR [9, 71], as it converts unstructured waveform data to a symbolic, musically meaningful representation. As such, it has striking similarities to ASR, yet is inherently a multi-source problem. Here, the problem of polyphonic music transcription is formulated as joint onset detection and multi-pitch estimation, where note onsets have to be detected along with the correct pitch.

A popular approach to multi-pitch estimation and polyphonic music transcription is based on NMF spectrogram decompositions, performing onset detection on the activations [1, 186, 201]. In several approaches, unsupervised NMF is used [1, 2, 186]. However, without further constraints it cannot be guaranteed that the NMF dictionary atoms resulting from this procedure have the desired musical meaning, i.e., that they represent different pitches of different instruments. As a result, the interpretation of the NMF decomposition, and hence transcription based on the activation matrix, can become challenging. Introducing musical constraints into NMF, such as in [13, 205], appears to be promising, yet from the results it seems that transcription using weakly supervised NMF techniques remains a notoriously difficult task [13, 205].

As an alternative, discriminative approaches [15, 123, 135, 149, 198] have been proposed, delivering most robust results [15]. In discriminative music transcription, a classifier is trained on positive and negative examples corresponding to signal frames where a given pitch is present or absent. This can be done in a 'one-versus-all' training paradigm for pitch-specific classifiers as will be considered below, based on the principle of Poliner and Ellis [149], or in a multi-task learning fashion as done by Böck and Schedl [15]. These paradigms avoid the combinatorial explosion reported by Emiya et al. [38] when all possible combinations of pitches are modeled as classes.

To combine the benefits of discriminative training with explicit signal decomposition and information reduction by NMF, here it is proposed to use the NMF activations computed from onset and non-onset parts as positive and negative data

points for the SVM classifier. In order to avoid matching unsupervisedly estimated dictionary atoms to pitches, supervised NMF is employed, where spectra corresponding to pitch-instrument pairs are pre-defined. As explained in Section 3.2.3, this has the additional advantage of being capable of low delay on-line processing, in contrast to the previous unsupervised or weakly supervised approaches. As in many previous studies [15, 38, 149, 186, 198], the evaluation is limited to piano music – the main reason being comparability, since for this task a large annotated evaluation database is available to the public (cf. Section 2.2.2.3).

7.3.1 Experiments

The method evaluated in this section is depicted as a flowchart in Figure 7.2. As a first step, the audio signal is converted to the time-frequency domain by either STFT or by the constant Q transformation (CQT) [167]. The time-frequency representation is then mapped to frame-wise note activations by means of supervised NMF with a dictionary of note spectra. From these 'raw' activations, 'high level' activation features are derived, which are then fed into a set of SVM classifiers that perform onset detection for each pitch-instrument pair. The frame level decisions of these classifiers are finally post-processed by a simple clustering method. Starting from this broad picture, let us now flesh out the details of each processing step.

7.3.1.1 Calculation of the NMF Activation Matrix

The magnitude of the time-frequency spectrogram (STFT or CQT) is computed, yielding a non-negative matrix $\hat{\mathbf{M}}$ (with observations in columns). This matrix is then subject to *data reduction*, i.e., 'down-sampling' by a factor N_c by merging time steps, yielding a matrix $\mathbf{M}$:

$$\mathbf{m}_t = \max\{\hat{\mathbf{m}}_{(t-1)N_c+1}, \ldots, \hat{\mathbf{m}}_{tN_c}\} \tag{7.2}$$

Then, NMF is applied to decompose $\mathbf{M}$ into the two factors $\mathbf{W}$ and $\mathbf{H}$, of which the first one represents a note dictionary and the second one the activity of notes over time. A supervised NMF approach (cf. Section 3.2.3) is followed, where the matrix $\mathbf{W}$ is pre-computed in a training phase according to the following section. During the transcription phase, the matrix $\mathbf{H}$ is calculated by iteratively minimizing the generalized Kullback-Leibler divergence (3.6) starting from a random solution for $\mathbf{H}$, until the solution has converged or a maximum number of K iterations has been reached.

7.3.1.2 Dictionary Matrix Estimation

In order to build the matrix $\mathbf{W}$, NMF is exploited in a weakly supervised fashion as follows. It is assumed that there are recordings of isolated notes available for

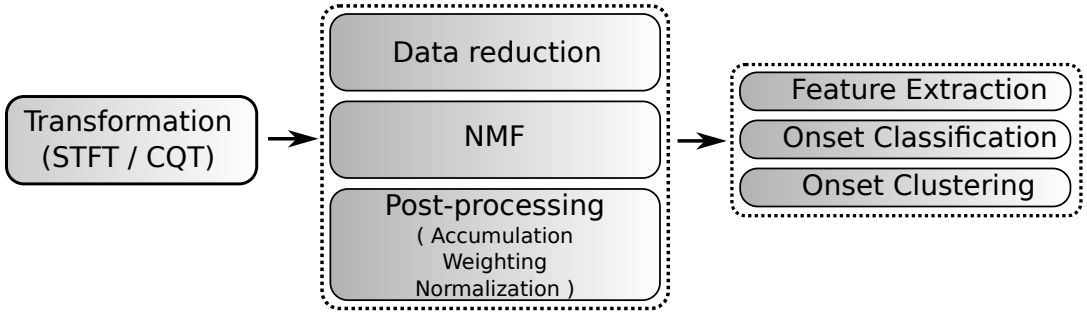


Figure 7.2: Overview of the proposed transcription method, consisting of low-level spectral feature extraction, calculation and post-processing of NMF activation features, classification by support vector machines and decision level post-processing.

the instrument that is to be transcribed. Then, 'characteristic spectra' for each of these notes are calculated from the spectrograms $\mathbf{S}^l$, where l is the pitch index. Treating notes of different pitches as sources in the supervised NMF framework (cf. Section 3.2.3), the most straightforward approach to dictionary learning is to apply unsupervised NMF on $\mathbf{S}^l$, using $R = 1$ and keeping the first factor (i. e., a column vector) as dictionary atom for pitch l. However, since the focus is on detecting the onsets of the notes, it seems useful to extract spectra representing the attack and the decay phases of a note (*onset sharpening*). This is motivated by the observations made by Ewert and Müller [40] in the context of weakly supervised NMF on piano music. In the presented approach, the $1 \times T$ activation matrix obtained by unsupervised NMF with $R = 1$ is considered as a row vector $\hat{\mathbf{h}}$, of which the maximum h^* and its position t^* are computed. Then, the frame index t' is set to the first frame after the maximum ($t' > t^*$) where

$$\hat{h}_{t'} < \sigma \cdot h^*. \tag{7.3}$$

Then, an 'onset atom' is obtained by applying unsupervised NMF only on the first t' columns of $\mathbf{S}^l$. Analogously, from the rest of $\mathbf{S}^l$ a 'decay atom' is obtained. The usage of onset and decay atoms will be evaluated later. Finally, the matrix $\mathbf{W}$ is simply the column-wise concatenation of the atom(s) estimated per pitch l, normalized to unity L2 norm per column.

7.3.1.3 Activation Post-Processing

Before performing onset detection on the NMF activations, a three-step post-processing stage (cf. Figure 7.2) is applied. First, the activations are *accumulated* for each pitch l in case that multiple dictionary atoms per pitch are used. The outcome of this step will be denoted by $\mathbf{h}(t)$ in the ongoing, and its components by $h_l(t)$. Second, since it was found that certain pitches had overall higher activations

than others despite the normalized atoms in $\mathbf{W}$, an 'inverse document frequency' *weighting* is applied to the activation vectors per time step:

$$h_l(t) \leftarrow h_l(t)/\overline{h}_l, \forall p \tag{7.4}$$

where $\overline{h}_l$ is the average activation of pitch l when NMF is applied to the training data. Finally, $\mathbf{h}(t)$ is *normalized* by its L1 norm adding a constant ϵ:

$$\mathbf{h}(t) \leftarrow \mathbf{h}(t)/(\epsilon + |\mathbf{h}(t)|_1) \tag{7.5}$$

The constant ϵ is needed because a naïve normalization would yield erroneous activations for segments without onsets. Subsequently, $\epsilon = 6$ will be used.

7.3.1.4 Activation Feature Extraction

In a baseline approach, a single activation difference feature is used per pitch. Precisely, defining $T_{\text{span}}(t)$ as the set of frame indices corresponding to the span in milliseconds after the frame with index t, this feature is computed as

$$g_1(t) = h_l(t) - \max_{t' \in T_{-50}(t)} h_l(t'), \tag{7.6}$$

i. e., the difference of the current activation to the maximum activation within the last 50 milliseconds.

Besides, a multi-dimensional feature set is considered where

$$\begin{aligned}
g_2(t) &= \max_{t' \in T_{50}(t)} h_l(t') - \min_{t' \in T_{-50}(t)} h_l(t'), && (7.7)\\
g_3(t) &= h_l(t) - \min_{t' \in T_{-100}(t)} h_l(t'), && (7.8)\\
g_4(t) &= \max_{t' \in T_{100}(t)} h_l(t'), && (7.9)\\
g_5(t) &= \min_{t' \in T_{-100}(t)} h_l(t'), \text{and} && (7.10)\\
g_6(t) &= \max_{t' \in T_{250}(t)} h_l(t') - h_l(t) && (7.11)
\end{aligned}$$

are added.

7.3.1.5 Classification and Onset Detection

For each pitch l, an SVM classifier is trained on set of feature vector – label pairs $\{(\mathbf{g}(t), y_t)\}_t$. For the multi-dimensional feature set, $\mathbf{g}(t) = (g_1(t), \ldots, g_6(t))^\intercal$. In case of the single-dimensional feature 'set', an SVM is used as well for consistency – this corresponds to a threshold decision on activation differences, where the threshold is optimized by a maximum margin criterion on the training data.

As positive examples for the SVM, feature vectors inside a detection window of 100 ms around the ground truth onsets are used that have a maximum acceleration of the activation, since rising activations indicate onsets and an attack phase may include several points of rising activations. Mathematically, $(\mathbf{g}(t^*), 1)$ is added to the training set with $t^* = \arg\max_t \{h_l(t) - h_l(t-1) - (h_l(t-1) - h_l(t-2))\}$.

Negative examples are taken from intervals outside the detection window. To reduce the redundancy caused by many similar data points representing silence, all data points yielding an activation difference less than the average are discarded with a probability close to one. Still, the above procedure yields a large training set. For example, the training set introduced in Section 7.3.1.6 corresponds to 14 h of music, resulting in 2.5 million data points. For efficiency reasons, and since usage of non-linear SVM kernels did not significantly improve transcription results, classifier training is done with LibLinear [45], providing an efficient method to train linear SVM.

After obtaining a classifier decision for each time step, the onset timing is computed by a 'clustering' step on the 'raw' decisions. A cluster is defined by a set of positively classified data points, such that there is no negatively classified data point between two points of the set, and the two neighboring data points around this set are classified as negative. An onset is predicted at the mid-point of each cluster.

7.3.1.6 Evaluation

As training and evaluation data, the MAPS family of databases as introduced in Section 2.2.2.3 is used. For NMF, an instrument-dependent $\mathbf{W}$ matrix is built using isolated notes in the training sets of the databases; in case of the MIDI database, some missing isolated notes were synthesized using the above-mentioned sound font. Onset classifiers are trained on the activation features computed from the union of the training and validation sets.

As the main evaluation measure, accuracy in the form introduced by Dixon [31] was used, which was later picked up by Böck and Schedl [15] for polyphonic transcription. Here, accuracy is defined as TP / (TP + FP + FN), where TP is the number of true positives, i. e., notes identified with the correct pitch within a symmetric window around the ground truth onset time, FP is the number of false positives (i. e., a note of the wrong pitch is detected), and FN is the number of false negatives (i. e., a note is missing in the transcript). This is a somewhat 'harsh' measure, as it counts substitutions, i. e., pitch errors, twice (one false negative for the correct pitch and one false positive for the incorrect pitch). Additionally, the standard F-measure is used, which is the harmonic mean of recall (TP / (TP + FN)) and precision (TP / (TP + FP)) following the notion of TP, FN and FP introduced above. Following Poliner and Ellis [149], the window of correct detection is set to 100 ms (ground truth timing ± 50 ms).

As a rule of thumb for the observed ranges of accuracy, accuracy improvements

Table 7.6: Threshold detection (1-dimensional SVM using $g_1(t)$) vs. SVM using 6-dimensional features ($\mathbf{g}(t)$), $N_c = 2$: Accuracy and F-measure (Fm).

[%]	MIDI		MAPS MIDI		MAPS D	
	Acc	Fm	Acc	Fm	Acc	Fm
1-dim.	62.4	76.8	72.5	84.0	45.1	62.2
6-dim.	**74.3**	**85.3**	**79.4**	**88.5**	**68.0**	**81.0**

of more than one percent are statistically significant at the 0.1 % level according to a one-tailed z-test (cf. Section 2.1.3.1) with the number of instances corresponding to the number of notes in the data set.

7.3.1.7 Parameterization

For the STFT, a window size of 3 072 samples and a step size of 512 samples are used as in a previous study using spectral features [149]. For the CQT, the toolbox presented in [167] is utilized, using 24 bins per octave over seven octaves and the default parameters for window size and step size. NMF is applied for up to $K = 200$ iterations.

7.3.2 Results

In a first step, the 6-dimensional feature set $\mathbf{g}(t)$ is evaluated against a simple threshold decision (1-dimensional SVM) based on the NMF pitch activation differences $g_1(t)$. The accuracies and F-measures resulting from either method are shown in Table 7.6. It is observed that for all three of the data sets, both measures are drastically increased by the proposed 6-dimensional feature set. This especially holds for the MAPS Disklavier set of real piano recordings, where 22.9 % absolute accuracy and 18.8 % F-measure are gained. As generally very similar trends were observed for accuracy and F-measure in the evaluations, the focus will be laid on accuracy below.

Next, the proposed merging of frames in the spectrogram matrix by the maximum operator (7.2) is evaluated. From the accuracies displayed in Figure 7.3, it can be seen that this technique consistently improves the performance over the baseline (no merging, $N_c = 1$), and best results are achieved by setting $N_c = 4$.

Next, in Figure 7.4, the influence of the spectral representation (CQT or STFT) is evaluated. There does not seem to be any improvement by using CQT instead of STFT, even if the factor N_c is increased, respecting the fact that the frame shift chosen for CQT is larger than for STFT. In fact, the accuracy is significantly lower when using CQT rather than STFT spectra as input for the NMF step. This is probably because the linear scaling of frequency bins in the STFT domain provides better discrimination of pitches from different octaves. In the ongoing, STFT features will be used.

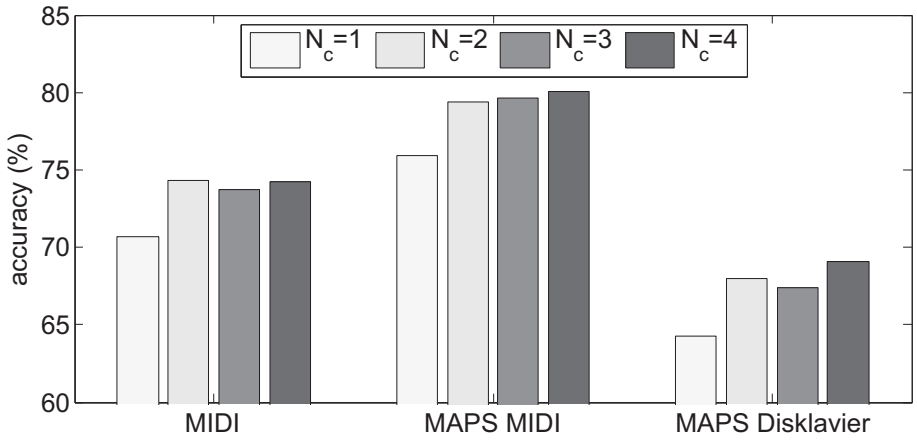


Figure 7.3: Effect of frame merging in the $\hat{\mathbf{M}}$ matrix of polyphonic spectra: Accuracy for different values of N_c (7.2), on test sets $1 \cup 2$.

Finally, the method is evaluated using different dictionary sizes ($R = 1, 2$ atoms per pitch), and optionally using onset sharpening ($\sigma = 0.8$) to subdivide the training notes into attack and decay phase. For $\sigma = 0.8$ and $R = 1$, only the attack (onset) atoms are used. Results of the evaluation on the testing 1 set (for comparability with [15]) are shown in Figure 7.5. If onset sharpening is not used, the number of atoms in the NMF dictionary only changes the outcome on the MIDI dataset (by 1 % absolute accuracy). Notably, the standard unsupervised NMF approach is outperformed by the proposed onset sharpening method, which delivers best results on each database. This indicates the usefulness of prior knowledge in the NMF dictionary learning process. However, on MAPS MIDI, results can only be improved over the baseline ($R = 1$, no onset sharpening) if the dictionary size (and thereby the computational complexity) is increased by including the decay atoms as well. Note that the dictionary size does not influence the number of features in classification, so that the performance improvement by using $R = 2$ cannot be attributed simply to having more features.

Let us now compare the obtained results (with $R = 2, \sigma = 0.8$) to the state-of-the-art in terms of accuracy, and display the results in Table 7.7. On the MIDI dataset, the proposed NMF+SVM method evidently delivers significantly higher accuracy (+ 14.8 % abs.) than the SVM method based on spectral features proposed by Poliner and Ellis [149]. However, the method is outperformed by the boosting and BLSTM approaches proposed in [15, 198]. It can be argued that this is due to the independent training of pitch-specific classifiers, and the presented method could be improved by exploiting the correlations of pitch activations, such as chord structures in tonal music. On the MAPS data set, the presented method is evidently superior to the results obtained by Böck and Schedl [15] (+2.3 % abs. on MIDI and +8.6 % on real piano). Yet, these results are not fully comparable since Böck and

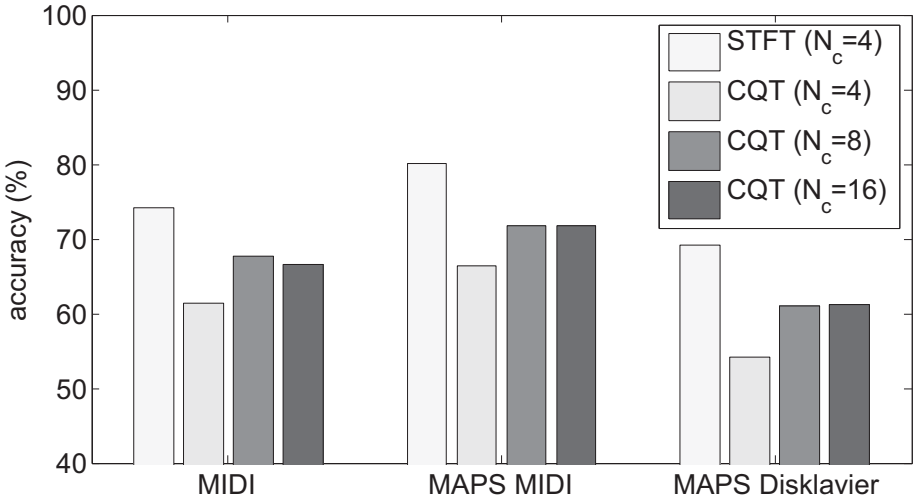


Figure 7.4: Effect of different spectral features: Accuracy using STFT vs. Constant-Q-Transform (CQT), for different values of N_c (7.2), on test sets $1 \cup 2$.

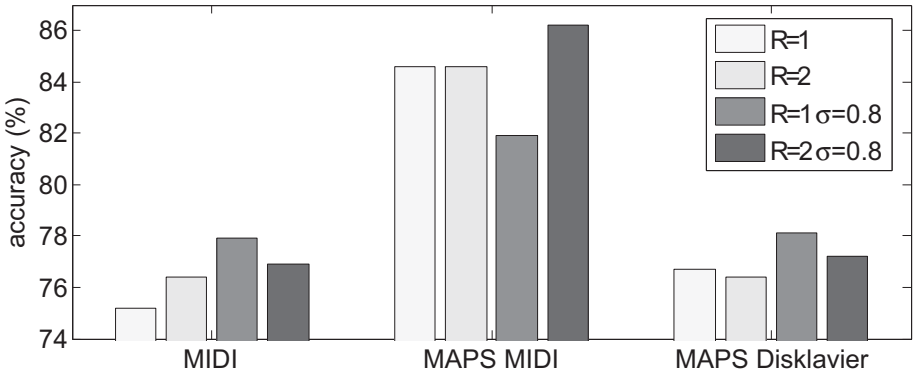


Figure 7.5: Effect of dictionary size (R atoms per pitch) and onset sharpening ($\sigma = 0.8$, cf. (7.3)) on accuracy; evaluation on test set 1.

Schedl [15] use a 'closed set' experimental setup where training data is collected from all pianos in the databases, and it is not known which piano is played in which test instance, whereas the presented results, as the ones of Poliner and Ellis [149] and v. d. Boogaart and Lienhart [198], are obtained in an instrument-dependent setup.

Note, however, that there are reasons to believe that NMF provides a convenient and effective method to perform transcription in a closed set setup, by building a joint $\mathbf{W}$ matrix of the instrument-dependent dictionaries, performing supervised NMF and then selecting the dictionary with the highest overall activation for transcription. In a preliminary experiment, 83 % average recall of the ten pianos in the test databases could be achieved by deciding for the instrument whose dictionary had the highest activation sum, respecting instrument-wise group sparsity constraints on the activations in analogy to the method proposed by Hurmalainen et al. [90] for

Table 7.7: Comparison of algorithms on testing 1 set. [1]: Instrument-dependent training; [2]: Multi-instrument (closed-set) training.

Accuracy [%]	MIDI	MAPS MIDI	MAPS D
SVM[1] [149]	62.3	–	–
BLSTM[2] [15]	88.9	84.0	68.7
Boosting[1] [198]	87.4	–	–
Proposed (NMF+SVM)[1]	77.1	86.3	77.1

speaker identification.

7.3.3 Conclusions

An effective and efficient method for the task of polyphonic piano transcription was presented. In line with the objectives of this thesis, it was shown that source separation motivated features extracted by NMF lead to better performance in discriminative one-versus-all classification by SVM compared to unprocessed spectral features ('multi-condition training'). The proposed method delivered state-of-the-art results on three test databases comprising synthesized MIDI as well as real piano recordings. In the future, it should be investigated if the NMF feature extraction method can be used advantageously in combination with LSTM-RNN based pitch onset classification.

Part IV

Concluding remarks

8

Concluding remarks

I dread success. To have succeeded is to have finished one's business on earth [...] I like a state of continual becoming, with a goal in front and not behind. – George Bernard Shaw

8.1 Summary

The research presented in this thesis was oriented on two main hypotheses: (a) that an explicit source separation step in the process of multi-source recognition is beneficial for recognition performance, and (b) that formulating source separation as a recognition task improves separation performance over unsupervised training (cf. Section 1.3).

The first hypothesis could be verified in the real-life uses cases of noise-robust speech recognition, singer characterization, and polyphonic music transcription. The word error rates achieved by state-of-the-art DNN-HMM acoustic models (multi-stream and bottleneck) with multi-condition training could be significantly improved by including NMF-based separation of speech and noise sources (Sections 5.1.1 and 5.1.2). Similar results were obtained in singer characterization, where the determination of singer age and gender could be enhanced by separation of the leading voice via semi-supervised NMF (Section 7.1). Furthermore, discriminatively trained automatic music transcription systems could be improved by including NMF-based features, instead of simple spectral features (Section 7.3).

When applying DNN-based feature enhancement in GMM acoustic models, it was found that the results in noise- and reverberation-robust ASR are similar to state-of-the-art connectionist approaches for acoustic modeling (Section 5.2). In a realistic reverberant speech recognition task (Section 6.2), DNN-based feature enhancement could only yield limited gains when combined with DNN-based acoustic modeling. This is interesting as on the contrary, NMF-based front-ends appeared to be complementary to DNN acoustic models. Another negative result in the context

of the first hypothesis is that in the case of singing style recognition (vibrato, Section 7.2), information loss due to source separation outweighed the benefits, as the task itself is tied to the fine-grained structure of the spectrum.

The second hypothesis regarding the formulation of source separation as a recognition task could be verified by showing that supervised discriminative training of source separation using state-of-the-art DNN models outperforms state-of-the-art weakly supervised methods such as NMF (Section 4.2). Furthermore, it was observed that the transfer of the supervised training scheme to NMF leads to significant performance gains also for NMF (Section 4.2). In the related application of ASR feature enhancement, it could also be demonstrated that stereo training using simulated parallel data is applicable to real noisy and reverberant speech (Section 6.1), and there is initial evidence for stereo-training based LSTM-RNN source separation being able to generalize to real-life data (Section 5.3).

8.2 Outlook

Novel architectures for deep learning In the light of the formulation of NMF as a deep learning method using supervised training [81, 216], and the results regarding NMF and DNN enhancement discussed above, it will be highly interesting to investigate a combination of NMF 'layers' with traditional (e.g., rectified linear or sigmoid) as well as recurrent or LSTM layers. Evidence for the complementarity of supervised NMF pre-processing and DNN acoustic modeling was also found by Geiger et al. [53], but they did not yet consider backpropagation of the DNN acoustic modeling error to the NMF part. The closer integration of NMF and DNN would serve to integrate model-based constraints into deep learning for multi-source recognition, possibly leading to better generalization especially with scarce training data, and to model-inspired techniques for on-line adaptation. For example, the semi-supervised NMF method presented in Section 4.1 yielded robust results with less than a minute of training data per speaker and on-line learning of noise models. It will be highly interesting to consider schemes similar to semi-supervised NMF in the context of the deep learning framework.

Better models vs. more data A large part of the contributions of this thesis can be subsumed under the umbrella of advanced modeling and model training. Yet, in pattern recognition, it is a common belief that 'there is no data like more data', and this belief is verified particularly in the case of industrial ASR applications, where drastic improvements by DNN acoustic models over the previous state-of-the-art GMM approaches can be obtained if and only if a large amount of training data are available [82]. This shows the importance of performing evaluations for varying, and particularly large, data set sizes. Conversely, especially in the noise-robustness and source separation research communities, a significant amount of research effort

is devoted to improving results with a fixed, usually small sized, amount of data, with frequently used data sets such as the CHiME and REVERB databases being comparable in size to the ones used for ASR research in the early 1990s [147]. This is not to devalue these benchmarks, as coping with scarce data, such as by model-based approaches like NMF, is clearly an interesting research problem that is of interest in many domains. However, from a practitioner's point of view, it is not clear whether the benefits of more complex learning models, such as combinations of NMF front-ends and DNN acoustic models, or better training schemes, such as the two-stage discriminative training process from Section 4.2, would still be significant for industrial training set sizes, such as the ones used for voice search applications with thousands of hours of speech [27]. Formally, in terms of (3.1), source separation can be thought of as a pre-processing step g_1 that can be subsumed by a powerful enough model z – in this case, it is not clear that it would still be beneficial to model z as a composition incorporating g_1: There could be a generic model which performs better, at least when given enough training data. A somewhat encouraging result for the proponents of model-inspired approaches is that LSTM-based acoustic modeling, which incorporates the constraint of audio processing requiring the storage of long-term context, improves over simpler DNN models in industrial ASR application [163]. This is notable as LSTM has been shown to be beneficial for source separation in this thesis (Sections 4.2, 5.3), but of course its actual benefit for source separation with large training sets remains to be evaluated.

Synergies between speech and music analysis As laid out in [221], music analysis has co-shaped many of the robust ASR techniques presented in this thesis, and vice versa. Considering the contributions of this thesis, obviously it will be interesting to evaluate discriminative and DNN-based source separation on a large scale in MIR tasks; yet also novel approaches for music analysis can be thought of, which are inspired by the results of this thesis. For instance, stereo-training based mappings of musically motivated features (e.g., timbre, pitch, vibrato) extracted from polyphonic music to features from single tracks could be exploited, in analogy to ASR feature enhancement. Both for feature 'enhancement' and source separation scenarios, stereo training seems even more appealing for MIR than for ASR, because in music recordings, 'stem tracks' (studio recordings of separated sources) are often available [14], along with professionally produced convolutive mixtures.

Towards holistic scene understanding Schuller [173] introduced 'intelligent audio analysis' and formulated the vision of holistic scene understanding, for example, to simultaneously analyze a mixture of speech and music in terms of the speaker's emotion, spoken words, musical genre, as well as the instruments played. This thesis contributes to a realization of this vision by methods for multi-source recognition, for example, separating speech and music (Section 4.1), and singer characterization

and polyphonic transcription (Sections 7.1 and 7.3). In using supervised approaches for holistic scene understanding, one needs to take into account the combinatorial explosion when multiple recognition tasks are learnt on mixtures of multiple sources, as the number of models to be trained in a traditional multi-condition training setup is exponential in the number of potential sources and polynomial in the number of possible recognition tasks that can be performed on a source. Therefore, it will likely be necessary to consider an effective combination of unsupervised methods, e.g., for source separation and acoustic clustering, and advanced supervised training schemes with a focus on generalization, in order to come closer to this goal.

Acronyms

ASR	Automatic Speech Recognition
ATLAS	Automatically Tuned Linear Algebra Software
BDRNN	Bidirectional Deep Recurrent Neural Network
BLAS	Basic Linear Algebra Subroutines
BLSTM	Bidirectional Long Short-Term Memory
bMMI	boosted Maximum Mutual Information
BPTT	Backpropagation Through Time
BRNN	Bidirectional Recurrent Neural Network
CHiME	Computational Hearing in Multisource Environments
CMN	Cepstral Mean Normalization
CPU	Central Processing Unit
CUBLAS	Compute Unified Basic Linear Algebra Subroutines
CUDA	Compute Unified Device Architecture
CURRENNT	CUda RecurREnt Neural Network Toolkit
DCT	Discrete Cosine Transformation
DFT	Discrete Fourier Transformation
DNN	Deep Neural Network
DRNN	Deep Recurrent Neural Network

EM	Expectation-Maximization
fMLLR	feature-space Maximum Likelihood Linear Regression
GPU	Graphics Processing Unit
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
ICA	Independent Component Analysis
LDA	Linear Discriminant Analysis
LSTM	Long Short-Term Memory
MAP	Maximum-A-Posteriori
MBR	Minimum Bayes Risk
MCT	Multi-Condition Training
MFCC	Mel-Frequency Cepstral Coefficient
MIR	Music Information Retrieval
ML	Maximum Likelihood
MLP	Multi-Layer Perceptron
NIST	National Institute of Standards and Technology
NMF	Non-Negative Matrix Factorization
PCA	Principal Component Analysis
PDF	Probability Density Function
PLP	Perceptual Linear Prediction
REVERB	Reverberant Voice Enhancement and Recognition Benchmark
RIR	Room Impulse Response
RNN	Recurrent Neural Network
RT	reverberation time

RTF	Real-Time Factor
SAT	Speaker Adaptive Training
SAR	Source-to-Artifacts Ratio
SDR	Source-to-Distortion Ratio
SNMF	Sparse NMF
SNR	Signal-to-Noise Ratio
SIR	Source-to-Interference Ratio
SVM	Support Vector Machine
STC	Semi-Tied Covariance
STFT	Short-Term Fourier Transformation
TAU	Tel Aviv University
TUM	Technische Universität München
WA	Word Accuracy
WER	Word Error Rate

List of Symbols

β	Divergence parameter
$\mathcal{B}$, $\lvert\mathcal{B}\rvert$	(Mini-)batch, batch size
B	Number of (Mel) frequency bands
C	Number of channels
D	Distance / divergence / cost function
$E_{\mathcal{T}}$	Training set error
f	Feature index
F	Number of features
g	Feature extractor
η	Gradient descent step size (learning rate)
$\mathcal{H}$	Activation function
$\mathbf{H}$	Activation matrix
id	Identity function (id: $x \mapsto x$)
k	Layer index (e.g., feature extraction, NMF update or DNN layer)
K	Number of layers
l	Source index
L	Number of classes
$\mathcal{M}$	Mixing process
$\overline{\mathcal{M}}$	Unmixing function
$m(\tau)$	Mixture time-domain signal
$\mathbf{m}_t$	Mixture features at time step t

$\mathbf{M}$	Mixture feature matrix (e.g., spectrogram)
p	Likelihood, p-value (in statistical testing)
q	Training epoch
R	NMF dictionary size (number of NMF components)
S	Number of sources
$s_l(\tau)$	Source l time-domain signal
$\hat{s}_l(\tau)$	Estimated source l time-domain signal
$\mathbf{s}_t^l$	Source l features at time step t
$\hat{\mathbf{s}}_t^l$	Estimated source l features at time step t
$\mathbf{S}^l$	Source l feature matrix (e.g., spectrogram)
$\hat{\mathbf{S}}^l$	Estimated source l feature matrix (e.g., spectrogram)
τ	Sample index (in a discrete-time signal)
t	Time step
T	Number of time steps / short-term feature windows
T_L	Number of left context frames
T_R	Number of right context frames
$\mathcal{T}$	Training set
$\mathbf{W}$	Weight matrix (e.g., DNN connections, NMF dictionary)
$x(\tau)$	Acoustic time-domain signal
$\mathbf{x}_t$	Acoustic features at time step t
$\mathbf{X}$	Acoustic feature matrix (e.g., spectrogram)
y	Recognition function
$\mathbf{y}$	Value of recognition function (e.g., DNN outputs)
y^*	Labeling function, training targets

Bibliography

[1] S. A. Abdallah and M. D. Plumbley, "Polyphonic transcription by non-negative sparse coding of power spectra," in *Proceedings of the 5th International Society for Music Information Retrieval Conference (ISMIR)*. Barcelona, Spain: International Society for Music Information Retrieval, 2004, pp. 318–325.

[2] ——, "Unsupervised analysis of polyphonic music by sparse coding," *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 179–196, 2006.

[3] N. Amir, O. Michaeli, and O. Amir, "Acoustic and perceptual assessment of vibrato quality of singing students," *Biomedical Signal Processing and Control*, vol. 1, no. 2, pp. 144–150, 2006.

[4] T. Anastasakos, J. McDonough, and J. Makhoul, "Speaker adaptive training: A maximum likelihood approach to speaker normalization," in *Proceedings of the 32nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Honolulu, HI, USA: IEEE, 2007, pp. 1043–1046.

[5] D. Baby, T. Virtanen, T. Barker, and H. Van hamme, "Coupled dictionary training for exemplar-based speech enhancement," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, 2014, pp. 2907–2911.

[6] M. H. Bahari, M. McLaren, H. Van hamme, and D. V. Leeuwen, "Age estimation from telephone speech using i-vectors," in *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Portland, OR, USA: ISCA, 2012, no pagination.

[7] J. P. Barker, E. Vincent, N. Ma, H. Christensen, and P. D. Green, "The PASCAL CHiME speech separation and recognition challenge," *Computer Speech and Language*, vol. 27, no. 3, pp. 621–633, 2013.

[8] E. Battenberg and D. Wessel, "Accelerating non-negative matrix factorization for audio source separation on multi-core and many-core architectures," in *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*. Kobe, Japan: International Society for Music Information Retrieval, 2009, pp. 501–506.

[9] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri, "Automatic music transcription: Breaking the glass ceiling," in *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*. Porto, Portugal: International Society for Music Information Retrieval, 2012, pp. 379–384.

[10] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[11] J. Bergmann, "Implementation of recurrent neural networks on multi-core architectures," Master's thesis, Technische Universität München, Munich, Germany, 2013.

[12] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, Austin, TX, USA, 2010, no pagination.

[13] N. Bertin, R. Badeau, and E. Vincent, "Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 18, no. 3, pp. 538–549, 2010.

[14] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "MedleyDB: A multitrack dataset for annotation-intensive MIR research," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*. Taipei, Taiwan: International Society for Music Information Retrieval, 2014, in press.

[15] S. Böck and M. Schedl, "Polyphonic piano note transcription with recurrent neural networks," in *Proceedings of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan: IEEE, 2012, pp. 121–124.

[16] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 27, no. 2, pp. 113–120, 1979.

[17] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in *Proccedings of the 29th International Conference on Machine Learning (ICML)*. Edinburgh, Scotland: International Machine Learning Society, 2012, pp. 1159–1166.

[18] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Berlin / Heidelberg: Springer Science & Business Media, 1994.

[19] F. Burkhardt, R. Huber, and A. Batliner, "Application of speaker classification in human machine dialog systems," in *Speaker Classification I: Fundamentals, Features, and Methods*, ser. Lecture Notes in Computer Science, C. Müller, Ed. Berlin / Heidelberg: Springer, 2007, vol. 4343, pp. 174–179.

[20] M. Cernanský, "Training recurrent neural network using multistream extended Kalman filter on multicore processor and CUDA enabled graphic processor unit," in *Proceedings of the 19th International Conference on Artificial Neural Networks (ICANN)*, ser. Lecture Notes in Computer Science, C. Alippi, M. M. Polycarpou, C. Panayiotou, and G. Ellinas, Eds., vol. 5769. Limassol, Cyprus: Springer, 2009, pp. 381–390.

[21] H. Christensen, J. Barker, N. Ma, and P. Green, "The CHiME corpus: a resource and a challenge for computational hearing in multisource environments," in *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Makuhari, Japan: ISCA, 2010, pp. 1918–1921.

[22] A. Cichocki, R. Zdunek, A. H. Phan, and S.-I. Amari, *Nonnegative Matrix and Tensor Factorizations*. John Wiley & Sons, 2009.

[23] M. Cooke, J. R. Hershey, and S. J. Rennie, "Monaural speech separation and recognition challenge," *Computer Speech and Language*, vol. 24, pp. 1–15, 2010.

[24] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 120, no. 5, pp. 2421–2424, 2006.

[25] R. Daido, A. Ito, M. Ito, and S. Makino, "Automatic evaluation of singing enthusiasm for karaoke," *Computer Speech and Language*, vol. 28, no. 2, pp. 501–517, 2014.

[26] A. de la Torre, A. M. Peinado, J. C. Segura, J. L. Perez-Cordoba, M. C. Benitez, and A. J. Rubio, "Histogram equalization of speech representation for robust speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 355–366, 2005.

[27] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le, and A. Y. Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, Inc., 2012, vol. 25, pp. 1223–1231.

[28] M. Delcroix, T. Nakatani, and S. Watanabe, "Static and dynamic variance compensation for recognition of reverberant speech with dereverberation preprocessing," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 2, pp. 324–334, 2009.

[29] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 8599–8603.

[30] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Computation*, vol. 10, pp. 1895–1923, 1998.

[31] S. Dixon, "On the computer recognition of solo piano music," in *Proceedings of the Australasian Computer Music Conference*, 2000, pp. 31–37.

[32] L. Donati, "Parallel GPU-Based Implementation of Evolutionary Artificial Neural Networks (in Italian)," Master's thesis, Università degli Studi di Parma, 2010.

[33] C. Drioli and R. D. Federico, "Toward an integrated sound analysis and processing framework for expressiveness rendering," in *Proceedings of the International Computer Music Conference (ICMC)*, Ann Arbor, MI, USA, 1998, pp. 175–178.

[34] Z. Duan, G. J. Mysore, and P. Smaragdis, "Speech enhancement by online non-negative spectrogram decomposition in non-stationary noise environments," in *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Portland, OR, USA: ISCA, 2012, no pagination.

[35] J.-L. Durrieu, G. Richard, and B. David, "An iterative approach to monaural musical mixture de-soloing," in *Proceedings of the 34th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Taipei, Taiwan: IEEE, 2009, pp. 105–108.

[36] J.-L. Durrieu, G. Richard, B. David, and C. Févotte, "Source/filter model for unsupervised main melody extraction from polyphonic audio signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 564–575, 2010.

[37] J. Eggert and E. Körner, "Sparse coding and NMF," in *Proceedings of the 2004 IEEE International Joint Conference on Neural Networks (IJCNN)*, vol. 4, Dalian, China, 2004, pp. 2529–2533.

[38] V. Emiya, R. Badeau, and B. David, "Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 18, no. 6, pp. 1643–1654, 2010.

[39] T. Esch and P. Vary, "Efficient musical noise suppression for speech enhancement system," in *Proceedings of the 34th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Taipei, Taiwan: IEEE, 2009, pp. 4409–4412.

[40] S. Ewert and M. Müller, "Using score-informed constraints for NMF-based source separation," in *Proceedings of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan: IEEE, 2012, pp. 129–132.

[41] S. Ewert, B. Pardo, M. Müller, and M. D. Plumbley, "Score-informed source separation for musical audio recordings," *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 116–124, 2014.

[42] F. Eyben, F. Weninger, F. Groß, and B. Schuller, "Recent developments in openSMILE, the Munich open-source multimedia feature extractor," in *Proceedings of the 21st ACM International Conference on Multimedia*. Barcelona, Spain: ACM, 2013, pp. 835–838, (Honorable Mention (2nd place) in the ACM MM 2013 Open-source Software Competition).

[43] F. Eyben, F. Weninger, and B. Schuller, "Affect recognition in real-life acoustic conditions – a new perspective on feature selection," in *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Lyon, France: ISCA, 2013, pp. 2044–2048.

[44] F. Eyben, F. Weninger, S. Squartini, and B. Schuller, "Real-life voice activity detection with LSTM recurrent neural networks and an application to Hollywood movies," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 483–487.

[45] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[46] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: with application to music analysis," *Neural Computation*, vol. 21, no. 3, pp. 793–830, 2009.

[47] W. M. Fisher, G. R. Doddington, and K. M. Goudie-Marshall, "The DARPA speech recognition research database: Specifications and status," in *Proceedings of the DARPA Workshop on Speech Recognition*, 1986, pp. 93–99.

[48] S. Furui, "Speaker-independent isolated word recognition based on emphasized spectral dynamics," in *Proceedings of the 11th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Tokyo, Japan: IEEE, 1986, pp. 1991–1994.

[49] M. J. F. Gales and Y. Q. Wang, "Model-based approaches to handling additive noise in reverberant environments," in *Proceedings of the IEEE Workshop on Hands-free Speech Communication and Microphone Arrays (HSCMA)*, Edinburgh, UK, 2011, pp. 121–126.

[50] M. J. F. Gales, "Semi-tied covariance matrices for hidden Markov models," *IEEE Transactions on Speech and Audio Processing*, vol. 7, pp. 272–281, 1999.

[51] J. T. Geiger, Z. Zhang, F. Weninger, B. Schuller, and G. Rigoll, "Robust speech recognition using Long Short-Term Memory recurrent neural networks for hybrid acoustic modelling," in *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Singapore, Singapore: ISCA, 2014, in press.

[52] J. T. Geiger, F. Weninger, A. Hurmalainen, J. F. Gemmeke, M. Wöllmer, B. Schuller, G. Rigoll, and T. Virtanen, "The TUM+TUT+KUL approach to the CHiME Challenge 2013: Multi-stream ASR exploiting BLSTM networks and sparse NMF," in *Proceedings of the 2nd International CHiME Workshop on Machine Listening in Multisource Environments*, Vancouver, Canada, 2013, pp. 25–30, (Best paper award).

[53] J. T. Geiger, F. Weninger, J. F. Gemmeke, M. Wöllmer, B. Schuller, and G. Rigoll, "Memory-enhanced neural networks and NMF for robust ASR," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 6, pp. 1037–1046, 2014.

[54] D. Gelbart and N. Morgan, "Evaluating long-term spectral subtraction for reverberant ASR," in *Proceedings of the 2001 IEEE Workshop on Automatic*

Speech Recognition and Understanding (ASRU). Madonna di Campiglio, Italy: IEEE, 2001, pp. 103–106.

[55] J. F. Gemmeke and T. Virtanen, "Noise robust exemplar-based connected digit recognition," in *Proceedings of the 35th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* Dallas, TX, USA: IEEE, 2010, pp. 4546–4549.

[56] J. F. Gemmeke, A. Hurmalainen, T. Virtanen, and Y. Sun, "Toward a practical implementation of exemplar-based noise robust ASR," in *Proceedings of the 19th European Signal Processing Conference (EUSIPCO).* Barcelona, Spain: European Association for Signal Processing, 2011, pp. 1490–1494.

[57] J. F. Gemmeke, T. Virtanen, and A. Hurmalainen, "Exemplar-based sparse representations for noise robust automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2067–2080, 2011.

[58] ——, "Exemplar-based speech enhancement and its application to noise-robust automatic speech recognition," in *Proceedings of the International CHiME Workshop on Machine Listening in Multisource Environments*, Florence, Italy, 2011, pp. 53–57.

[59] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[60] L. Gillick and S. J. Cox, "Some statistical issues in the comparison of speech recognition algorithms," in *Proceedings of the 14th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* Speech City, Speechland: IEEE, 1989, pp. 23–26.

[61] S. Gonzalez and M. Brookes, "Mask-based enhancement for very low quality speech," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* Florence, Italy: IEEE, 2014, pp. 7079–7083.

[62] A. Graves, "Supervised sequence labelling with recurrent neural networks," Ph.D. dissertation, Technische Universität München, 2008.

[63] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[64] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems (NIPS)*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds. Granada, Spain: MIT Press, 2011, vol. 24, pp. 2348–2356.

[65] ——, "RNNLIB: A recurrent neural network library for sequence learning problems," http://sourceforge.net/projects/rnnl/, 2013.

[66] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proccedings of the 31st International Conference on Machine Learning (ICML)*. Beijing, China: International Machine Learning Society, 2014, no pagination.

[67] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 6645–6649.

[68] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proccedings of the 27th International Conference on Machine Learning (ICML)*. Haifa, Israel: International Machine Learning Society, 2010, pp. 399–406.

[69] F. Grezl and P. Fousek, "Optimizing bottle-neck features for LVCSR," in *Proceedings of the 33rd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Las Vegas, NV, USA: IEEE, 2008, pp. 4729–4732.

[70] F. Grezl, M. Karafiat, K. Stanislav, and J. Cernocky, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proceedings of the 32nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Honolulu, HI, USA: IEEE, 2007, pp. 757–760.

[71] P. Grosche, B. Schuller, M. Müller, and G. Rigoll, "Automatic transcription of recorded music," *Acta Acustica united with Acustica*, vol. 98, no. 2, pp. 199–215(17), 2012.

[72] E. Habets, "Speech dereverberation using statistical reverberation models," in *Speech Dereverberation*, P. A. Naylor and N. D. Gaubitch, Eds. Springer, 2010, pp. 57–93.

[73] T. Haderlein, C. Moers, B. Möbius, F. Rosanowski, and E. Nöth, "Intelligibility rating with automatic speech recognition, prosodic, and cepstral evaluation," in *Proceedings of Text, Speech and Dialogue (TSD)*, ser. Lecture Notes in Artificial Intelligence, vol. 6836. Berlin, Heidelberg: Springer, 2011, pp. 195–202.

[74] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newsletter*, vol. 11, pp. 10–18, 2009.

[75] J. Han and C.-W. Chen, "Improving melody extraction using probabilistic latent component analysis," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 33–36.

[76] T. Heittola, A. Mesaros, T. Virtanen, and A. Eronen, "Sound event detection in multisource environments using source separation," in *Proceedings of the International CHiME Workshop on Machine Listening in Multisource Environments*, Florence, Italy, 2011, pp. 86–90.

[77] M. Helen and T. Virtanen, "Separation of drums from polyphonic music using non-negative matrix factorization and support vector machine," in *Proceedings of the 13th European Signal Processing Conference (EUSIPCO)*. Antalya, Turkey: European Association for Signal Processing, 2005.

[78] H. Hermansky, "Perceptual linear predictive analysis for speech," *The Journal of The Acoustical Society of America*, vol. 87, pp. 1738–1752, 1990.

[79] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn, "RASTA-PLP speech analysis technique," in *Proceedings of the 17th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. San Francisco, CA, USA: IEEE, 1992, pp. 121–124.

[80] D. J. Hermes, "Measurement of pitch by subharmonic summation," *The Journal of the Acoustical Society of America*, vol. 83, no. 1, pp. 257–264, 1988.

[81] J. Hershey, J. Le Roux, and F. Weninger, "Deep unfolding: model-based inspiration of novel deep architectures," Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, Tech. Rep. TR2014-117, 2014.

[82] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[83] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, pp. 1771–1800, 2002.

[84] H.-G. Hirsch and D. Pearce, "The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *Proceedings of the ISCA Workshop on Automatic Speech Recognition: Challenges for the new Millennium (ASR-2000)*, 2000, pp. 181–188.

[85] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[86] M. Hollander and D. A. Wolfe, *Nonparametric Statistical Methods*. New York, NY, USA: John Wiley & Sons, 1973.

[87] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Deep learning for monaural speech separation," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, 2014, pp. 1581–1585.

[88] A. Hurmalainen, J. Gemmeke, and T. Virtanen, "Non-negative matrix deconvolution in noise robust speech recognition," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 4588–4591.

[89] A. Hurmalainen, K. Mahkonen, J. F. Gemmeke, and T. Virtanen, "Exemplar-based recognition of speech in highly variable noise," in *Proceedings of the International CHiME Workshop on Machine Listening in Multisource Environments*, Florence, Italy, 2011, pp. 1–5.

[90] A. Hurmalainen, R. Saeidi, and T. Virtanen, "Group sparsity for speaker identity discrimination in factorisation-based speech recognition," in *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Portland, OR, USA: ISCA, 2012, no pagination.

[91] A. Hurmalainen, J. F. Gemmeke, and T. Virtanen, "Modelling non-stationary noise with spectral factorisation in automatic speech recognition," *Computer Speech and Language, Special Issue on Speech Separation and Recognition in Multisource Environments*, vol. 27, no. 3, pp. 763–779, 2013.

[92] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. New York, NY, USA: John Wiley & Sons, 2001.

[93] T. Ishii, H. Komiyama, T. Shinozaki, Y. Horiuchi, and S. Kuroiwa, "Reverberant speech recognition based on denoising autoencoder," in *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Lyon, France: ISCA, 2013, pp. 3512–3516.

[94] M. Jessen, "Speaker classification in forensic phonetics and acoustics," in *Speaker Classification I: Fundamentals, Features, and Methods*, ser. Lecture Notes in Computer Science, C. Müller, Ed. Berlin / Heidelberg: Springer, 2007, vol. 4343, pp. 180–204.

[95] M. Jeub, M. Schäfer, and P. Vary, "A binaural room impulse response database for the evaluation of dereverberation algorithms," in *Proceedings of International Conference on Digital Signal Processing (DSP)*. Santorini, Greece: IEEE, 2009, pp. 1–4.

[96] C. Joder, F. Weninger, F. Eyben, D. Virette, and B. Schuller, "Real-time speech separation by semi-supervised nonnegative matrix factorization," in *Proceedings of the 10th International Conference on Latent Variable Analysis and Signal Separation (LVA ICA)*, F. J. Theis, A. Cichocki, A. Yeredor, and M. Zibulevsky, Eds., vol. 7191. Tel Aviv, Israel: Springer, 2012, pp. 322–329.

[97] C. Joder, F. Weninger, D. Virette, and B. Schuller, "A comparative study on sparsity penalties for NMF-based speech separation: Beyond Lp-norms," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 858–862.

[98] D. Johnson and D. Dudgeon, *Array Signal Processing*. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.

[99] I. T. Jolliffe, *Principal component analysis*. Berlin: Springer, 2002.

[100] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice Hall, 2008.

[101] S. M. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recogniser," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 3, pp. 400–401, 1987.

[102] M. Khadkevich and M. Omologo, "Use of hidden markov models and factored language models for automatic chord recognition," in *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*. Kobe, Japan: International Society for Music Information Retrieval, 2009, pp. 561–566.

[103] K. Kinoshita, M. Delcroix, T. Nakatani, and M. Miyoshi, "Suppression of late reverberation effect on speech signal using long-term multiplestep linear prediction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 4, pp. 534–545, 2009.

[104] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, E. Habets, R. Haeb-Umbach, V. Leutnant, A. Sehr, W. Kellermann, R. Maas, S. Gannot, and B. Raj, "The REVERB Challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. New Paltz, NY, USA: IEEE, 2013, no pagination.

[105] C. Kirst, F. Weninger, C. Joder, P. Grosche, J. Geiger, and B. Schuller, "Online NMF-based stereo up-mixing of speech improves perceived reduction of non-stationary noise," in *Proceedings of the Audio Engineering Society (AES) 53rd International Conference*, K. Brandenburg and M. Sandler, Eds. London, UK: Audio Engineering Society, 2014, pp. 1–7, Best Student Paper Award.

[106] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, pp. 320–327, 1976.

[107] K. Kroschel, G. Rigoll, and B. Schuller, *Statistische Informationstechnik*, 5th ed. Berlin/Heidelberg: Springer, 2011.

[108] K. Kumatani, J. McDonough, and B. Raj, "Microphone array processing for distant speech recognition: From close-talking microphones to far-field sensors," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 127–140, 2012.

[109] N. Landwehr, M. Hall, and E. Frank, "Logistic model trees," *Machine Learning*, vol. 59, pp. 161–205, 2005.

[110] J. Le Roux and E. Vincent, "A categorization of robust speech processing datasets," Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, Tech. Rep. TR2014-116, 2014.

[111] J. Le Roux, S. Watanabe, and J. R. Hershey, "Ensemble learning for speech enhancement," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. New Paltz, NY, USA: IEEE, 2013, no pagination.

[112] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems (NIPS)*. Vancouver, Canada: MIT Press, 2001, pp. 556–562.

[113] K. Lee, "Application of non-negative spectrogram decomposition with sparsity constraints to single-channel speech enhancement," *Speech Communication*, vol. 58, pp. 69–80, 2014.

[114] J. Li, L. Deng, Y. Gong, and R. Haeb-Umbach, "An overview of noise-robust automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 745–777, 2014.

[115] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, 2007.

[116] M. Lincoln, I. McCowan, J. Vepa, and H. Maganti, "The multi-channel Wall Street Journal audio visual corpus (MC-WSJ-AV): Specification and initial experiments," in *Proceedings of the 2005 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. San Juan, PR, USA: IEEE, 2005, pp. 357–362.

[117] Y. Liu, P. Zhang, and T. Hain, "Using neural network front-ends on far field multiple microphones based speech recognition," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, 2014, pp. 5579–5583.

[118] N. Lopes, R. Quintas, and J. Gonçalves, "GPUMLib," http://gpumlib.sourceforge.net/, 2013. [Online]. Available: http://gpumlib.sourceforge.net

[119] X. Lu, Y. Tsao, S. Matsuda, and C. Hori, "Speech enhancement based on deep denoising autoencoder," in *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Lyon, France: ISCA, 2013, pp. 3444–3448.

[120] I. Luengo, I. Saratxaga, E. Navas, I. Hernáez, J. Sanchez, and I. Sainz, "Evaluation of pitch detection algorithms under real life conditions," in *Proceedings of the 32nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Honolulu, HI, USA: IEEE, 2007, pp. 1057–1060.

[121] A. Maas, Q. Le, T. O'Neil, O. Vinyals, P. Nguyen, and A. Ng, "Recurrent neural networks for noise reduction in robust asr," in *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Portland, OR, USA: ISCA, 2012, no pagination.

[122] A. L. Maas, T. M. O'Neil, A. Y. Hannun, and A. Y. Ng, "Recurrent neural network feature enhancement: The 2nd CHiME Challenge," in *Proceedings of the 2nd International CHiME Workshop on Machine Listening in Multisource Environments*, Vancouver, Canada, 2013, pp. 79–80.

[123] M. Marolt, "A connectionist approach to automatic transcription of polyphonic piano music," *IEEE Transactions on Multimedia*, vol. 6, no. 3, pp. 439–449, 2004.

[124] R. Martin, "Noise Power Spectral Density Estimation Based on Optimal Smoothing and Minimum Statistics," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 9, no. 5, pp. 504–512, 2001.

[125] G. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*. Hoboken, NJ, USA: John Wiley & Sons, 2004.

[126] A. Mesaros and T. Virtanen, "Automatic recognition of lyrics in singing," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2010, no. 546047, pp. 1–11, 2010.

[127] A. Mesaros, T. Virtanen, and A. Klapuri, "Singer identification in polyphonic music using vocal separation and pattern recognition methods," in *Proceedings of the 8th International Society for Music Information Retrieval Conference (ISMIR)*. Vienna, Austria: International Society for Music Information Retrieval, 2007, pp. 375–378.

[128] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic event detection in real life recordings," in *Proceedings of the 18th European Signal Processing Conference (EUSIPCO)*. Aalborg, Denmark: European Association for Signal Processing, 2010, pp. 1267–1271.

[129] N. Mesgarani and E. F. Chang, "Selective cortical representation of attended speaker in multi-talker speech perception," *Nature*, vol. 485, pp. 233–237, 2012.

[130] F. Metze, J. Ajmera, R. Englert, U. Bub, F. Burkhardt, J. Stegmann, C. Müller, R. Huber, B. Andrassy, J. G. Bauer, and B. Littel, "Comparison of four approaches to age and gender recognition for telephone applications," in *Proceedings of the 32nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Honolulu, HI, USA: IEEE, 2007, pp. 1089–1092.

[131] A. Mohamed, G. Hinton, and G. Penn, "Understanding how deep belief networks perform acoustic modelling," in *Proceedings of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan: IEEE, 2012, pp. 4273–4276.

[132] N. Mohammadiha, P. Smaragdis, and A. Leijon, "Prediction based filtering and smoothing to exploit temporal dependencies in NMF," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 873–877.

[133] G. Montavon, G. B. Orr, and K.-R. Müller, Eds., *Neural Networks: Tricks of the Trade*, ser. Lecture Notes in Computer Science. Berlin / Heidelberg: Springer, 2012, vol. 7700.

[134] G. J. Mysore and P. Smaragdis, "A non-negative approach to semi-supervised separation of speech from noise with the use of temporal dynamics," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 17–20.

[135] J. Nam, J. Ngiam, H. Lee, and M. Slaney, "A classification-based polyphonic piano transcription approach using learned feature representations," in *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*. Miami Beach, FL, USA: ISMIR, 2011, pp. 175–180.

[136] A. Narayanan and D. Wang, "Ideal ratio mask estimation using deep neural networks for robust speech recognition," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 7092–7096.

[137] R. S. Nickerson, "Null hypothesis significance testing: A review of an old and continuing controversy," *Psychological Methods*, vol. 5, pp. 241–301, 2000.

[138] T. Nose, Y. Kato, and T. Kobayashi, "Style estimation of speech based on multiple regression hidden semi-Markov model," in *Proceedings of the 8th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Antwerp, Belgium: ISCA, 2007, pp. 2285–2288.

[139] T. L. Nwe and H. Li, "Exploring vibrato-motivated acoustic features for singer identification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 2, pp. 519–530, 2007.

[140] P. D. O'Grady and B. A. Pearlmutter, "Discovering convolutive speech phones using sparseness and non-negativity," in *Proceedings of the 7th International Conference on Independent Component Analysis and Signal Separation (ICA 2007)*, ser. Lecture Notes in Computer Science, M. E. Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, Eds. London, UK: Springer, 2007, vol. 4666, pp. 520–527.

[141] N. Ono, Z. Koldovsky, S. Miyabe, and N. Ito, "The 2013 Signal Separation Evaluation Campaign," in *Proceedings of the 2013 IEEE International Workshop on Machine Learning for Signal Processing*, Southampton, UK, 2013, pp. 1–6.

[142] A. Ozerov, C. Févotte, and M. Charbit, "Factorial scaled hidden Markov model for polyphonic audio representation and source separation," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. New Paltz, NY, USA: IEEE, 2009, pp. 121–124.

[143] D. S. Pallett, J. G. Fiscus, and M. A. Przybocki, "1996 preliminary broadcast news benchmark tests," in *Proceedings of the 1997 DARPA Speech Recognition Workshop*, 1997, no pagination.

[144] H.-S. Pang and D.-H. Yoon, "Automatic detection of vibrato in monophonic music," *Pattern Recognition*, vol. 38, no. 7, pp. 1135–1138, 2005.

[145] S. Parveen and P. Green, "Speech enhancement with missing data techniques using recurrent neural networks," in *Proceedings of the 29th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Montreal, Canada: IEEE, 2004, pp. 733–736.

[146] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014. [Online]. Available: http://arxiv.org/abs/1312.6026

[147] D. B. Paul and J. M. Baker, "The design for the Wall Street Journal-based CSR corpus," in *Proceedings of the Workshop on Speech and Natural Language (HLT-91)*, 1992, pp. 357–362.

[148] M. Pitt, L. Dilley, K. Johnson, S. Kiesling, W. Raymond, E. Hume, and E. Fosler-Lussier, *Buckeye Corpus of Conversational Speech (2nd release)*. Columbus, OH, USA: Department of Psychology, Ohio State University (Distributor), 2007, www.buckeyecorpus.osu.edu.

[149] G. E. Poliner and D. P. W. Ellis, "A discriminative model for polyphonic piano transcription," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, no. 48317, pp. 1–9, 2007.

[150] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.

[151] D. Povey and K. Yao, "A basis method for robust estimation of Constrained MLLR," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 4460–4463.

[152] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlícek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *Proceedings of the 2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. Honolulu, HI, USA: IEEE, 2011.

[153] D. Povey and K. Yao, "A basis representation of constrained MLLR transforms for robust adaptation," *Computer Speech and Language*, vol. 26, pp. 35–51, 2012.

[154] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for model and feature-space discriminative training," in *Proceedings of the 33rd IEEE International Conference on*

Acoustics, Speech and Signal Processing (ICASSP). Las Vegas, NV, USA: IEEE, 2008, pp. 4057–4060.

[155] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," in *Proceedings of the IEEE*, vol. 77, 1989, pp. 257–286.

[156] L. R. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.

[157] B. Raj, T. Virtanen, S. Chaudhuri, and R. Singh, "Non-negative matrix factorization based compensation of music for automatic speech recognition," in *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Makuhari, Japan: ISCA, 2010, pp. 717–720.

[158] R. Ratnam, D. L. Jones, B. C. Wheeler, W. D. O'Brien, Jr, C. R. Lansing, and A. S. Feng, "Blind estimation of reverberation time," *The Journal of the Acoustical Society of America*, vol. 114, no. 5, pp. 2877–2892, 2003.

[159] S. J. Rennie, J. R. Hershey, and P. A. Olsen, "Efficient model-based speech separation and denoising using non-negative subspace analysis," in *Proceedings of the 33rd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Las Vegas, NV, USA: IEEE, 2008, pp. 1833–1836.

[160] ——, "Single-channel multitalker speech recognition," *IEEE Signal Processing Magazine*, vol. 27, no. 6, pp. 66–80, 2010.

[161] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the RPROP algorithm," in *Proceedings of the IEEE International Conference on Neural Networks*, 1993, pp. 586–591.

[162] T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals, "WSJCAM0: A British English speech corpus for large vocabulary continuous speech recognition," in *Proceedings of the 20th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Detroit, MI, USA: IEEE, 1995, pp. 81–84.

[163] H. Sak, O. Vinyals, G. Heigold, A. Senior, E. McDermott, R. Monga, and M. Mao, "Sequence discriminative distributed training of long short-term memory recurrent neural networks," in *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Singapore, Singapore: ISCA, 2014, in press.

[164] J. Salamon, B. Rocha, and E. Gómez, "Musical genre classification using melody features extracted from polyphonic music signals," in *Proceedings*

of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Kyoto, Japan: IEEE, 2012, pp. 81–84.

[165] T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber, "PyBrain," *Journal of Machine Learning Research*, vol. 11, pp. 743–746, 2010.

[166] M. N. Schmidt and R. K. Olsson, "Single-channel speech separation using sparse non-negative matrix factorization," in *Proceedings of the Ninth International Conference on Spoken Language Processing (INTERSPEECH - ICSLP)*. Pittsburgh, PA, USA: ISCA, 2006.

[167] C. Schörkhuber and A. Klapuri, "Constant-Q transform toolbox for music processing," in *Proceedings of the 7th Sound and Music Computing Conference (SMC-2010)*. Barcelona, Spain: Universitat Pompeu Fabra, 2010, (no pagination).

[168] B. Schuller, "The Computational Paralinguistics Challenge," *IEEE Signal Processing Magazine*, vol. 29, no. 4, pp. 97–101, 2012.

[169] B. Schuller, F. Eyben, and G. Rigoll, "Tango or waltz? – Putting ballroom dance style into tempo detection," *EURASIP Journal on Audio, Speech, and Music Processing (JASMP), Special Issue on Intelligent Audio, Speech, and Music Processing Applications*, vol. 2008, no. 846135, pp. 1–12, 2008.

[170] B. Schuller, A. Lehmann, F. Weninger, F. Eyben, and G. Rigoll, "Blind enhancement of the rhythmic and harmonic sections by NMF: Does it help?" in *Proc. of the International Conference on Acoustics (NAG/DAGA 2009)*. Rotterdam, Netherlands: DEGA, 2009, pp. 361–364.

[171] B. Schuller, M. Wöllmer, T. Moosmayr, and G. Rigoll, "Recognition of noisy speech: A comparative survey of robust model architecture and feature enhancement," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2009, no. 942617, pp. 1–17, 2009.

[172] B. Schuller, F. Weninger, M. Wöllmer, Y. Sun, and G. Rigoll, "Non-negative matrix factorization as noise-robust feature extractor for speech recognition," in *Proceedings of the 35th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Dallas, TX, USA: IEEE, 2010, pp. 4562–4565.

[173] B. Schuller, *Intelligent Audio Analysis*, ser. Signals and Communication Technology. Springer, 2013, 350 pages.

[174] B. Schuller and A. Batliner, *Computational Paralinguistics: Emotion, Affect and Personality in Speech and Language Processing*. Hoboken, NJ, USA: John Wiley & Sons, 2014, in press.

[175] B. Schuller, C. Kozielski, F. Weninger, F. Eyben, and G. Rigoll, "Vocalist gender recognition in recorded popular music," in *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*. Utrecht, The Netherlands: International Society for Music Information Retrieval, 2010, pp. 613–618.

[176] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and S. Narayanan, "The INTERSPEECH 2010 Paralinguistic Challenge," in *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Makuhari, Japan: ISCA, 2010, pp. 2794–2797.

[177] B. Schuller, M. Valstar, F. Eyben, G. McKeown, R. Cowie, and M. Pantic, "AVEC 2011 – the first international Audio/Visual Emotion Challenge," in *Proceedings First International Audio/Visual Emotion Challenge and Workshop (AVEC), held in conjunction with the International HUMAINE Association Conference on Affective Computing and Intelligent Interaction (ACII)*, B. Schuller, M. Valstar, R. Cowie, and M. Pantic, Eds. Memphis, TN, USA: Springer, 2011, pp. 415–424.

[178] B. Schuller, F. Pokorny, S. Ladstätter, M. Fellner, F. Graf, and L. Paletta, "Acoustic Geo-Sensing: Recognising cyclists' route, route direction, and route progress from cell-phone audio," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 453–457.

[179] A. Sehr, R. Maas, and W. Kellermann, "Model-based dereverberation in the Logmelspec domain for robust distant-talking speech recognition," in *Proceedings of the 35th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Dallas, USA: IEEE, 2010, pp. 4298–4301.

[180] ——, "Frame-wise HMM adaptation using state-dependent reverberation estimates," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 5484–5487.

[181] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," in *Proceedings of the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Florence, Italy: ISCA, 2011, pp. 437–440.

[182] M. L. Seltzer and J. Droppo, "Multi-task learning in deep neural networks for improved phoneme recognition," in *Proceedings of the 38th IEEE International*

Conference on Acoustics, Speech and Signal Processing (ICASSP). Vancouver, Canada: IEEE, 2013, pp. 6965–6969.

[183] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 7398–7402.

[184] P. Smaragdis, "Discovering auditory objects through non-negativity constraints," in *Proceedings of the ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing (SAPA)*. Jeju, Korea: ISCA, 2004, no pagination.

[185] ——, "Convolutive speech bases and their application to supervised speech separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 1, pp. 1–14, 2007.

[186] P. Smaragdis and J. C. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. New Paltz, NY, USA: IEEE, 2003, pp. 177–180.

[187] P. Smaragdis, B. Raj, and M. Shashanka, "Supervised and semi-supervised separation of sounds from single-channel mixtures," in *Proceedings of the 7th International Conference on Independent Component Analysis and Signal Separation (ICA 2007)*, ser. Lecture Notes in Computer Science, M. E. Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, Eds., vol. 4666. London, UK: Springer, 2007, pp. 414–421.

[188] P. Sprechmann, R. Litman, T. B. Yakar, A. M. Bronstein, and G. Sapiro, "Supervised sparse analysis and synthesis operators," in *Advances in Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, USA, 2013, pp. 908–916.

[189] J. Stadermann, W. Koska, and G. Rigoll, "Multi-task learning strategies for a recurrent neural net in a hybrid tied-posteriors acoustic model," in *Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*. Lisbon, Portugal: ISCA, 2005, pp. 2993–2996.

[190] A. Stupakov, E. Hanusa, D. Vijaywargi, D. Fox, and J. Bilmes, "The design and collection of COSINE, a multi-microphone in situ speech corpus recorded in noisy environments," *Computer Speech and Language*, vol. 26, no. 1, pp. 52–66, 2011.

[191] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proccedings of the 30th*

International Conference on Machine Learning (ICML). Atlanta, GA, USA: International Machine Learning Society, 2013, no pagination.

[192] Y. Tachioka, S. Watanabe, and J. R. Hershey, "Effectiveness of discriminative training and feature transformation for reverberated and noisy speech," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 6935–6939.

[193] Y. Tachioka, T. Hanazawa, and T. Iwasaki, "Dereverberation method with reverberation time estimation using floored ratio of spectral subtraction," *Acoustical Science and Technology*, vol. 34, no. 3, pp. 212–215, 2013.

[194] Y. Tachioka, T. Narita, F. Weninger, and S. Watanabe, "Dual system combination approach for various reverberant environments with dereverberation techniques," in *Proceedings of the REVERB Workshop held in conjunction with ICASSP 2014*, Florence, Italy, 2014, no pagination.

[195] Y. Teng, "Objective speech intelligibility assessment using speech recognition and bigram statistics with application to low bit-rate codec evaluation," Ph.D. dissertation, University of Wyoming, Laramie, WY, USA, 2006.

[196] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. Schuller, "A deep semi-NMF model for learning hidden representations," in *Proccedings of the 31st International Conference on Machine Learning (ICML)*. Beijing, China: International Machine Learning Society, 2014, no pagination.

[197] C. Uhle, C. Dittmar, and T. Sporer, "Extraction of drum tracks from polyphonic music using independent subspace analysis," in *Proceedings of the 4th International Symposium on Independent Component Analysis and Signal Separation (ICA 2003)*, S.-I. Amari, A. Cichocki, S. Makino, and N. Murata, Eds., Nara, Japan, 2003, pp. 834–848.

[198] C. G. v. d. Boogaart and R. Lienhart, "Note onset detection for the transcription of polyphonic piano music," in *Proceedings of the 2009 IEEE International conference on Multimedia and Expo*, 2009, pp. 446–449.

[199] K. Vertanen, "Baseline WSJ acoustic models for HTK and Sphinx: Training recipes and recognition experiments," Cavendish Laboratory, University of Cambridge, Tech. Rep., 2006.

[200] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[201] E. Vincent, N. Bertin, and R. Badeau, "Two nonnegative matrix factorization methods for polyphonic pitch transcription," in *Proceedings of Music Information Retrieval Evaluation eXchange (MIREX)*, Vienna, Austria, 2007, pp. 23–30.

[202] E. Vincent, J. Barker, S. Watanabe, J. Le Roux, F. Nesta, and M. Matassoni, "The second 'CHiME' speech separation and recognition challenge: Datasets, tasks and baselines," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 126–130.

[203] ——, "The second CHiME speech separation and recognition challenge: an overview of challenge systems and outcomes," in *Proceedings of the 2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. Olomouc, Czech Republic: IEEE, 2013.

[204] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proccedings of the 25th International Conference on Machine Learning (ICML)*. Helsinki, Finland: International Machine Learning Society, 2008, pp. 1096–1103.

[205] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 3, pp. 1066–1074, 2007.

[206] ——, "Speech recognition using factorial Hidden Markov Models for separation in the feature space," in *Proceedings of the Ninth International Conference on Spoken Language Processing (INTERSPEECH - ICSLP)*. Pittsburgh, PA, USA: ISCA, 2006, pp. 1–4.

[207] R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *Journal of the ACM*, vol. 21, no. 1, pp. 168–173, 1974.

[208] W. Wang, A. Cichocki, and J. A. Chambers, "A multiplicative algorithm for convolutive non-negative matrix factorization based on squared Euclidean distance," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2858–2864, 2009.

[209] C. Weng, D. Yu, S. Watanabe, and B.-H. Juang, "Recurrent deep neural networks for robust speech recognition," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, 2014, pp. 5569–5573.

[210] C. Weng, D. Yu, M. L. Seltzer, and J. Droppo, "Single-channel mixed speech recognition using deep neural networks," in *Proceedings of the 39th IEEE*

International Conference on Acoustics, Speech and Signal Processing (ICASSP). Florence, Italy: IEEE, 2014, pp. 5669–5673.

[211] F. Weninger, J. Geiger, M. Wöllmer, B. Schuller, and G. Rigoll, "The Munich 2011 CHiME Challenge contribution: NMF-BLSTM speech enhancement and recognition for reverberated multisource environments," in *Proceedings of the International CHiME Workshop on Machine Listening in Multisource Environments*, Florence, Italy, 2011, pp. 24–29.

[212] F. Weninger, A. Lehmann, and B. Schuller, "openBliSSART: Design and evaluation of a research toolkit for Blind Source Separation in Audio Recognition Tasks," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 1625–1628.

[213] F. Weninger, M. Wöllmer, and B. Schuller, "Automatic assessment of singer traits in popular music: Gender, age, height and race," in *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*. Miami Beach, FL, USA: International Society for Music Information Retrieval, 2011, pp. 37–42.

[214] F. Weninger, J. Krajewski, A. Batliner, and B. Schuller, "The voice of leadership: Models and performances of automatic analysis in on-line speeches," *IEEE Transactions on Affective Computing*, vol. 3, no. 4, pp. 496–508, 2012.

[215] F. Weninger, J. R. Hershey, J. Le Roux, and B. Schuller, "Discriminatively trained recurrent neural networks for single-channel speech separation," in *Proceedings of the 2nd IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Atlanta, GA, USA: IEEE, 2014, in press.

[216] F. Weninger, J. Le Roux, J. R. Hershey, and S. Watanabe, "Discriminative NMF and its application to single-channel source separation," in *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Singapore, Singapore: ISCA, 2014, in press.

[217] F. Weninger and B. Schuller, "Optimization and parallelization of monaural source separation algorithms in the openBliSSART toolkit," *Journal of Signal Processing Systems*, vol. 69, no. 3, pp. 267–277, 2012.

[218] F. Weninger, J.-L. Durrieu, F. Eyben, G. Richard, and B. Schuller, "Combining monaural source separation with long short-term memory for increased robustness in vocalist gender recognition," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 2196–2199.

[219] F. Weninger, N. Amir, O. Amir, I. Ronen, F. Eyben, and B. Schuller, "Robust feature extraction for automatic recognition of vibrato singing in recorded polyphonic music," in *Proceedings of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan: IEEE, 2012, pp. 85–88.

[220] F. Weninger, J. Feliu, and B. Schuller, "Supervised and semi-supervised supression of background music in monaural speech recordings," in *Proceedings of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan: IEEE, 2012, pp. 61–64.

[221] F. Weninger, B. Schuller, C. Liem, F. Kurth, and A. Hanjalic, "Music Information Retrieval: An inspirational guide to transfer from related disciplines," in *Multimodal Music Processing*, ser. Dagstuhl Follow-Ups, M. Müller and M. Goto, Eds., vol. 11041, Schloss Dagstuhl, Germany, 2012, pp. 195–215.

[222] F. Weninger, M. Wöllmer, J. Geiger, B. Schuller, J. Gemmeke, A. Hurmalainen, T. Virtanen, and G. Rigoll, "Non-negative matrix factorization for highly noise-robust ASR: to enhance or to recognize?" in *Proceedings of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan: IEEE, 2012, pp. 4681–4684.

[223] F. Weninger, M. Wöllmer, and B. Schuller, "Combining Bottleneck-BLSTM and semi-supervised sparse NMF for recognition of conversational speech in highly instationary noise," in *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Portland, OR, USA: ISCA, 2012, no pagination.

[224] ——, "Sparse, hierarchical and semi-supervised base learning for monaural enhancement of conversational speech," in *Proceedings 10th ITG Conference on Speech Communication*, T. Fingscheidt and W. Kellermann, Eds., ITG. Braunschweig, Germany: IEEE, 2012, pp. 1–4.

[225] F. Weninger, J. Geiger, M. Wöllmer, B. Schuller, and G. Rigoll, "The Munich feature enhancement approach to the 2013 CHiME Challenge using BLSTM recurrent neural networks," in *Proceedings of the 2nd International CHiME Workshop on Machine Listening in Multisource Environments*, Vancouver, Canada, 2013, pp. 86–90.

[226] F. Weninger, C. Kirst, B. Schuller, and H.-J. Bungartz, "A discriminative approach to polyphonic piano note transcription using non-negative matrix factorization," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 6–10.

[227] F. Weninger, C. Wagner, M. Wöllmer, B. Schuller, and L.-P. Morency, "Speaker trait characterization in web videos: Uniting speech, language, and facial features," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 3647–3651.

[228] F. Weninger, J. Bergmann, and B. Schuller, "Introducing CURRENNT – the Munich open-source CUDA RecurREnt Neural Network Toolkit," *Journal of Machine Learning Research*, vol. 15, 2014, in press.

[229] F. Weninger, F. Eyben, and B. Schuller, "Single-channel speech separation with memory-enhanced recurrent neural networks," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, 2014, pp. 3737–3741.

[230] F. Weninger, J. Geiger, M. Wöllmer, B. Schuller, and G. Rigoll, "Feature enhancement by deep LSTM networks for ASR in reverberant multisource environments," *Computer Speech and Language*, vol. 28, no. 4, pp. 888–902, 2014.

[231] F. Weninger, S. Watanabe, J. Le Roux, J. R. Hershey, Y. Tachioka, J. Geiger, B. Schuller, and G. Rigoll, "The MERL/MELCO/TUM system for the REVERB challenge using deep recurrent neural network feature enhancement," in *Proceedings of the REVERB Workshop held in conjunction with ICASSP 2014*, Florence, Italy, 2014, no pagination.

[232] F. Weninger, S. Watanabe, Y. Tachioka, and B. Schuller, "Deep recurrent de-noising auto-encoder and blind de-reverberation for reverberated speech recognition," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, 2014, pp. 4656–4660.

[233] R. C. Whaley, A. Petitet, and J. Dongarra, "Automated Empirical Optimization of Software and the ATLAS project," *Parallel Computing*, vol. 27, no. 1-2, pp. 3–35, 2001.

[234] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[235] M. Wöllmer, B. Schuller, and G. Rigoll, "A novel Bottleneck-BLSTM front-end for feature-level context modeling in conversational speech recognition," in *Proceedings of the 2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. Honolulu, HI, USA: IEEE, 2011, pp. 36–41.

[236] M. Wöllmer, Z. Zhang, F. Weninger, B. Schuller, and G. Rigoll, "Feature enhancement by bidirectional LSTM networks for conversational speech recognition in highly non-stationary noise," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vancouver, Canada: IEEE, 2013, pp. 6822–6826.

[237] M. Wöllmer, "Context-sensitive machine learning for intelligent human behavior analysis," Ph.D. dissertation, Technische Universität München, 2013.

[238] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "A multi-stream ASR framework for BLSTM modeling of conversational speech," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Prague, Czech Republic: IEEE, 2011, pp. 4860–4863.

[239] M. Wöllmer, E. Marchi, S. Squartini, and B. Schuller, "Multi-stream LSTM-HMM decoding and histogram equalization for noise robust keyword spotting," *Cognitive Neurodynamics*, vol. 5, no. 3, pp. 253–264, 2011.

[240] M. Wöllmer, B. Schuller, and G. Rigoll, "Feature frame stacking in RNN-based Tandem ASR systems – learned vs. predefined context," in *Proceedings of the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Florence, Italy: ISCA, 2011, pp. 1233–1236.

[241] M. Wöllmer, F. Weninger, S. Steidl, A. Batliner, and B. Schuller, "Speech-based non-prototypical affect recognition for child-robot interaction in reverberated environments," in *Proceedings of the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Florence, Italy: ISCA, 2011, pp. 3113–3116.

[242] M. Wöllmer, F. Weninger, J. Geiger, B. Schuller, and G. Rigoll, "Noise robust ASR in reverberated multisource environments applying convolutive NMF and Long Short-Term Memory," *Computer Speech and Language, Special Issue on Speech Separation and Recognition in Multisource Environments*, vol. 27, no. 3, pp. 780–797, 2013.

[243] H. Xu, D. Povey, L. Mangu, and J. Zhu, "Minimum bayes risk decoding and system combination based on a recursion for edit distance," *Computer Speech & Language*, vol. 25, no. 4, pp. 802–828, 2011.

[244] K. Yasuda, "Accelerating Density Functional Calculations with Graphics Processing Unit," *Journal of Chemical Theory and Computation*, vol. 4, pp. 1230–1236, 2008.

[245] S. J. Young, G. Evermann, M. J. F. Gales, D. Kershaw, G. Moore, J. J. Odell, D. G. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK book*

version 3.4. Cambridge, UK: Cambridge University Engineering Department, 2006.

[246] D. Yu, L. Deng, F. Seide, and G. Li, "Discriminative pretraining of deep neural networks," US Patent 13/304 643, 2011, pending.

[247] D. Yu, M. L. Seltzer, J. Li, J. Huang, and F. Seide, "Feature learning in deep neural networks – A study on speech recognition tasks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2013, no pagination. [Online]. Available: http://arxiv.org/abs/1301.3605

[248] C. Zhang and J. H. L. Hansen, "Analysis and classification of speech mode: whispered through shouted," in *Proceedings of the 8th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. Antwerp, Belgium: ISCA, 2007, pp. 2396–2399.