Deep neural network based multichannel audio source separation

Aditya Arie Nugraha, Antoine Liutkus, Emmanuel Vincent

Chapter 1
Deep neural network based multichannel audio source separation

Aditya Arie Nugraha, Antoine Liutkus, and Emmanuel Vincent

Abstract

This chapter presents a multichannel audio source separation framework where deep neural networks (DNNs) are used to model the source spectra and combined with the classical multichannel Gaussian model to exploit the spatial information. The parameters are estimated in an iterative expectation-maximization (EM) fashion and used to derive a multichannel Wiener filter. Different design choices and their impact on the performance are discussed. They include the cost functions for DNN training, the number of parameter updates, the use of multiple DNNs, and the use of weighted parameter updates. Finally, we present its application to a speech enhancement task and a music separation task. The experimental results show the benefit of the multichannel DNN-based approach over a single-channel DNN-based approach and the multichannel nonnegative matrix factorization based iterative EM framework.

1.1 Introduction

Audio source separation aims to recover the underlying constitutive source signals of an observed mixture signal [1–5]. Related research can be divided into speech separation and music separation. Speech separation aims to recover the speech signal from a mixture containing multiple background noise sources with possibly interfering speech. This is important for speech enhancement (including in hearing aids) and noise-robust automatic speech recognition (ASR). Music separation aims to recover singing voice and musical instruments from a mixture. This has various applications, including music editing (remixing, upmixing, etc.), music information

Inria, Villers-lès-Nancy, F-54600, France
Université de Lorraine, LORIA, UMR 7503, Vandœuvre-lès-Nancy, F-54506, France
CNRS, LORIA, UMR 7503, Vandœuvre-lès-Nancy, F-54506, France
E-mail: {aditya.nugraha, antoine.liutkus, emmanuel.vincent}@inria.fr

retrieval, and dialogue extraction (e.g. from a mixture containing music accompaniment).

Acquiring (recording) audio using a single microphone and performing single-channel separation on the acquired signal is practical, especially for speech in real-world environments. However, along with the technology development, it is getting easier and cheaper to acquire audio using multiple microphones. The acquired multichannel signal captures additional information useful for separation. As a simple analogy, humans are able to predict the direction from which a sound comes based on the difference of amplitude and phase between the signals captured by the left and right ears. These inter-channel differences are related to the position of the sound sources relative to the microphones and can be exploited in multichannel separation to provide better results than single-channel separation. Multichannel separation is also preferable for music since most professionally-produced recordings available nowadays are in stereo (two-channel) format. If we consider the application on film audio tracks, we will deal with either six- or eight-channel surround sound formats.

Recent studies have shown that deep neural networks (DNNs) are able to model complex functions and perform well on various tasks, notably ASR [6, 7]. DNNs also have been applied to single-channel speech enhancement and shown to provide a significant increase in ASR performance compared to earlier approaches based on beamforming or nonnegative matrix factorization (NMF) [8]. The DNNs typically operate on magnitude or log-magnitude spectra in the Mel domain or the short time Fourier transform (STFT) domain. Various other features have been studied [9]. The DNNs can be used either to predict the source spectrograms [10–15] whose ratio yields a time-frequency mask or directly to predict a time-frequency mask [16–22]. The estimated source signal is then obtained as the product of the input mixture signal and the estimated time-frequency mask. Various DNN architectures and training criteria have been investigated and compared [19, 21, 23]. Although the authors in [13] considered both speech and music separation, most studies focused either on speech separation [10, 12, 14, 16–22] or on music separation [11, 15]. The approaches above considered single-channel source separation, where the input signal is either one of the channels of the original multichannel mixture signal or the result of delay-and-sum (DS) beamforming [19]. As a result, they do not fully exploit the benefits of multichannel data as achieved by multichannel filtering [1, 4].

There also exist a few approaches exploiting multichannel data. These approaches can be divided into three categories: (1) the ones deriving DNN input features from multichannel data for estimating a single-channel mask [14, 18]; (2) the ones estimating multichannel filters directly from time-domain signals using DNNs [24]; and (3) the ones estimating mask or spectra using DNNs which then are used to derive multichannel filters [25, 26].

In this chapter, we discuss the DNN-based multichannel source separation framework proposed in [26] which belongs to the third category. In this framework, DNNs are used to model the source spectra and combined with the classical multichannel Gaussian model to exploit the spatial information. These spectral and spatial parameters are then re-estimated in an iterative expectation-maximization (EM) fashion and used to derive a multichannel Wiener filter. This framework is built upon the

classical iterative EM framework in [27], which was also used up to some variants in [28–33]. This chapter summarizes and reuses the materials from our works in [26, 34, 35]. This chapter presents a brief study of the impact of different design choices on the performance of the DNN-based framework, including the cost function used for the DNN training, the number of spatial parameter updates, the number of EM iterations, the use of weighted parameter updates, and the use of multiple DNNs. Finally, this chapter also presents the application of the DNN-based framework to a speech enhancement task and a music separation task.

The rest of this chapter is organized as follows. Section 1.2 formulates the problem of multichannel audio source separation and describes the classical iterative EM framework, which is the basis for the DNN-based iterative framework described in Section 1.3. Section 1.4 presents the application of the framework to a speech enhancement task and a music separation task. We also present the impact of different design choices and the comparison to other separation techniques in these sections. Finally, Section 1.5 concludes the chapter.

1.2 Background

In this section, the problem of multichannel audio source separation is formulated. Following this formulation, the classical iterative EM framework is then described.

1.2.1 Problem formulation

Let I denote the number of channels, J the number of sources, $\mathbf{x}(t) \in \mathbb{R}^{I\times 1}$ the observed I-channel mixture signal, and $\mathbf{c}_j(t) \in \mathbb{R}^{I\times 1}$ the I-channel spatial image of source j. This source spatial image $\mathbf{c}_j(t)$ is a version of the signal of source j which presents in the observed mixture $\mathbf{x}(t)$. Both $\mathbf{x}(t)$ and $\mathbf{c}_j(t)$ are in the time domain and related by

$$\mathbf{x}(t) = \sum_{j=1}^{J} \mathbf{c}_j(t). \tag{1.1}$$

Figure 1.1 shows an example of studio music production where a song consists of vocal, guitars, and drums. As defined by (1.1), this stereo song is composed by the stereo spatial images of vocals, guitars, and drums. These stereo spatial images are not necessarily the original recordings. In most cases, sound engineers modify and mix the original sources. Since they can do downmixing (combining several channels to get fewer channels) or upmixing (splitting few channels to get more channels), the number of channels of the spatial images and the mixture may be chosen freely. In contrast, for the cases of speech recordings in real-world environments or live music recordings, the number of channels of the spatial images and

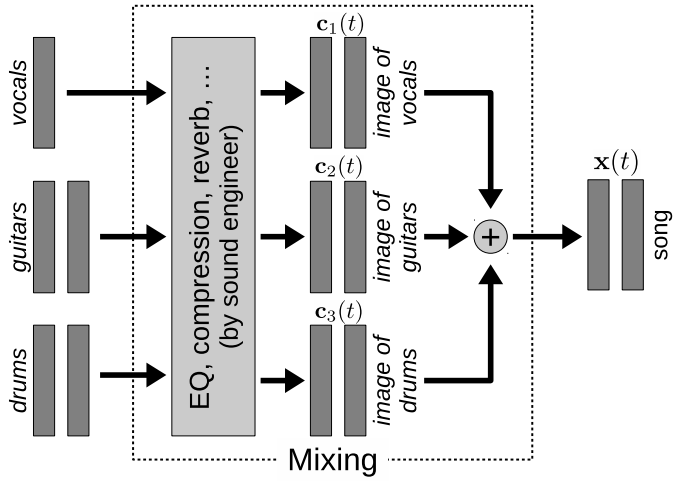


Fig. 1.1 Illustration of a studio music production.

the mixture corresponds to the number of microphones. Each channel of a source spatial image is the signal captured by each microphone after traveling from the source.

Multichannel audio source separation aims to recover the multichannel source spatial images $\mathbf{c}_j(t)$ from the observed multichannel mixture signal $\mathbf{x}(t)$.

1.2.2 Multichannel Gaussian model

Let $\mathbf{x}(f,n) \in \mathbb{C}^{I\times 1}$ and $\mathbf{c}_j(f,n) \in \mathbb{C}^{I\times 1}$ denote the short-time Fourier transform (STFT) coefficients [36] of $\mathbf{x}(t)$ and $\mathbf{c}_j(t)$, respectively, for frequency bin f and time frame n. Also, let F be the number of frequency bins and N the number of time frames.

We assume that $\mathbf{c}_j(f,n)$ are independent for different j, f, or n and follow a multivariate complex-valued zero-mean Gaussian distribution [27, 37]

$$\mathbf{c}_j(f,n) \sim \mathcal{N}_c\left(\mathbf{0}, v_j(f,n)\mathbf{R}_j(f)\right), \tag{1.2}$$

where $v_j(f,n) \in \mathbb{R}_+$ denotes the power spectral density (PSD) of source j for frequency bin f and time frame n, and $\mathbf{R}_j(f) \in \mathbb{C}^{I\times I}$ is the spatial covariance matrix of source j for frequency bin f. This $I \times I$ matrix represents spatial information by encoding the spatial position and the spatial width of the corresponding source [27]. Since the mixture $\mathbf{x}(f,n)$ is the sum of $\mathbf{c}_j(f,n)$, it is consequently distributed as

$$\mathbf{x}(f,n) \sim \mathcal{N}_c\left(\mathbf{0}, \sum_{j=1}^{J} v_j(f,n)\mathbf{R}_j(f)\right). \tag{1.3}$$

Given the PSDs $v_j(f,n)$ and the spatial covariance matrices $\mathbf{R}_j(f)$ of all sources, the spatial source images can be estimated in the minimum mean squared error sense using multichannel Wiener filtering [27]

$$\widehat{\mathbf{c}}_j(f,n) = \mathbf{W}_j(f,n)\mathbf{x}(f,n), \tag{1.4}$$

where the Wiener filter $\mathbf{W}_j(f,n)$ is given by

$$\mathbf{W}_j(f,n) = v_j(f,n)\mathbf{R}_j(f)\left(\sum_{j'=1}^{J} v_{j'}(f,n)\mathbf{R}_{j'}(f)\right)^{-1}. \tag{1.5}$$

Finally, the time-domain source estimates $\widehat{\mathbf{c}}_j(t)$ are recovered from $\widehat{\mathbf{c}}_j(f,n)$ by inverse STFT.

Following this formulation, instead of estimating the source spatial image $\mathbf{c}_j(t)$, source separation becomes the problem of estimating the PSDs $v_j(f,n)$ and the spatial covariance matrices $\mathbf{R}_j(f)$ of all sources.

1.2.3 General iterative EM framework

The general iterative EM framework for estimating the PSDs $v_j(f,n)$ and the spatial covariance matrices $\mathbf{R}_j(f)$ of all sources is summarized in Algorithm 1. For the following discussions, the term 'spectral parameters' refers to the PSDs and they are used interchangeably. Further, they are also used interchangeably with 'spectrograms', which are the graphical representations of PSDs over time. Likewise, the term 'spatial parameters' refers to the spatial covariance matrices.

In the beginning, the estimated PSDs $v_j(f,n)$ are initialized in the *spectrogram initialization* step. This can be done, for instance, by computing the PSD of the mixture and then dividing it with the number of sources, which implies each source contributes equally to the mixture. Initialization is also done for the estimated spatial covariance matrices $\mathbf{R}_j(f)$, for instance by assigning $I \times I$ identity matrices.

The following iterations can be divided into E-step and M-step. In the E-step, given the estimated parameters $v_j(f,n)$ and $\mathbf{R}_j(f)$ of each source, the source image estimates $\widehat{\mathbf{c}}_j(f,n)$ are obtained by multichannel Wiener filtering (1.4) and the posterior second-order raw moments of the spatial source images $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$ are computed as

$$\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) = \widehat{\mathbf{c}}_j(f,n)\widehat{\mathbf{c}}_j^H(f,n) + (\mathbf{I} - \mathbf{W}_j(f,n))\, v_j(f,n)\mathbf{R}_j(f), \tag{1.6}$$

where $\mathbf{I}$ denotes the $I \times I$ identity matrix and $\cdot^H$ is the Hermitian transposition.

In the M-step, the spatial covariance matrices $\mathbf{R}_j(f)$ are updated as

$$\mathbf{R}_j(f) = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{v_j(f,n)}\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n). \tag{1.7}$$

The source PSDs $v_j(f,n)$ are first estimated without constraints as

$$z_j(f,n) = \frac{1}{I}\mathrm{tr}\left(\mathbf{R}_j^{-1}(f)\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)\right), \tag{1.8}$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix. Then, they are updated according to a given spectral model by fitting $v_j(f,n)$ from $z_j(f,n)$ in the *spectrogram fitting* step. The spectrogram initialization and the spectrogram fitting steps depend on how the spectral parameters are modeled. Spectral models used in this context (denoted by M_0, M_1, …, M_J in Algorithm 1) may include NMF [29], which is a linear model with nonnegativity constraints, KAM [32, 33], which relies on the local regularity of the sources, and continuity models [28]. In this chapter, we present the use of DNNs for this purpose.

1.3 DNN-based multichannel source separation

In this section, the DNN-based multichannel source separation framework is presented. The cost functions for DNN training are also discussed. Finally, the weighted spatial parameter updates is introduced.

1.3.1 Algorithm

As indicated in the end of previous section, in this DNN-based framework, DNNs are used to model the source specta $v_j(f,n)$. This framework works with a single DNN or multiple DNNs. In the single DNN case, the DNN is used for spectrogram initialization (without any following spectrogram fitting). In the multiple DNNs case, one DNN is used for spectrogram initialization and one or more DNNs are used for spectrogram fitting. We can train different DNNs for spectrogram fitting at different iterations. Thus, the maximum number of DNNs for spectrogram fitting is equal to the number of iterations L. Let DNN_0 and DNN_l be the DNNs used for spectrogram initialization and spectrogram fitting, respectively. DNN_0 estimates the source spectra from the observed mixture and DNN_l aims to improve the source spectra estimated at iteration l. DNN_0 and DNN_l estimate the spectra of all sources simultaneously. This is similar to the DNNs used in the context of single-channel source separation in [10, 12, 13]. Besides, DNN_l is similar to the DNNs used in the context of single-channel speech enhancement in [38, 39] since they estimate clean spectra from the corresponding noisier spectra.

In this chapter, we consider features in the magnitude STFT domain as the inputs and outputs of DNNs. The inputs of DNN_0 and DNN_l are denoted by $\sqrt{z_x(f,n)}$

Algorithm 1 General iterative EM framework [27]

Inputs:

STFT of mixture $\mathbf{x}(f,n)$
Number of channels I and number of sources J
Number of EM iterations L
Spectral models M_0, M_1, …, M_J

for each source j of J **do**
  Initialize the spectrogram: $v_j(f,n) \leftarrow$ *spectrogram initialization*
  Initialize the spatial covariance matrix: $\mathbf{R}_j(f) \leftarrow I \times I$ identity matrix
end for
for each EM iteration l of L **do**
  Compute the mixture covariance matrix:
    $\mathbf{R}_\mathbf{x}(f,n) \leftarrow \sum_{j=1}^{J} v_j(f,n)\mathbf{R}_j(f)$
  for each source j of J **do**
    Compute the Wiener filter gain:
      $\mathbf{W}_j(f,n) \leftarrow$ (1.5) given $v_j(f,n)$, $\mathbf{R}_j(f)$, $\mathbf{R}_\mathbf{x}(f,n)$
    Compute the spatial image:
      $\widehat{\mathbf{c}}_j(f,n) \leftarrow$ (1.4) given $\mathbf{x}(f,n)$, $\mathbf{W}_j(f,n)$
    Compute the posterior second-order raw moment of the spatial image:
      $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \leftarrow$ (1.6) given $v_j(f,n)$, $\mathbf{R}_j(f)$, $\mathbf{W}_j(f,n)$, $\widehat{\mathbf{c}}_j(f,n)$
    Update the spatial covariance matrix:
      $\mathbf{R}_j(f) \leftarrow$ (1.7) given $v_j(f,n)$, $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$
    Compute the unconstrained spectrogram:
      $z_j(f,n) \leftarrow$ (1.8) given $\mathbf{R}_j(f)$, $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$
    Update the spectrogram:
      $v_j(f,n) \leftarrow$ *spectrogram fitting* given $z_j(f,n)$, M_j
  end for
end for
for each source j of J **do**
  Compute the final spatial image:
    $\widehat{\mathbf{c}}_j(f,n) \leftarrow$ (1.4) given all $v_j(f,n)$, all $\mathbf{R}_j(f)$, $\mathbf{x}(f,n)$
end for

Outputs:

All spatial source images $[\widehat{\mathbf{c}}_1(f,n),\ldots,\widehat{\mathbf{c}}_J(f,n)]$

and $\sqrt{z_j(f,n)}$, respectively. The outputs of both types of DNNs are denoted by $\sqrt{v_j(f,n)}$ and the training targets are denoted by $\sqrt{\widetilde{v}_j(f,n)}$. DNN_0 takes the magnitude spectrum $\sqrt{z_x(f,n)}$ and yields the initial magnitude spectra $\sqrt{v_j(f,n)}$ for all sources simultaneously. Then, DNN_l takes the estimated magnitude spectra $\sqrt{z_j(f,n)}$ of all sources and yields the improved magnitude spectra $\sqrt{v_j(f,n)}$ for all sources simultaneously.

Instead of alternately doing spatial and spectral parameter updates as in classical EM iterations (see Algorithm 1), the spatial parameters are updated several times before the spectral parameters are updated. This is motivated by the use of DNN for spectrogram initialization. The initial spectrograms should be close to the targets already, while the initial spatial covariance matrices are far.

The DNN-based iterative framework is described in Algorithm 2.

1.3.2 Cost functions

In this chapter, we present the use of the following cost functions for the DNN training. These cost functions measure the difference between the target $\sqrt{\widetilde{v}_j(f,n)}$ and the estimate $\sqrt{v_j(f,n)}$.

1. The *Itakura-Saito (IS) divergence* [40] is expressed as

$$\mathscr{D}_{\text{IS}} = \frac{1}{JFN} \sum_{j,f,n} \left(\frac{\widetilde{v}_j(f,n)}{v_j(f,n)} - \log \frac{\widetilde{v}_j(f,n)}{v_j(f,n)} - 1 \right). \tag{1.9}$$

   Since this metric is known to yield signals with good perceptual quality, it becomes a popular metric in the audio processing community, including for NMF-based audio source separation [40–42]. From the theoretical point of view of the framework presented in this chapter, this metric is attractive because it results in maximum likelihood (ML) estimation of the spectra [40] and the whole Algorithm 2 then achieves ML estimation.
2. The (generalized) *Kullback-Leibler (KL) divergence* [43] is expressed as

$$\mathscr{D}_{\text{KL}} = \frac{1}{JFN} \sum_{j,f,n} \left(\sqrt{\widetilde{v}_j(f,n)} \log \frac{\sqrt{\widetilde{v}_j(f,n)}}{\sqrt{v_j(f,n)}} - \sqrt{\widetilde{v}_j(f,n)} + \sqrt{v_j(f,n)} \right). \tag{1.10}$$

   This metric is also a popular choice for NMF-based audio source separation [40] and has been shown to be effective for DNN training [11].
3. The *Cauchy cost function* is expressed as

$$\mathscr{D}_{\text{Cau}} = \frac{1}{JFN} \sum_{j,f,n} \left(\frac{3}{2} \log \left(\widetilde{v}_j(f,n) + v_j(f,n) \right) - \log \sqrt{v_j(f,n)} \right). \tag{1.11}$$

   This metric has been proposed recently for NMF-based audio source separation and advocated as performing better than the IS divergence in some cases [44].
4. The *phase-sensitive (PS) cost function* is defined as

$$\mathscr{D}_{\text{PS}} = \frac{1}{2JFN} \sum_{j,f,n} |m_j(f,n)\widetilde{x}(f,n) - \widetilde{c}_j(f,n)|^2, \tag{1.12}$$

Algorithm 2 DNN-based iterative framework

Inputs:

STFT of mixture $\mathbf{x}(f,n)$
Number of channels I and number of sources J
Number of spatial updates K and number of EM iterations L
DNN spectral models DNN_0, DNN_1, …, DNN_L

Do pre-processing on the observed mixture:
$\sqrt{z_x(f,n)} \leftarrow \mathrm{preprocess}(\mathbf{x}(f,n))$
Initialize all source spectrograms simultaneously:
$[v_1(f,n),\ldots,v_J(f,n)] \leftarrow \mathrm{DNN}_0\left(\sqrt{z_x(f,n)}\right)^2$
for each source j of J **do**
  Initialize the spatial covariance matrix: $\mathbf{R}_j(f) \leftarrow I \times I$ identity matrix
end for
for each EM iteration l of L **do**
  for each spatial update k of K **do**
    Compute the mixture covariance matrix:
    $\mathbf{R}_\mathbf{x}(f,n) \leftarrow \sum_{j=1}^{J} v_j(f,n)\,\mathbf{R}_j(f)$
    for each source j of J **do**
      Compute the Wiener filter gain:
      $\mathbf{W}_j(f,n) \leftarrow$ (1.5) given $v_j(f,n)$, $\mathbf{R}_j(f)$, $\mathbf{R}_\mathbf{x}(f,n)$
      Compute the spatial image:
      $\widehat{\mathbf{c}}_j(f,n) \leftarrow$ (1.4) given $\mathbf{x}(f,n)$, $\mathbf{W}_j(f,n)$
      Compute the posterior second-order raw moment of the spatial image:
      $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \leftarrow$ (1.6) given $v_j(f,n)$, $\mathbf{R}_j(f)$, $\mathbf{W}_j(f,n)$, $\widehat{\mathbf{c}}_j(f,n)$
      Update the spatial covariance matrix:
      $\mathbf{R}_j(f) \leftarrow$ (1.7) given $v_j(f,n)$, $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$
    end for
  end for
  for each source j of J **do**
    Compute the unconstrained source spectrogram:
    $z_j(f,n) \leftarrow$ (1.8) given $\mathbf{R}_j(f)$, $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$
  end for
  Update all source spectrograms simultaneously:
  $[v_1(f,n),\ldots,v_J(f,n)] \leftarrow \mathrm{DNN}_l\left(\left[\sqrt{z_1(f,n)},\ldots,\sqrt{z_J(f,n)}\right]\right)^2$
end for
for each source j of J **do**
  Compute the final spatial image:
  $\widehat{\mathbf{c}}_j(f,n) \leftarrow$ (1.4) given all $v_j(f,n)$, all $\mathbf{R}_j(f)$, $\mathbf{x}(f,n)$
end for

Outputs:

All spatial source images $[\widehat{\mathbf{c}}_1(f,n),\ldots,\widehat{\mathbf{c}}_J(f,n)]$

where $m_j(f,n) = v_j(f,n)/\sum_{j'} v_{j'}(f,n)$ is the single-channel Wiener filter [8, 23], while $\widetilde{x}(f,n)$ and $\widetilde{c}_j(f,n)$ are the single-channel versions of the multichannel mixture $\mathbf{x}(f,n)$ and the multichannel ground truth source spatial images $\mathbf{c}_j(f,n)$, respectively. These single-channel signals can be obtained, for instance, by DS beamforming [45, 46]. This metric minimizes the error in the complex-valued STFT domain, as opposed to the error in the magnitude STFT domain as the other cost functions considered here.

5. The *mean squared error (MSE)* [40] is expressed as

$$\mathscr{D}_{\text{MSE}} = \frac{1}{2JFN} \sum_{j,f,n} \left(\sqrt{\widetilde{v}_j(f,n)} - \sqrt{v_j(f,n)} \right)^2. \tag{1.13}$$

This metric is the most widely used cost function for various optimization processes, including DNN training for regression tasks.

1.3.3 Weighted spatial parameter updates

We also consider a general form of spatial parameter update as

$$\mathbf{R}_j(f) = \left(\sum_{n=1}^{N} \omega_j(f,n) \right)^{-1} \sum_{n=1}^{N} \frac{\omega_j(f,n)}{v_j(f,n)} \widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n), \tag{1.14}$$

where $\omega_j(f,n)$ denotes the weight of source j for frequency bin f and frame n. When $\omega_j(f,n)$ is set to be equal for all time-frequency (TF) bins (f,n), e.g. $\omega_j(f,n) = 1$, it reduces to the exact EM formulation (1.7). When $\omega_j(f,n) = v_j(f,n)$, as used in [33, 34], it reduces to

$$\mathbf{R}_j(f) = \left(\sum_{n=1}^{N} v_j(f,n) \right)^{-1} \sum_{n=1}^{N} \widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n). \tag{1.15}$$

Experience shows that these weights are able to handle bad estimates $v_j(f,n)$. This weighting trick mitigates the importance of problematic underestimated TF bins. When $v_j(f,n)$ for a specific TF bin (f,n) is very low, its value of $v_j(f,n)^{-1}\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$ will be very big and cause a detrimental effect to the following computations (e.g. $\mathbf{R}_j(f)$ becomes ill-conditioned) and, ultimately, the performance. This weighting trick also increases the importance of high energy TF bins, whose value of $v_j(f,n)^{-1}\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$ is closer to the true $\mathbf{R}_j(f)$ on average.

1.4 Experimental evaluation

In this section, we present the application of the DNN-based iterative framework for speech enhancement in the context of the CHiME-3 Challenge [47] and music separation in the context of the SiSEC-2016 Campaign [48]. Section 1.4.1 describes the general system design which applies to both experiment categories. Then, Sections 1.4.2 and 1.4.3 present the specific framework settings and the experimental results for speech and music separation, respectively.

1.4.1 General system design

Framework

The framework can be divided into three main successive steps as follows.

Pre-processing This step is required to prepare the real-valued input of DNN_0 $\sqrt{z_x(f,n)}$ by deriving it from the complex-valued STFT coefficients of the multichannel mixture signal $\mathbf{x}(f,n)$.

Initialization In this step, the initial source PSDs are estimated simultaneously by DNN_0 given the input prepared above. Besides, the source spatial covariance matrices are initialized as $I \times I$ identity matrix.

Multichannel filtering The source PSDs and spatial covariance matrices are then re-estimated and updated using the iterative framework (Algorithm 2), in which DNN_l is employed for spectrogram fitting at iteration l. In order to avoid numerical instabilities due to the use of single precision, the PSDs $v_j(f,n)$ are floored to 10^{-5} in the parameter update iterations.

DNN spectral models

Four design aspects are discussed below: the architecture, the inputs and outputs, the training criterion, and the training algorithm.

Architecture

The DNNs follow a fully-connected feedforward network architecture. The number of hidden layers and the number of units in each input or hidden layer may vary. The number of units in the output layer equals the dimension of spectra multiplied by the number of sources. The activation functions of the hidden and output layers are rectified linear units (ReLUs) [49]. Other network architectures, e.g. recurrent neural network and convolutional neural network, may be used instead of the one

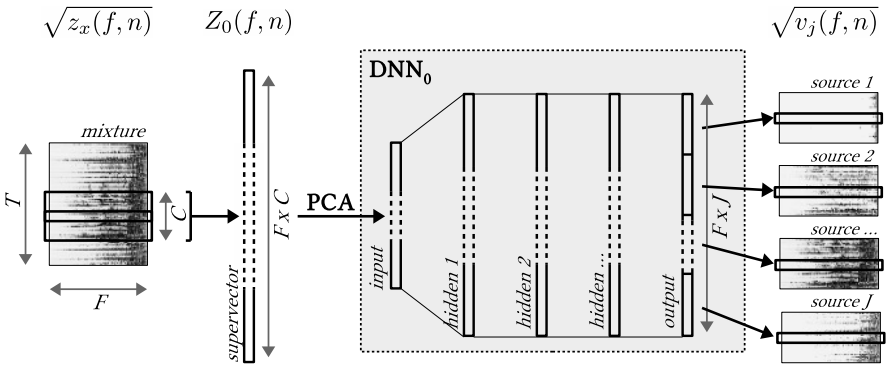


Fig. 1.2 Illustration of the inputs and outputs of the DNN for spectrogram initialization. Inputs: magnitude spectrum of the mixture (left). Outputs: magnitude spectra of the sources (right).

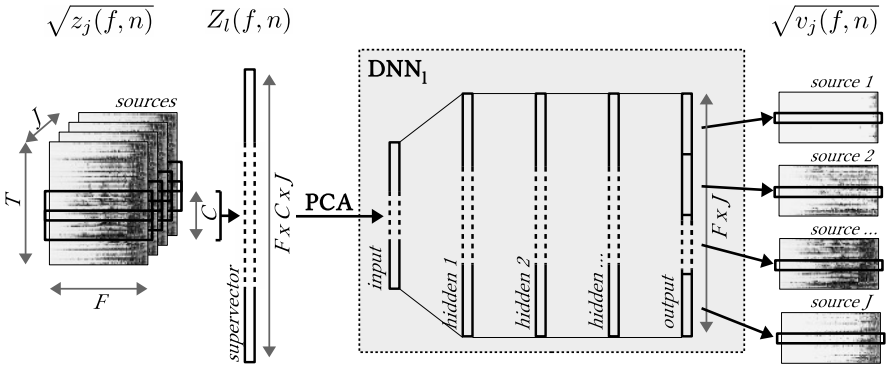


Fig. 1.3 Illustration of the inputs and outputs of the DNNs for spectrogram fitting. Inputs: stack of magnitude spectra of all sources (left). Outputs: magnitude spectra of the sources (right).

used here. The performance comparison with different architectures is beyond the scope of this chapter.

Inputs and outputs

In order to provide temporal context, the input frames are concatenated into *supervectors* consisting of a center frame, left context frames, and right context frames. In choosing the context frames, we use every second frame relative to the center frame in order to reduce the redundancies caused by the windowing of STFT. Although this causes some information loss, this enables the supervectors to represent a longer context [15, 50]. In addition, we do not use the magnitude spectra of the context frames directly, but the difference of magnitude between the context frames

and the center frame. These differences act as complementary features similar to delta features [26].

The dimension of the supervectors is reduced by principal component analysis (PCA) to the dimension of the DNN input. Dimensionality reduction by PCA significantly minimizes the computational cost of DNN training with a negligible effect on the performance of DNN provided enough components are kept [51]. Standardization (zero mean, unit variance) is done element-wise before and after PCA over the training data. The standardization factors and the PCA transformation matrix are then kept for pre-processing of any input. Thus, strictly speaking, the inputs of DNNs are not the supervectors of magnitude spectra $Z_0(f,n)$ and $Z_l(f,n)$ (see Fig. 1.2 and 1.3), but their transformation into reduced dimension vectors.

Figures 1.2 and 1.3 illustrates the inputs and outputs of the DNNs for spectrogram initialization and spectrogram fitting, respectively. F denotes the dimension of the spectra, $C = 2c+1$ the context length, and J the number of sources. In this chapter, we considered $c = 2$, so the supervectors for the input of the DNNs were composed by 5 time frames (2 left context, 1 center, and 2 right context frames).

Training criterion

Beside using a cost function from Section 1.3.2, an ℓ_2 weight regularization term is used to prevent overfitting [52]. It can be expressed as

$$\mathscr{D}_{\ell_2} = \frac{\lambda}{2}\sum_q w_q^2 \tag{1.16}$$

where w_q are the DNN weights and the regularization parameter is fixed to $\lambda = 10^{-5}$. No regularization is applied to the biases.

In order to avoid numerical instabilities, our implementation used regularized formulations of IS, KL, and Cauchy costs by adding the regularization parameter $\delta_{cf} = 10^{-3}$ in the logarithm computation. It should be noted that the use of regularization in this case is a common practice to avoid instabilities [42, 53]. In addition, geometric analysis on the PS cost function by considering that $m_j(f,n) \in \mathbb{R}_+^{F\times N}$ leads to a simplified formula as

$$\mathscr{D}_{\overline{\mathrm{PS}}} = \frac{1}{2JFN}\sum_{j,f,n}\left(m_j(f,n)\,|\widetilde{x}(f,n)| - \left|\widetilde{c}_j(f,n)\right|\cos\left(\angle\widetilde{x}(f,n) - \angle\widetilde{c}_j(f,n)\right)\right)^2, \tag{1.17}$$

where $\angle\cdot$ denotes the angle of complex-valued STFT spectra. See [26] for further implementation details.

Training algorithm

Following [54], the weights are initialized randomly from a zero-mean Gaussian distribution with standard deviation of $\sqrt{2/n_l}$, where n_l is the number of inputs to the neuron and, in this case, equals to the size of the previous layer. The biases are initialized to zero.

The DNNs are trained by greedy layer-wise supervised training [55] where the hidden layers are added incrementally. In the beginning, a network with one hidden layer is trained after random weight initialization. The output layer of this trained network is then substituted by new hidden and output layers to form a new network, while the parameters of the existing hidden layer are kept. Thus, we can view this as a pre-training method for the training of a new deeper network. After random initialization for the parameters of new layers, the new network is entirely trained. This procedure is done iteratively until the target number of hidden layers is reached.

Training is done by backpropagation with minibatches size of 100 and the ADADELTA parameter update algorithm [56]. Compared to standard stochastic gradient descent (SGD), ADADELTA employs adaptive dimension-wise learning rates and does not require manual setting of the learning rate. The hyperparameters of ADADELTA are set to $\rho = 0.95$ and $\varepsilon = 10^{-6}$ following [56]. The validation error is computed every epoch and the training is stopped after 10 consecutive epochs failed to obtain better validation error. The latest model which yields the best validation error is kept. Besides, the maximum number of training epochs is set to 100.

1.4.2 Application: speech enhancement

Task and dataset

We consider the problem of speech enhancement in the context of the CHiME-3 Challenge. This speech separation and recognition challenge considers the use of ASR in real-world noisy environments for a multi-microphone tablet device. The challenge provides real and simulated 6-channel microphone array data in 4 varied noise settings (bus, cafe, pedestrian area, and street junction) divided into training, development, and test sets. The training set consists of 1,600 real and 7,138 simulated utterances (`tr05_real` and `tr05_simu`), the development set consists of 1,640 real and 1,640 simulated utterances (`dt05_real` and `dt05_simu`), while the test set consists of 1,320 real and 1,320 simulated utterances (`et05_real` and `et05_simu`). The utterances are taken from the 5k vocabulary subset of the Wall Street Journal corpus [57]. All data are sampled at 16 kHz. For further details, please refer to [47]. In short, we deal with the separation of two sources ($J = 2$), namely speech and noise, from a 6-channel mixture ($I = 6$).

We used the source separation performance metrics defined in the BSS Eval toolbox 3.0[1] [58] in most of the experiments presented in this section. The metrics include signal-to-distortion ratio (SDR), source-image-to-spatial-distortion ratio (ISR), signal-to-interference ratio (SIR), and signal-to-artifacts ratio (SAR). In addition, at the end of this section, we use the best speech enhancement system as the front-end, combine it with the best back-end in [34], and evaluate the ASR performance in terms of word error rate (WER).

The multichannel ground truth speech and noise signals for the real data were extracted using the baseline simulation tool provided by the challenge organizers [47]. These signals are not perfect because they are extracted based on an estimation of the impulse responses between the close-talking microphone and the microphones on the tablet device. Since the resulting source separation performance metrics for the real data are unreliable, we evaluate the separation performance on the simulated data, which has reliable ground truth signals, for studying the impact of the different design choices. Nevertheless, since the ground truth transcriptions for ASR are reliable, we evaluate the ASR performance on real data.

Algorithm settings

The DNN-based multichannel speech enhancement framework is depicted in Fig. 1.4. Following [19], the input of DNN_0 $\sqrt{z_x(f,n)} = |\widetilde{x}(f,n)|$ was the magnitude of single-channel signals obtained from the multichannel noisy signals $\mathbf{x}(f,n)$ by DS beamforming [45, 46]. In doing so, the time-varying time difference of arrivals (TDOAs) between the speaker's mouth and each of the microphones are first measured using the provided baseline speaker localization tool [47], which relies on a nonlinear variant of steered response power using the phase transform (SRP-PHAT) [59, 60]. All channels are then aligned with each other by shifting the phase of STFT of the input noisy signal $\mathbf{x}(f,n)$ in all time-frequency bins (f,n) by the opposite of the measured delay. This preprocessing is required to satisfy the model in (1.2) which assumes that the sources do not move over time. Finally, a single-channel signal is obtained by averaging the realigned channels together. On the output side, the estimated speech spatial image are averaged over channels to obtain a single-channel signal for the speech recognition evaluation, which empirically provided better ASR performance than the use of one of the channels. Likewise, the training target $\sqrt{\widetilde{v}_j(f,n)} = |\widetilde{c}_j(f,n)|$ was the magnitude of DS beamforming outputs applied on the multichannel ground truth speech and noise signals $\mathbf{c}_j(f,n)$. Recall that the ground truth signals for the real data are unreliable, thus the training target for the real data is not as clean as it should be.

The STFT coefficients were computed using a Hamming window of length 1024 and hopsize 512 resulting in $F = 513$ frequency bins. DNN_0 and DNN_l have a similar architecture. They have an input layer, three hidden layers, and an output

[1] `http://bass-db.gforge.inria.fr/bss_eval/`

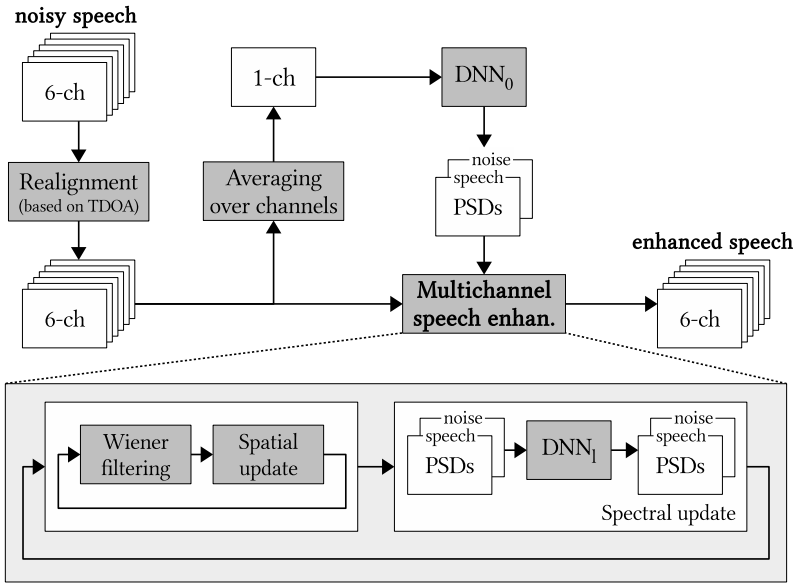


Fig. 1.4 DNN-based multichannel speech enhancement framework.

layer. Both types of DNNs have hidden and output layer size of $F \times J = 1026$. DNN_0 has an input layer size of $F = 513$ and DNN_l of $F \times J = 1026$.

The DNNs for *the source separation experiment* were trained on both the real and simulated training sets (`tr05_real` and `tr05_simu`) with the real and simulated development sets (`dt05_real` and `dt05_simu`) as validation data. Conversely, we trained the DNNs for *the speech recognition experiment* on the real training set only (`tr05_real`) and validated them on the real development set only (`dt05_real`). The same DNNs were also used for the performance comparison to the NMF-based iterative EM framework discussed in the end of this subsection. See [34] for the perfomance comparison between these two different training settings.

Impact of cost functions and spatial parameter updates

Figure 1.5 shows the performance comparison for different cost functions and also different numbers of spatial updates. In this case, the spectral parameters $v_j(f,n)$ are initialized by DNN_0 and kept fixed during the iterative procedure. In other words, the iteration only updates the spatial parameters. The performance metrics were computed on the resulting 6-channel estimated speech signals. The x-axis of each chart corresponds to the number of spatial updates k. Thus, $k = 0$ is equivalent to single-channel source separation for each channel.

In general, the performance increases along spatial updates. We observe that different cost functions behave differently along these updates. The performance of the PS cost is the best according to most metrics for the first few updates, but then it saturates quickly. On the contrary, the other cost functions are still getting better. Interestingly, after many updates (in this case, after $k = 20$), we can observe that some cost functions are better than the others for some metrics. Thus, the cost function selection should depend on the task (e.g. fewer artifacts are preferable to low interference) and the computational constraints (e.g. only few updates can be done). For general purposes, KL is the most reasonable choice because it improved all of the metrics well. Although IS is theoretically-motivated, there are better alternatives.

Impact of spectral parameter updates

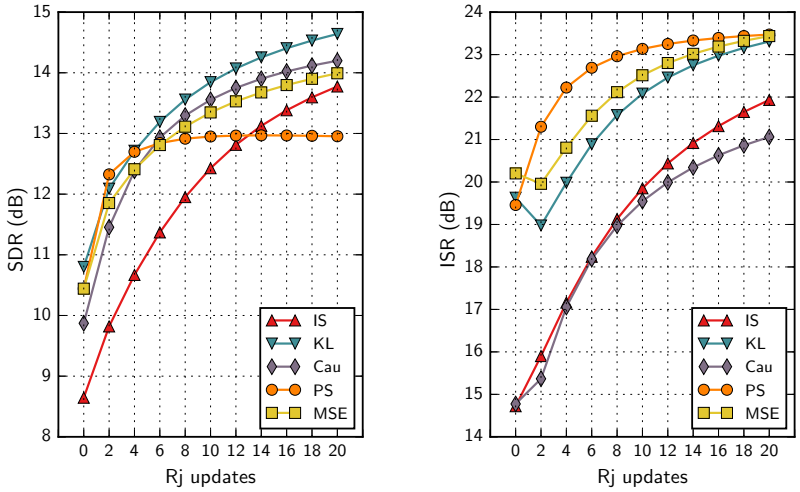
Figure 1.6 shows the performance comparison for different numbers of iterations after fixing the number of spatial updates $K = 20$. We trained two additional DNNs for spectrogram fitting, i.e. DNN_1 and DNN_2 for $l = 1$ and $l = 2$, respectively. This figure shows that generally the iterative spectral and spatial updates improve the performance, although we can observe that Cauchy and IS tend to saturate more quickly than other costs. Overall, the performance saturates after few iterations. Finally, the multichannel approach outperformed the single-channel DNN-based approach even when using DNN_0 only. Additional experiments where one DNN (namely DNN_1) was used for spectrogram fitting of multiple iterations are presented in [26].

Comparison to NMF-based iterative EM framework

In this subsection, we compare the performance of DNN-based framework described in this chapter to that of NMF-based framework [29]. We used the implementation of the latter framework found in the Flexible Audio Source Separation Toolbox (FASST)[2] and followed the settings used in [61]. The speech spectral and spatial models for this framework were trained on the real training set (`tr05_real`). Besides, the noise spectral and spatial models were initialized for each mixture using 5 seconds of background noise context based on the available annotation. This setting is favourable to the NMF-based framework. However, because of this, the comparison is not completely fair since the DNN-based framework does not exploit this context information. As described earlier, the DNNs used in this evaluation were also trained on the real training set only. The separation results from this evaluation were then used for the following speech recognition evaluation.

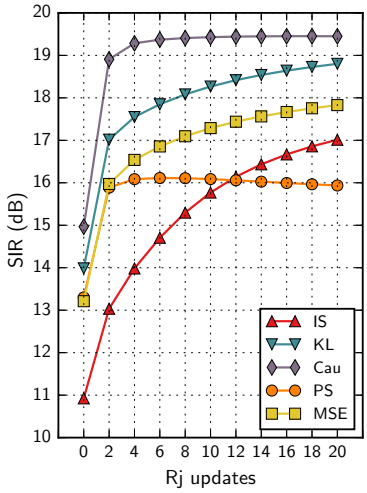
Table 1.1 compares the performance of the NMF-based framework after 50 EM iterations and the performance of the DNN-based framework after the spatial update

[2] http://bass-db.gforge.inria.fr/fasst

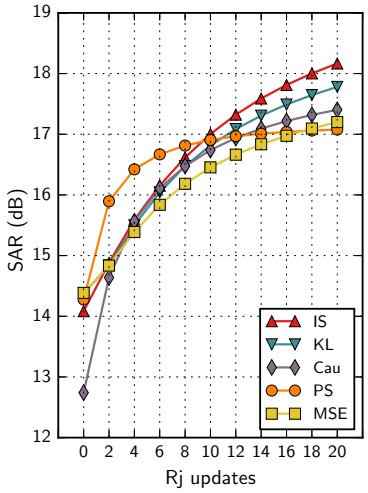


(a) SDR

(b) ISR

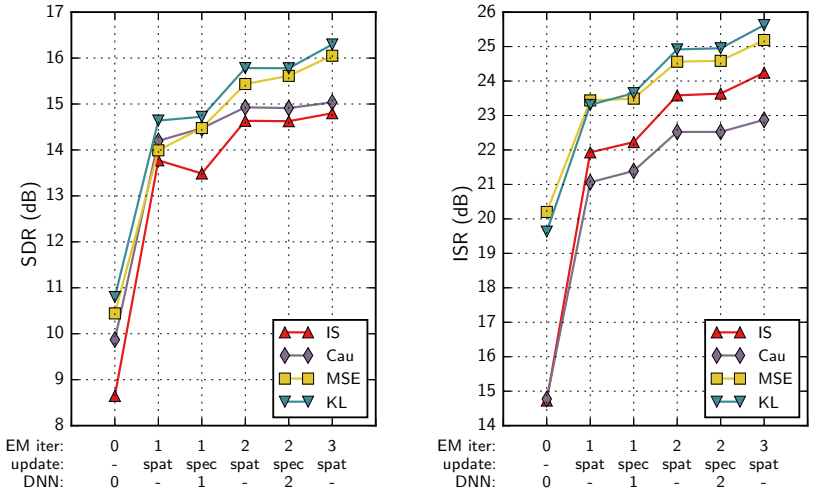


(c) SIR



(d) SAR

Fig. 1.5 Performance comparison for various numbers of spatial updates with the DNNs trained using different cost functions. The PSDs $v_j(f,n)$ are estimated by DNN_0 and the spatial covariance matrices $\mathbf{R}_j(f)$ are updated in the iterative procedure. The evaluation was done on the simulated test set (`et05_simu`). The figures show the mean value. Higher is better.



(a) SDR (b) ISR

(c) SIR (d) SAR

Fig. 1.6 Performance comparison for each update of the EM iterations with the DNNs trained using different cost functions. Different DNNs are used for each EM iteration. The spatial covariance matrices $\mathbf{R}_j(f)$ are updated with $K = 20$. The evaluation was done on the simulated test set (`et05_simu`). The figures show the mean value. Higher is better.

Table 1.1 Performance comparison in terms of source separation metrics (in dB). The evaluation was done on the simulated test set (`et05_simu`). The table shows the mean value. Higher is better.

Enhancement method	SDR	ISR	SIR	SAR
NMF-based iterative EM [29]	7.72	10.77	13.29	12.29
Proposed: KL (3 DNNs)	13.25	24.25	15.58	18.23

Table 1.2 Average WERs (%) using the DNN+sMBR back-end trained with enhanced multi-condition data followed by 5-gram KN smoothing and RNN-LM rescoring. The evaluation was done on the real sets. Boldface numbers show the best performance for each dataset. Lower is better.

Enhancement method	EM iter.	Update type	Dev	Test
Observed	-	-	9.65	19.28
DS beamforming	-	-	6.35	13.70
NMF-based [29]	50	-	6.10	13.41
DNN-based: KL (3 DNNs)	0	-	6.64	15.18
	1	spatial	5.37	11.46
		spectral	5.19	11.46
	2	spatial	**4.87**	10.79
		spectral	4.99	11.12
	3	spatial	4.88	**10.14**

of the EM iteration $l = 3$. The DNN-based framework is clearly better than the NMF-based iterative EM framework for all metrics. This confirms that DNNs are able to model spectral parameters much better than NMF does.

Table 1.2 shows the speech recognition evaluation results in terms of word error rate (WER). This evaluation followed the Kaldi setup distributed by the CHiME-3 challenge organizers[3] [47, 62]. It includes the use of (a) feature-space maximum likelihood regression features [63]; (b) acoustic models based on Gaussian Mixture Model and DNN trained with the cross entropy criterion followed by state-level minimum Bayes risk (sMBR) criterion [64]; and (c) language models with 5-gram Kneser-Ney (KN) smoothing [65] and rescoring using recurrent neural network-based language model (RNN-LM) [66]. The acoustic models are trained on enhanced multi-condition real and simulated data. See [62] for the further details of the methods used in the evaluation. It should be noted that we did not do any further optimization on the speech recognition back-end.

The evaluation results include the baseline performance (observed), DS beamforming, and NMF-based iterative EM framework [29]. The baseline performance was measured using only channel 5 of the observed 6-channel mixture. This channel is considered as the most useful channel because the corresponding microphone faces the user and is located at the bottom-center of the tablet device. DS beamforming was performed on the 6-channel mixture. For both the NMF-based and DNN-based frameworks, we compute the average over channels of the separation results from the earlier source separation evaluation.

[3] https://github.com/kaldi-asr/kaldi/tree/master/egs/chime3

For the DNN-based single-channel enhancement (see EM iteration $l = 0$), the WER on the real test set decreases by 21% relative w.r.t. the observed WER. This single-channel enhancement takes the output of DS beamforming on the 6-channel mixture. However, this single-channel enhancement did not provide better performance compared to the DS beamforming alone. It indicates that proper exploitation of multichannel information is crucial. The DNN-based multichannel enhancement then decreases the WER on the real test set by 33% relative w.r.t. the corresponding single-channel enhancement, 26% relative w.r.t. the DS beamforming alone, and 24% relative w.r.t. the NMF-based iterative EM framework [29].

1.4.3 Application: music separation

Task and dataset

We also consider the problem of music separation in the context of the professionally-produced music recordings task (labeled as 'MUS') of SiSEC 2016. In this task, we want to separate music recordings into their constitutive sources, namely *vocals*, *bass*, *drums*, and *other*. The dataset, called Demixing Secrets Dataset (DSD100)[4], comprises 100 full-track songs of diverse music genres by various artists with their corresponding sources. All mixtures and sources are stereo signals sampled at 44.1 kHz. The dataset is divided evenly into development and evaluation sets. In short, we deal with the separation of four sources ($J = 4$) from a stereo mixture ($I = 2$).

Algorithm settings

The DNN-based music separation framework is depicted in Fig. 1.7. In this evaluation, we used one DNN for spectrogram initialization and another DNN for spectrogram fitting, namely DNN_0 and DNN_1, respectively.

The STFT coefficients were computed using a Hamming window of length 2048 and hopsize 1024 resulting in $F = 1025$ frequency bins. DNN_0 has an input layer size of 2050 with three hidden layers, while DNN_1 has an input layer size of 4100 with two hidden layers. These settings are chosen based on preliminary experiments and computational considerations. The hidden layers and output layers of both DNNs have a size of $F \times J = 4100$. Dropout [67] with a rate 0.5 is implemented for all hidden layers.

The input of DNN_0 $\sqrt{z_x(f,n)}$ was derived from the multichannel mixture signal $\mathbf{x}(f,n)$ as

[4] See MUS 2016 task on `http://sisec.inria.fr`.

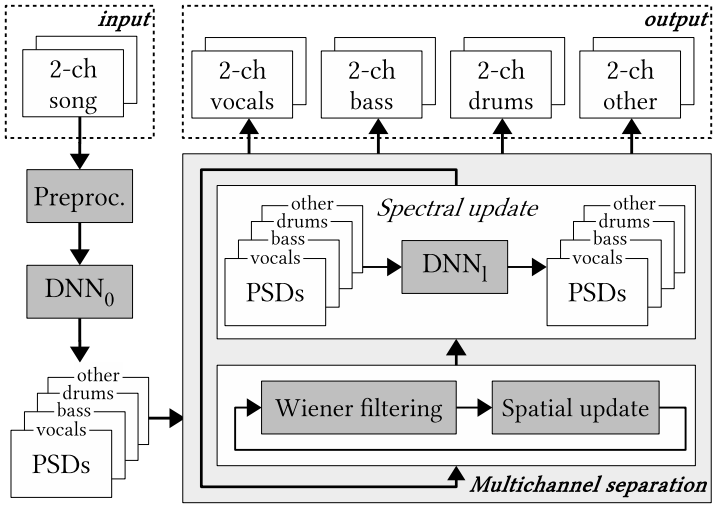


Fig. 1.7 DNN-based multichannel music separation framework.

$$\sqrt{z_x(f,n)} = \sqrt{\frac{1}{I}\|\mathbf{x}(f,n)\|^2}. \tag{1.18}$$

The training target $\sqrt{\widetilde{v}_j(f,n)}$ was derived from the true source spatial image $\mathbf{c}_j(f,n)$ as

$$\sqrt{\widetilde{v}_j(f,n)} = \sqrt{\frac{1}{I}\mathrm{tr}\left(\widetilde{\mathbf{R}}_j(f)^{-1}\mathbf{c}_j(f,n)\mathbf{c}_j^H(f,n)\right)}, \tag{1.19}$$

where $\widetilde{\mathbf{R}}_j(f)$ is an estimate of the true spatial covariance matrix computed as

$$\widetilde{\mathbf{R}}_j(f) = \frac{I}{N}\sum_{n=1}^{N}\frac{\mathbf{c}_j(f,n)\mathbf{c}_j^H(f,n)}{\|\mathbf{c}_j(f,n)\|^2}. \tag{1.20}$$

Compared to that was done in the speech enhancement task (Section 1.4.2), this provides better targets for the sources which are not mixed to the center (corresponding to $\widetilde{\mathbf{R}}_j(f) = \mathbf{I}$), e.g. *drums* and *other*, and consequently allows the DNN to provide better estimates.

We randomly divided the supervectors ($Z_0(f,n)$ and $Z_l(f,n)$ in Fig. 1.2 and 1.3) of each song from the development set into training and validation sets with a ratio of 8 to 2. By doing so, these two sets contains *different* parts of the same set of songs. However, the fact that these parts come from the same set of songs makes the DNN training prone to overfitting. Another data splitting scheme is by dividing the available 50 development songs, for example, into 40 songs for training and 20 songs for validation (note that, we keep the training-validation ratio of 8 to 2). Using this scheme, we observed that the early stopping mechanism of the DNN training

was triggered too early resulting a DNN with poor performance. The cost function used for DNN training is MSE with an ℓ_2 regularization term.

In addition, after every spatial parameter update in the multichannel iteration procedure, the parameter is normalized and regularized with $\delta_{\mathbf{R}} = 10^{-5}$ as

$$\mathbf{R}_j(f) = \frac{I}{\mathrm{tr}(\mathbf{R}_j(f))}\mathbf{R}_j(f) + \delta_{\mathbf{R}}\mathbf{I}. \tag{1.21}$$

Impact of weighted spatial parameter updates

Figure 1.8 shows the impact of different spatial parameter update strategies, namely 'exact', 'weighted', and 'weighted+simplified', on the performance in terms of SDR. The performance is computed on all songs on 30 seconds excerpts, taken every 15 seconds. The differences between these strategies are listed below.

- 'exact': $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \leftarrow$ (1.6), $\mathbf{R}_j(f) \leftarrow$ (1.7) (as in [27, 29] and the speech enhancement experiment discussed in Section 1.4.2)
- 'weighted': $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \leftarrow$ (1.6), $\mathbf{R}_j(f) \leftarrow$ (1.15)
- 'weighted+simplified': $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) = \widehat{\mathbf{c}}_j(f,n)\widehat{\mathbf{c}}_j(f,n)^H$; $\mathbf{R}_j(f) \leftarrow$ (1.15) (as in [33, 34])

The results show that the 'exact' strategy fails to improve the performance. It should be noted that in the oracle setting, in which $v_j(f,n)$ is computed from the true source image, this 'exact' strategy works well. Hence, it does not work in this case probably because of $v_j(f,n)$ is badly estimated by the DNN. Following this reasoning, the 'weighted' and 'weighted+simplified' strategies show that the weighted spatial parameter updates handle bad estimation of $v_j(f,n)$ effectively. We observe that 'weighted' is more robust to the setting of K than 'weighted+simplified'. This also shows that the inclusion of prior information in the computation of $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)$ allows the system to be more stable.

It is also worth mentioning that the 'exact' strategy works for our speech enhancement task (Section 1.4.2). This might be because, in that task, we dealt with fewer sources, fewer frequency bins, and more training data which lead to better DNNs providing better estimation of $v_j(f,n)$. In addition, when the 'weighted' strategy is used in that speech enhancement task followed by the speech recognition evaluation, we observed an improvement of WER up to 2% absolute.

Comparison to various music separation techniques

Figure 1.9 shows the performance comparison between four systems based on the framework described in this chapter (`NUG{1,2,3,4}`) and other music separation techniques. `NUG1` and `NUG3` correspond to 'weighted+simplified' after spatial up-

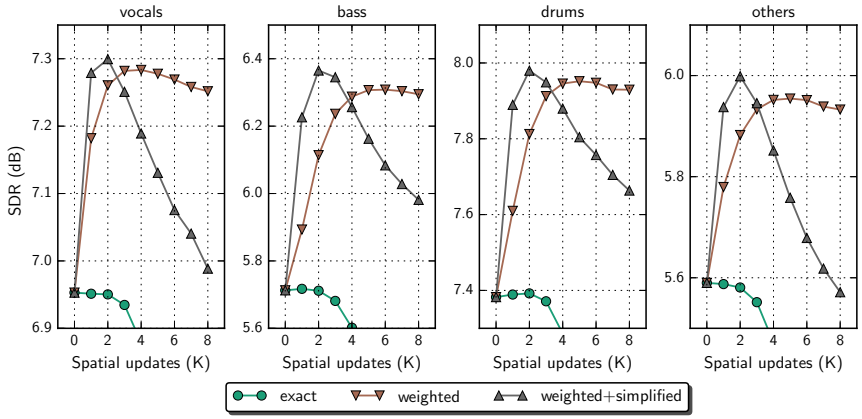


Fig. 1.8 Performance comparison on the *development* set for various numbers of spatial updates with different parameter updates. The PSDs $v_j(f,n)$ are initialized by DNN_0 and the spatial covariance matrices $\mathbf{R}_j(f)$ are updated in the iterative procedure. Higher is better.

dates of EM iterations 1 and 2 with $K = 2$, respectively. Similarly, `NUG2` and `NUG4` correspond to 'weighted' with $K = 4$. To be clear, `NUG3` and `NUG4` used additional DNNs for spectrogram fitting. These systems are compared to the other techniques listed below. See [68] for the implementation details of these techniques.

- Matrix factorization systems include `OZE` [29], `DUR` [69], and `HUA`[70]
- `KAM{1,2}` are variants of KAM [32]
- `RAF{1,2,3}` are variants of REPET [71–73]
- `UHL{1,2}` are variants of the DNN-based method in [15]

Among these other techniques, `UHL{1,2}` are also DNN-based ones. We can observe that DNN-based techniques performed better than the classical techniques. At a glance, the performance of `NUG{1,2,3,4}` is comparable to that of `UHL{1,2}`. However, we could provide lower spatial error (`NUG{3,4}`), interference (`NUG3`), or artifact `NUG{1,2}`. The most important difference between these systems is that `UHL{1,2}` do single-channel filtering, instead of multichannel filtering. This shows that we can also benefit from multichannel filtering in a music separation task.

1.5 Closing remarks

This chapter focused on the use of DNNs for multichannel audio source separation. The separation framework combines DNNs to model the source spectra and the classical multichannel Gaussian model to exploit the spatial information. Experimental results demonstrated the effectiveness of this framework for both speech enhancement and music separation tasks. Beside assessing the source separation per-

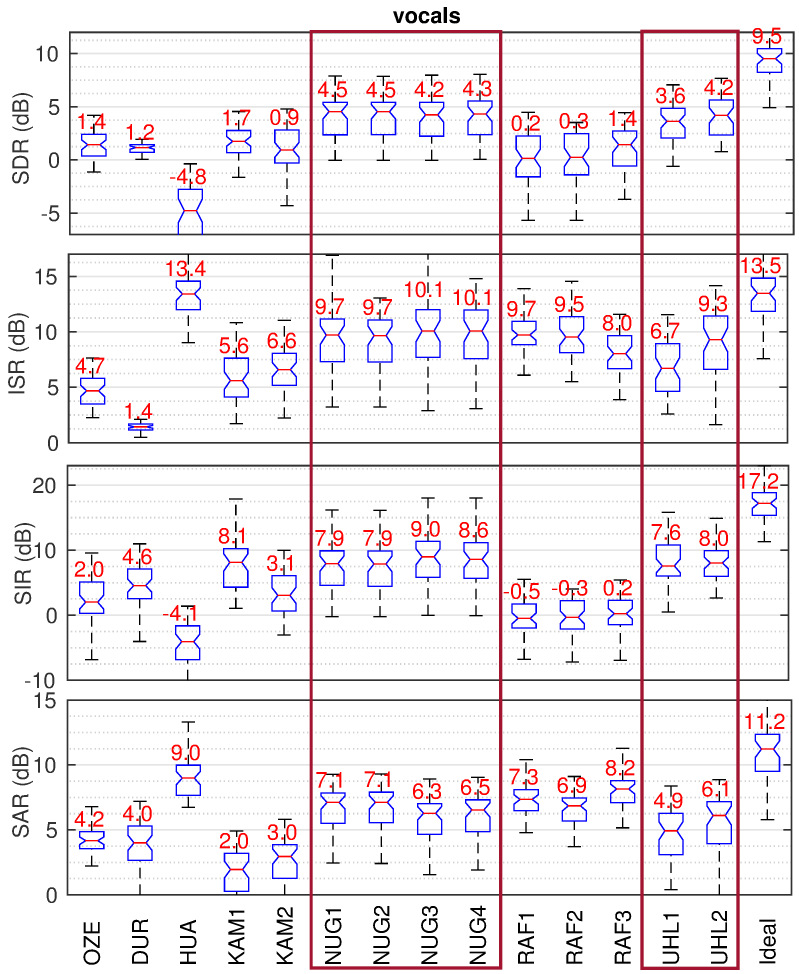


Fig. 1.9 Performance comparison on the *vocals* of *evaluation* set. The numbers shown above boxplots indicate the median values. Higher is better. The systems shown inside the red boxes are based on DNNs. `NUG{1,2,3,4}` are multichannel separation systems based on the framework described in this chapter, while `UHL{1,2}` are single-channel separation systems as in [15].

formance, the speech recognition performance was also evaluated for the first task. Several design choices and their comparative importance are presented. Finally, the results show the benefit of the presented DNN-based multichannel approach over a single-channel DNN-based approach and multichannel NMF.

Acknowledgment

The authors would like to thank the developers of Theano [74] and Kaldi [75]. Experiments presented in this article were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

References

1. S. Makino, H. Sawada, and T.-W. Lee, Eds., *Blind Speech Separation*, ser. Signals and Communication Technology. Dordrecht, The Netherlands: Springer, 2007.
2. M. Wölfel and J. McDonough, *Distant Speech Recognition*. Chichester, West Sussex, UK: Wiley, 2009.
3. T. Virtanen, R. Singh, and B. Raj, Eds., *Techniques for Noise Robustness in Automatic Speech Recognition*. Chicester, West Sussex, UK: Wiley, 2012.
4. G. R. Naik and W. Wang, Eds., *Blind Source Separation: Advances in Theory, Algorithms and Applications*, ser. Signals and Communication Technology. Berlin, Germany: Springer, 2014.
5. E. Vincent, N. Bertin, R. Gribonval, and F. Bimbot, "From blind to guided audio source separation: How models and side information can improve the separation of sound," *IEEE Signal Process. Mag.*, vol. 31, no. 3, pp. 107–115, May 2014.
6. L. Deng and D. Yu, "Deep learning: Methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3-4, pp. 197–387, 2014.
7. 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: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
8. F. Weninger, H. Erdogan, S. Watanabe, E. Vincent, J. Le Roux, J. R. Hershey, and B. Schuller, "Speech enhancement with LSTM recurrent neural networks and its application to noise-robust ASR," in *Proc. Int. Conf. Latent Variable Analysis Signal Separation (LVA/ICA)*, Liberec, Czech Republic, Aug. 2015, pp. 91–99.
9. J. Chen, Y. Wang, and D. Wang, "A feature study for classification-based speech separation at low signal-to-noise ratios," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 12, pp. 1993–2002, Dec. 2014.
10. Y. Tu, J. Du, Y. Xu, L. Dai, and C.-H. Lee, "Speech separation based on improved deep neural networks with dual outputs of speech features for both target and interfering speakers," in *Proc. Int. Symp. Chinese Spoken Lang. Process. (ISCSLP)*, Singapore, Sept. 2014, pp. 250–254.
11. P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Singing-voice separation from monaural recordings using deep recurrent neural networks," in *Proc. Int. Soc. Music Inf. Retrieval (ISMIR)*, Taipei, Taiwan, Oct. 2014, pp. 477–482.
12. ——, "Deep learning for monaural speech separation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Florence, Italy, May 2014, pp. 1562–1566.
13. ——, "Joint optimization of masks and deep recurrent neural networks for monaural source separation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 12, pp. 2136–2147, Dec. 2015.
14. S. Araki, T. Hayashi, M. Delcroix, M. Fujimoto, K. Takeda, and T. Nakatani, "Exploring multi-channel features for denoising-autoencoder-based speech enhancement," in *Proc. IEEE*

Int. Conf. Acoust. Speech Signal Process. (ICASSP), Brisbane, Australia, Apr. 2015, pp. 116–120.
15. S. Uhlich, F. Giron, and Y. Mitsufuji, "Deep neural network based instrument extraction from music," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 2135–2139.
16. Y. Wang and D. Wang, "Towards scaling up classification-based speech separation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 7, pp. 1381–1390, July 2013.
17. A. Narayanan and D. Wang, "Ideal ratio mask estimation using deep neural networks for robust speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Vancouver, Canada, May 2013, pp. 7092–7096.
18. Y. Jiang, D. Wang, R. Liu, and Z. Feng, "Binaural classification for reverberant speech segregation using deep neural networks," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 12, pp. 2112–2121, Dec. 2014.
19. F. Weninger, J. Le Roux, J. R. Hershey, and B. Schuller, "Discriminatively trained recurrent neural networks for single-channel speech separation," in *Proc. IEEE Global Conf. Signal Information Process. (GlobalSIP)*, Atlanta, GA, USA, Dec. 2014, pp. 577–581.
20. A. Narayanan and D. Wang, "Improving robustness of deep neural network acoustic models via speech separation and joint adaptive training," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 1, pp. 92–101, Jan. 2015.
21. Y. Wang and D. Wang, "A deep neural network for time-domain signal reconstruction," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 4390–4394.
22. D. S. Williamson, Y. Wang, and D. Wang, "Complex ratio masking for monaural speech separation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 3, pp. 483–492, Mar. 2016.
23. H. Erdogan, J. R. Hershey, S. Watanabe, and J. Le Roux, "Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 708–712.
24. X. Xiao, S. Watanabe, H. Erdogan, L. Lu, J. Hershey, M. L. Seltzer, G. Chen, Y. Zhang, M. Mandel, and D. Yu, "Deep beamforming networks for multi-channel speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 5745–5749.
25. J. Heymann, L. Drude, and R. Haeb-Umbach, "Neural network based spectral mask estimation for acoustic beamforming," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 196–200.
26. A. A. Nugraha, A. Liutkus, and E. Vincent, "Multichannel audio source separation with deep neural networks," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 9, pp. 1652–1664, Sept. 2016.
27. N. Q. K. Duong, E. Vincent, and R. Gribonval, "Under-determined reverberant audio source separation using a full-rank spatial covariance model," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 18, no. 7, pp. 1830–1840, July 2010.
28. N. Q. K. Duong, H. Tachibana, E. Vincent, N. Ono, R. Gribonval, and S. Sagayama, "Multichannel harmonic and percussive component separation by joint modeling of spatial and spectral continuity," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Prague, Czech Republic, May 2011, pp. 205–208.
29. A. Ozerov, E. Vincent, and F. Bimbot, "A general flexible framework for the handling of prior information in audio source separation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 4, pp. 1118–1133, May 2012.
30. T. Gerber, M. Dutasta, L. Girin, and C. Févotte, "Professionally-produced music separation guided by covers," in *Proc. Int. Soc. Music Inf. Retrieval (ISMIR)*, Porto, Portugal, Oct. 2012, pp. 85–90.
31. M. Togami and Y. Kawaguchi, "Simultaneous optimization of acoustic echo reduction, speech dereverberation, and noise reduction against mutual interference," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 11, pp. 1612–1623, Nov. 2014.

32. A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet, "Kernel Additive Models for source separation," *IEEE Trans. Signal Process.*, vol. 62, no. 16, pp. 4298–4310, Aug. 2014.
33. A. Liutkus, D. Fitzgerald, and Z. Rafii, "Scalable audio separation with light Kernel Additive Modelling," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 76–80.
34. S. Sivasankaran, A. A. Nugraha, E. Vincent, J. A. Morales-Cordovilla, S. Dalmia, I. Illina, and A. Liutkus, "Robust ASR using neural network based speech enhancement and feature simulation," in *Proc. IEEE Automat. Speech Recognition and Understanding Workshop (ASRU)*, Scottsdale, AZ, USA, Dec. 2015, pp. 482–489.
35. A. A. Nugraha, A. Liutkus, and E. Vincent, "Multichannel music separation with deep neural networks," in *Proc. European Signal Process. Conf. (EUSIPCO)*, Budapest, Hungary, Aug. 2016, pp. 1748–1752.
36. J. O. Smith, *Spectral Audio Signal Processing*. W3K Publishing, 2011.
37. E. Vincent, M. G. Jafari, S. A. Abdallah, M. D. Plumbley, and M. E. Davies, "Probabilistic modeling paradigms for audio source separation," in *Machine Audition: Principles, Algorithms and Systems*, W. Wang, Ed. Hershey, PA, USA: IGI Global, 2011, ch. 7, pp. 162–185.
38. D. Liu, P. Smaragdis, and M. Kim, "Experiments on deep learning for speech denoising," in *Proc. ISCA INTERSPEECH*, Singapore, Sept. 2014, pp. 2685–2688.
39. Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal Process. Lett.*, vol. 21, no. 1, pp. 65–68, Jan. 2014.
40. C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural Comput.*, vol. 21, no. 3, pp. 793–830, Mar. 2009.
41. N. Bertin, C. Févotte, and R. Badeau, "A tempering approach for Itakura-Saito non-negative matrix factorization. with application to music transcription," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Taipei, Taiwan, Apr. 2009, pp. 1545–1548.
42. A. Lefèvre, F. Bach, and C. Févotte, "Online algorithms for nonnegative matrix factorization with the Itakura-Saito divergence," in *Proc. IEEE Workshop on Appl. of Signal Process. to Audio and Acoust. (WASPAA)*, New Paltz, NY, USA, Oct. 2011, pp. 313–316.
43. C. Févotte and A. Ozerov, "Notes on nonnegative tensor factorization of the spectrogram for audio source separation: statistical insights and towards self-clustering of the spatial cues," in *Proc. Int. Symp. on Comput. Music Modeling and Retrieval*, Málaga, Spain, June 2010, pp. 102–115.
44. A. Liutkus, D. Fitzgerald, and R. Badeau, "Cauchy nonnegative matrix factorization," in *Proc. IEEE Workshop on Appl. of Signal Process. to Audio and Acoust. (WASPAA)*, New Paltz, NY, USA, Oct. 2015, pp. 1–5.
45. J. McDonough and K. Kumatani, "Microphone arrays," in *Techniques for Noise Robustness in Automatic Speech Recognition*, T. Virtanen, R. Singh, and B. Raj, Eds. Chicester, West Sussex, UK: Wiley, 2012, ch. 6.
46. K. Kumatani, J. McDonough, and B. Raj, "Microphone array processing for distant speech recognition: From close-talking microphones to far-field sensors," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 127–140, Nov. 2012.
47. J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. IEEE Automat. Speech Recognition and Understanding Workshop (ASRU)*, Scottsdale, AZ, USA, Dec. 2015, pp. 504–511.
48. A. Liutkus, F.-R. Stöter, Z. Rafii, D. Kitamura, B. Rivet, N. Ito, N. Ono, and J. Fontecave, "The 2016 signal separation evaluation campaign," in *Proc. Int. Conf. Latent Variable Analysis Signal Separation (LVA/ICA)*, Grenoble, France, Feb. 2017, to be published.
49. X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier networks," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, vol. 15, Fort Lauderdale, FL, USA, Apr. 2011, pp. 315–323.
50. A. A. Nugraha, K. Yamamoto, and S. Nakagawa, "Single-channel dereverberation by feature mapping using cascade neural networks for robust distant speaker identification and speech recognition," *EURASIP J. Audio, Speech and Music Process.*, vol. 2014, no. 13, 2014.

51. X. Jaureguiberry, E. Vincent, and G. Richard, "Fusion methods for speech enhancement and audio source separation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 7, pp. 1266–1279, July 2016.
52. Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*, ser. Lecture Notes in Computer Science, G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Berlin, Germany: Springer, 2012, vol. 7700, ch. 19, pp. 437–478.
53. P. Sprechmann, A. M. Bronstein, and G. Sapiro, "Supervised non-negative matrix factorization for audio source separation," in *Excursions in Harmonic Analysis, Volume 4*, ser. Applied and Numerical Harmonic Analysis, R. Balan, M. Begu, J. J. Benedetto, W. Czaja, and K. A. Okoudjou, Eds. Switzerland: Springer, 2015, pp. 407–420.
54. K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *ArXiv e-prints*, Feb. 2015. [Online]. Available: http://arxiv.org/abs/1502.01852
55. Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Conf. Neural Information Processing Systems (NIPS)*, Vancouver, Canada, Dec. 2006, pp. 153–160.
56. M. D. Zeiler, "ADADELTA: An adaptive learning rate method," *ArXiv e-prints*, Dec. 2012. [Online]. Available: http://arxiv.org/abs/1212.5701
57. J. Garofalo, D. Graff, D. Paul, and D. Pallett, "CSR-I (WSJ0) Complete LDC93S6A," DVD, 2007, Philadelphia: Linguistic Data Consortium.
58. E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 4, pp. 1462–1469, July 2006.
59. B. Loesch and B. Yang, "Adaptive segmentation and separation of determined convolutive mixtures under dynamic conditions," in *Proc. Int. Conf. Latent Variable Analysis Signal Separation (LVA/ICA)*, Saint-Malo, France, 2010, pp. 41–48.
60. C. Blandin, A. Ozerov, and E. Vincent, "Multi-source TDOA estimation in reverberant audio using angular spectra and clustering," *Signal Processing*, vol. 92, no. 8, pp. 1950–1960, Aug. 2012.
61. Y. Salaün, E. Vincent, N. Bertin, N. Souviraà-Labastie, X. Jaureguiberry, D. T. Tran, and F. Bimbot, "The Flexible Audio Source Separation Toolbox Version 2.0," IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP), Florence, Italy, May 2014, Show & Tell. [Online]. Available: https://hal.inria.fr/hal-00957412
62. T. Hori, Z. Chen, H. Erdogan, J. R. Hershey, J. Le Roux, V. Mitra, and S. Watanabe, "The MERL/SRI system for the 3rd CHiME challenge using beamforming, robust feature extraction, and advanced speech recognition," in *Proc. IEEE Automat. Speech Recognition and Understanding Workshop (ASRU)*, Scottsdale, AZ, USA, Dec. 2015, pp. 475–481.
63. M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech & Language*, vol. 12, no. 2, pp. 75–98, Apr. 1998.
64. K. Veselý, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks," in *Proc. ISCA INTERSPEECH*, Lyon, France, Aug. 2013, pp. 2345–2349.
65. R. Kneser and H. Ney, "Improved backing-off for M-gram language modeling," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, vol. 1, Detroit, MI, USA, May 1995, pp. 181–184.
66. T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. ISCA INTERSPEECH*, Chiba, Japan, Sept. 2010, pp. 1045–1048.
67. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Research*, vol. 15, pp. 1929–1958, June 2014.
68. N. Ono, D. Kitamura, Z. Rafii, N. Ito, and A. Liutkus, "The 2015 signal separation evaluation campaign (SiSEC2015)," in *Proc. Int. Conf. Latent Variable Analysis Signal Separation (LVA/ICA)*, Liberec, Czech Republic, Aug. 2015.

69. J.-L. Durrieu, B. David, and G. Richard, "A musically motivated mid-level representation for pitch estimation and musical audio source separation," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 6, pp. 1180–1191, Oct. 2011.
70. P.-S. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson, "Singing-voice separation from monaural recordings using robust principal component analysis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Kyoto, Japan, Mar. 2012, pp. 57–60.
71. Z. Rafii and B. Pardo, "REpeating Pattern Extraction Technique (REPET): A simple method for music/voice separation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 1, pp. 73–84, Jan. 2013.
72. A. Liutkus, Z. Rafii, R. Badeau, B. Pardo, and G. Richard, "Adaptive filtering for music/voice separation exploiting the repeating musical structure," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Kyoto, Japan, Mar. 2012, pp. 53–56.
73. Z. Rafii and B. Pardo, "Music/voice separation using the similarity matrix," in *Proc. Int. Soc. Music Inf. Retrieval (ISMIR)*, Porto, Portugal, Oct. 2012, pp. 583–588.
74. Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *ArXiv e-prints*, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688
75. D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Veselý, "The Kaldi speech recognition toolkit," in *Proc. IEEE Automat. Speech Recognition and Understanding Workshop (ASRU)*, Hawaii, USA, Dec. 2011, pp. 1–4.