Multichannel audio source separation with deep neural networks

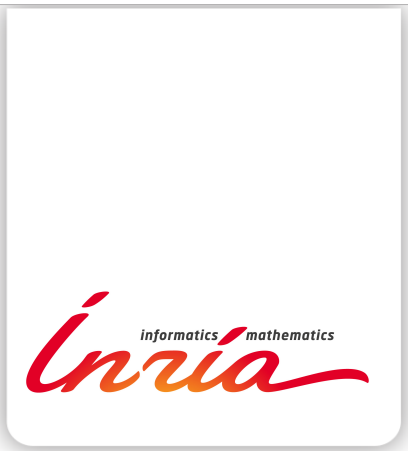
Aditya Arie Nugraha, Antoine Liutkus, Emmanuel Vincent

Multichannel audio source separation with deep neural networks

Aditya A. Nugraha, Antoine Liutkus, Emmanuel Vincent

RESEARCH REPORT
N° 8740
June 2015
Project-Team MULTISPEECH

ISSN 0249-6399 ISRN INRIA/RR--8740--FR+ENG



Multichannel audio source separation with deep neural networks

Aditya A. Nugraha, Antoine Liutkus, Emmanuel Vincent

Project-Team MULTISPEECH

Research Report n° 8740 — June 2015 — 15 pages

Abstract: This technical report considers the problem of multichannel audio source separation. A few studies have addressed the problem of single-channel audio source separation with deep neural networks (DNNs). We introduce a new framework for multichannel source separation where (1) spectral and spatial parameters are updated iteratively similarly to the expectation-maximization (EM) algorithm and (2) DNNs are used in the spectral updates. We evaluated several systems based on the proposed framework by participating in the "professionally-produced music recording" task of SiSEC 2015. Experimental results show that the framework performed well in separating singing voice and other instruments from a mixture containing multiple musical instruments.

Key-words: source separation, deep neural networks, expectation-maximization (EM) algorithm, SiSEC

RESEARCH CENTRE
NANCY – GRAND EST

615 rue du Jardin Botanique
CS20101
54603 Villers-lès-Nancy Cedex

Séparation de sources audio multicanale par réseaux de neurones profonds

Résumé : Ce rapport de recherche traite du problème de la séparation de sources audio multicanale. Quelques travaux ont traité le problème de la séparation de sources monocanale par réseaux de neurones profonds (DNNs). Nous présentons une nouvelle approche pour la séparation de sources multicanale où (1) les paramètres spectraux et spatiaux sont mis à jour itérativement de façon similaire à l'algorithme Espérance-Maximisation (EM) et (2) des DNNs sont utilisés pour la mise à jour des paramètres spectraux. Nous évaluons plusieurs systèmes basés sur cette approche en participant à la tâche "enregistrements musicaux professionnels" de SiSEC 2015. Les résultats montrent que cette approche fonctionne bien pour la séparation de la voix chantée et des autres instruments dans un mélange contenant plusieurs instruments.

Mots-clés : séparation de sources, réseaux de neurones profonds, algorithme Espérance-Maximisation (EM), SiSEC

1 Introduction

Audio source separation aims to recover the signals of underlying sound sources from an observed mixture signal. Recent research on source separation can be divided into speech separation and music separation. Speech separation mainly aims to recover the speech signal from a mixture containing background noise signals, while music separation aims to recover the singing voice and possibly other instruments from a mixture containing multiple musical instruments. Although the basic concepts are the same in either case, there are several properties that can be exploited for each specific problem. For example, exploiting the long-term repetitive structure of sound is only applicable for music separation.

Let I denote the number of channels, $\mathbf{x}(t)$ the observed I-channel mixture signal, $s_j(t)$ and $\mathbf{c}_j(t)$ the j-th source signal and its I-channel spatial image, respectively. The observed mixture signal can be expressed as

$$\mathbf{x}(t) = \sum_{j=1}^{J} \mathbf{c}_j(t) = \sum_{j=1}^{J} \sum_{\tau} \mathbf{h}_j(\tau) s_j(t-\tau) \tag{1}$$

where J denotes the number of sources and $\mathbf{h}_j(\tau)$ denotes the filter coefficients for source j, which may represent the room impulse response in the scenario of recording in a reverberant environment or a mixing function in the scenario of multitrack music recording.

In our work, instead of estimating the source signal $s_j(t)$, we estimate the source image $\mathbf{c}_j(t)$ which can be seen as a filtered version of $s_j(t)$. For the scenario of recording in a reverberant environment where the filter $\mathbf{h}_j(\tau)$ corresponds to the room impulse response, estimating $s_j(t)$ from $\mathbf{c}_j(t)$ is a dereverberation problem which is a major separate topic in signal processing research.

Let $\mathbf{c}_j(f,n)$ denote the short-time Fourier transform (STFT) coefficients of $\mathbf{c}_j(t)$ for time-frequency (T-F) bin (f,n). $\mathbf{c}_j(f,n)$ is assumed to have a multivariate complex isotropic Gaussian distribution [1, 2]

$$\mathbf{c}_j(f,n) \sim \mathcal{N}_c(0, v_j(f,n)\mathbf{R}_j(f)) \tag{2}$$

where $v_j(f,n)$ denotes the power spectral density (PSD), also known as spectrogram, of source j for T-F bin (f,n) and $\mathbf{R}_j(f)$ denotes the $I \times I$ spatial covariance matrix of source j for frequency bin f.

Assuming that the number of sources J is known, the estimated source image $\widehat{c}_j(f,n)$ can be calculated in the minimum mean square error (MMSE) sense using generalized multichannel Wiener filtering given the estimated PSDs $\widehat{v}_j(f,n)$ and the estimated spatial covariance matrices $\widehat{\mathbf{R}}_j(f)$ of all sources as

$$\widehat{\mathbf{c}}_j(f,n) = \widehat{v}_j(f,n)\widehat{\mathbf{R}}_j(f,n) \left(\sum_{j'=1}^{J} \widehat{v}_{j'}(f,n)\widehat{\mathbf{R}}_{j'}(f,n) \right)^{-1} \mathbf{x}(f,n) \tag{3}$$

where $\mathbf{x}(f,n)$ denotes the STFT coefficients of $\mathbf{x}(t)$ for T-F bin (f,n). Source separation then becomes the problem of estimating the PSD and the spatial covariance matrix of each source. Finally, $\widehat{\mathbf{c}}_j(t)$ is recovered from $\widehat{\mathbf{c}}_j(f,n)$ by inverse STFT.

Recent studies have shown that neural networks (NNs) trained by deep learning are able to model complex functions and perform well on various tasks, such as automatic speech recognition [3]. There are many variants of neural networks trained by deep learning, including deep neural network (DNN) and deep recurrent neural network (DRNN). However, the term "DNN" itself is defined differently in the literature. We follow the definition of DNNs in [4] where a DNN is

defined as a feed-forward, artificial neural network that has more than one layer of hidden units. In addition, a DNN that is pretrained generatively as a deep belief network (DBN) is called a DBN-DNN.

A few studies have addressed the problem of single-channel source separation with deep learning. DNNs are used to predict either T-F masks or the source spectrogram.

Narayanan and Wang [5] used DBN-DNNs combined with a multilayer perceptron (MLP) to estimate the ideal ratio mask (IRM) in the Mel spectral domain. The DBN-DNN and the MLP were trained for each output dimension. Weninger et al. [6] used DNNs and DRNNs with long short-term memory (LSTM) neurons to estimate the IRM in the Mel spectral domain. Instead of simply using a distance measure between the target mask and the estimated mask as the objective function, they used a distance measure between the target signal and the estimated signal, which is the input signal masked by the estimated mask.

Tu et al. [7] used a DBN-DNN to estimate the log-power spectrogram of two sources (target and interfering sources) simultaneously. Huang et al. [8, 9, 10] also used a DBN-DNN and a DRNN to estimate the spectrogram of two sources simultaneously, but they jointly optimized the time-frequency masking function so that the sum of the estimated sources is equal to the mixture. They experimented using magnitude and log Mel spectra. Uhlich et al. [11] used a DNN to estimate the magnitude spectrum of the target musical instrument.

From the studies mentioned above, [5, 6, 7, 8] discuss the speech separation problem, [9, 11] discuss the music separation problem, while [10] discusses both problems.

Motivated by these studies and considering that the use of DNNs for source separation is still a new topic of research, we want to explore the use of DNNs and the wisdom gained from more conventional source separation techniques which have been proved to perform well, e.g. techniques based on the iterative expectation-maximization (EM) algorithm [1], robust principal component analysis (PCA) [12], or kernel additive modelling (KAM) [13].

In this technical report, we introduce a framework for multichannel source separation where (1) spectral and spatial parameters are updated similarly to EM and (2) DNNs are used in the spectral updates.

The rest of this report is organized as follows. Section 2 describes the proposed framework. Section 3 presents the experimental setups and results. Finally, Section 4 concludes the report and presents future directions.

2 Proposed framework

2.1 Iterative procedure

Our approach builds upon the iterative procedure in [13] that is a computational simplification of the exact EM algorithm [14]. This iterative procedure can be divided into separation and fitting steps.

In the separation step, given the estimated parameters $\widehat{v}_j(f,n)$ and $\widehat{\mathbf{R}}_j(f)$ of each source, the source image estimates $\widehat{\mathbf{c}}_j(f,n)$ are obtained by multichannel Wiener filtering. Instead of Equation (3), regularized multichannel Wiener filtering is used:

$$\widehat{\mathbf{c}}_j(f,n) = \widehat{v}_j(f,n)\widehat{\mathbf{R}}_j(f,n)\left(\sum_{j'=1}^{J}\widehat{v}_{j'}(f,n)\widehat{\mathbf{R}}_{j'}(f,n) + \delta_1\mathbf{I}_I\right)^{-1}\mathbf{x}(f,n) \tag{4}$$

where $\mathbf{I}_I$ is the $I \times I$ identity matrix and δ_1 is the regularization coefficient.

In the fitting step, given the source image estimates $\widehat{\mathbf{c}}_j(f,n)$, the parameters $\widehat{v}_j(f,n)$ and $\widehat{\mathbf{R}}_j(f)$ are updated.

In our framework, the fitting step can be divided into spectral and spatial updates. Spectral updates are done by DNNs to estimate the PSDs $\widehat{v}_j(f,n)$, while spatial updates are done in the same way or in a similar way to the maximum likelihood (ML) update for the spatial covariance matrices $\widehat{\mathbf{R}}_j(f)$ in [14]. The general iterative procedure is described in Algorithm 1.

Algorithm 1 General iterative procedure.

1. **Input**:
   - Mixture STFT $\mathbf{x}(f,n)$
   - Number L of iterations
2. **Initialization**:
   - $l \leftarrow 1$
   - initialize $\widehat{v}_j(f,n)$ (depends on the system)
   - $\widehat{\mathbf{R}}_j(f) \leftarrow I \times I$ identity matrix
3. For each source j:
   A. Separation step: compute $\widehat{\mathbf{c}}_j(f,n)$ by Equation (4) where δ_1 depends on the system
   B. Fitting step:
      i. Update the spatial parameters $\widehat{\mathbf{R}}_j(f)$ (depends on the system)
      ii. (Optional) Update the spectral parameters $\widehat{v}_j(f,n)$ (depends on the system)
4. If $l < L$ then set $l \leftarrow l+1$ and go to step 3
5. **Output**: source estimates $\widehat{\mathbf{c}}_j(f,n)$ computed by Equation (4) where δ_1 depends on the system

2.2 Deep neural network spectral model

2.2.1 DNN architecture

The DNNs follow an MLP architecture with an input layer, some hidden layers, and an output layer. 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 depends on how many sources we are estimating. It should be equal to the dimension of the features multiplied by the number of sources. These numbers determine the size of the DNN and the number of its parameters. Thus, they should be carefully considered so that the computational cost can be handled by the available resources. The activation functions of the hidden layers and the output layer are rectified linear unit (ReLU) [15] and linear activation functions, respectively.

2.2.2 DNN input and output

The DNN maps a concatenation of input frames (called "supervector") to one output frame. In this initial proposal, we use magnitude STFT coefficients as input and output features. The supervector consists 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 it causes some information loss, it enables the supervector to represent a longer context in the time domain. This method was also used in [16, 11]. In addition, we do not use the feature values of context frames directly, but the difference between the values of the context frames and the center frame. These values act as complementary features which can be viewed as temporal features similar to delta features in the MFCC domain. The supervector can be expressed as

$$\Xi_j(f,n) = \begin{bmatrix} \widehat{\xi}_j(f,n-2k)^{\frac{1}{2}} - \widehat{\xi}_j(f,n)^{\frac{1}{2}} \\ \vdots \\ \widehat{\xi}_j(f,n)^{\frac{1}{2}} \\ \vdots \\ \widehat{\xi}_j(f,n+2k)^{\frac{1}{2}} - \widehat{\xi}_j(f,n)^{\frac{1}{2}} \end{bmatrix} \tag{5}$$

where k is the length of one-side context in frames.

The dimension of the supervector is then reduced by principal component analysis (PCA) to the dimension of the DNN input. Standardization (zero mean, unit variance) is done element-wise before and after PCA over the training data as in [17]. The output is also standardized element-wise over the training data. The standardization factors and the PCA transformation matrix are then kept for pre-processing and post-processing for any input and output. In addition, a flooring function is employed at the end of post-processing so that the final output is nonnegative.

2.2.3 DNN training

The DNNs are trained by greedy layer-wise supervised training [18] where the hidden layers are added incrementally. In the beginning, a NN with one hidden layer is trained after random initialization of all its parameters (weights and biases). The output layer of this trained NN is then substituted by new hidden and output layers to form a new NN, 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 NN. After random initialization for the parameters of new layers, the new NN is entirely trained. This procedure is done iteratively until the target number of hidden layers is reached (in this case, three hidden layers).

Training is done by backpropagation with adaptive learning rate and minibatch. The learning rate update follows the algorithm proposed in [19], in which the learning rate is driven by the validation error of previous epochs. Besides, the algorithm also allows us to revert to the last best parameters (weights and biases) when several training iterations have failed to get better parameters. The original algorithm uses only the maximum number of epochs as the stopping condition, but we added the maximum number of reversions (called "patience") as an early-stopping condition. Nesterov's Accelerated Gradient (NAG) is then used for updating the weights instead of standard stochastic gradient descent with classical momentum (SGD-CM) as NAG behaves more stably in many situations [20]. The key hyper-parameters of training include the minibatch size, the initial learning rate, the decrement rate of the learning rate, the increment rate of the learning rate, the momentum, the maximum number of iterations, the

maximum number of iterations before reversion, and the patience, which are set to 100, 10^{-3}, 0.7, 1.1, 0.9, 250, 5, and 3, respectively, in our experiments.

The weights for the hidden layers having ReLU activation functions are initialized randomly from a zero-mean Gaussian distribution with standard deviation of $\sqrt{2/n_l}$, where n_l is the fan-in (the number of inputs to the neuron which is equal to the size of the previous layer in our case) [21]. The weights for the output layer are initialized randomly from a zero-mean Gaussian distribution with standard deviation of 0.01. Finally, the biases are initialized to zero.

The loss function used for training is the sum of the mean square error (MSE) and an L2 regularization term

$$L = \frac{1}{2PQ}\sum_{p=1}^{P}\sum_{q=1}^{Q}(\widehat{v}_{pq} - v_{pq})^2 + \frac{\lambda}{2}\sum_{w} w^2 \tag{6}$$

where P is the number of training samples, Q is the dimension of the output layer, $\widehat{v}$ is the estimated output, v is the training target, λ is the regularization parameter, and w are the DNN weights. In our experiments, the value of λ is set to 10^{-5}. There is no regularization applied to the biases.

3 Experiments

3.1 Task and dataset

We evaluated our proposed framework by participating in SiSEC 2015, which is a community-based signal separation evaluation campaign. One of the tasks in SiSEC 2015 is named "MUS" whose objective is to estimate one or more sources from a professionally-produced music recording.

The dataset used for this task contains 100 full-track songs (mixtures) of various music genres by various artists with their corresponding sources. The sources consist of four one- or two-channel tracks containing vocals, bass, drums, and other musical instruments. Both mixture and source tracks are sampled at 44.1 kHz. The dataset is then divided evenly into development and evaluation sets. By using BSS Eval toolbox 3.0[1] [22], the following performance metrics are computed: signal to distortion ratio (SDR), source image to spatial distortion ratio (ISR), signal to interference ratio (SIR), signal to artifacts ratio (SAR). The organizers provided a script for computing these performance metrics. For further details, please refer to the official website[2] and the paper [23].

3.2 Algorithm settings

We aim to estimate all of the four source tracks. For this, we defined three systems based on the proposed framework in Section 2. The input of these three systems is a two-channel mixture signal, while the outputs are four two-channel estimated source signals.

As instructed by the organizers of the challenge, a supervised approach should be trained only on the development set which contains 50 full-track songs. We divided this set into training and validation sets with a ratio of 9 to 1. After dividing each song into 20 chunks, we took two chunks from the center of each song for the validation set and used the rest for the training set. By doing so, the validation set should represent the whole development dataset very well. This was favorable because the training approach we used was strongly driven by the validation error.

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

[2] http://sisec.inria.fr/professionally-produced-music-recordings/

The error determines when to keep the model, what learning rate to be used in next training iteration, and when to stop the training.

The STFT coefficients were extracted by using a Hamming window whose length and overlap are 2048 and 50%, respectively. This means that we used the non-overlapping frames when creating the supervectors. The supervectors were constructed by five frames (two left context, one center, and two right context frames).

3.2.1 System I

Initialization

- $\widehat{v}_j(f,n) \leftarrow$ mean of PSD $|x(f,n)|^2$ over channels

Separation step A regularization coefficient δ_1 of 10^{-20} was used.

Fitting step: spatial updates

(a) $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \leftarrow \widehat{\mathbf{c}}_j(f,n)\widehat{\mathbf{c}}_j(f,n)^H$

(b) $\tilde{v}_j(f,n) \leftarrow$ mean of PSD $|\widehat{\mathbf{c}}_j(f,n)|^2$ over channels

(c) $\widehat{\mathbf{R}}_j(f) \leftarrow \sum_{n=1}^{N} \widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \left(\sum_{n=1}^{N} \tilde{v}_j(f,n) \right)^{-1}$

Fitting step: spectral updates System I used four DNNs, one for each output track. The DNNs were trained by using the original mixture. The input of these DNNs was a single-channel spectrogram, while the output was also a single-channel spectrogram. The DNNs had an input layer size of 2050, three hidden layers with a size of 2050, and an output layer size of 1025. A regularization coefficient δ_2 of 10^{-10} was used.

(a) $\widehat{\xi}_j(f,n) \leftarrow \mathrm{tr}\left(\left[\widehat{\mathbf{R}}_j(f) + \delta_2 \mathbf{I}_I \right]^{-1} \widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \right)$

(b) $\widehat{v}_j(f,n) \leftarrow$ squared output of DNN with input $\widehat{\xi}_j(f,n)^{\frac{1}{2}}$

3.2.2 System II

Initialization Same as System I.

Separation step Same as System I.

Fitting step: spatial updates Same as System I

Fitting step: spectral updates System II used two sets of DNNs. Each set contained four DNNs, one for each output track. The first set was used for the first iteration only and the second set was used for the subsequent iterations. The first set was trained by using the original mixture, while the second set was trained by using the output of the separation step after the fitting step of the first iteration. This was done because the output of the separation step after the first iteration has the characteristic of separated sources, instead of a mixture. By using the second set, we expected that the subsequent updates would be better. The input of these DNNs was a single-channel spectrogram, while the output was also a single-channel spectrogram. The DNNs have an input layer size of 2050, three hidden layers with a size of 2050, and an output layer size of 1025. A regularization coefficient δ_2 of 10^{-10} was used. In the experiments, the first set was the same as the set of DNNs used in System I.

(a) $\widehat{\xi}_j(f,n) \leftarrow \mathrm{tr}\left(\left[\widehat{\mathbf{R}}_j(f) + \delta_2 \mathbf{I}_I\right]^{-1} \widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n)\right)$

(b) $\widehat{v}_j(f,n) \leftarrow$ squared output of DNN_1 (for $l=1$) or DNN_2 (for $l>1$) with input $\widehat{\xi}_j(f,n)^{\frac{1}{2}}$

3.2.3 System III

Initialization System III used a single DNN. The DNN estimated all four output spectrograms simultaneously so as to share the DNN parameters between sources. The DNN was trained by using the original mixture. The input of the DNN was a two-channel spectrogram, while the outputs were four single-channel spectrograms. The DNNs have an input layer size of 2050, three hidden layers with a size of 4100, and an output layer size of 4100. Properly speaking, the standardization of the output for this system was not element-wise but frequency-bin-wise, because the computation of standardization factors for frequency bin f considers all data of frequency bin f from the four target sources. This was done to maintain the ratio between each source in the standardized feature space.

- $\widehat{v}_j(f,n) \leftarrow$ squared outputs of DNN with input $|\mathbf{x}(f,n)|$

Separation step We defined two variants of this system, i.e. Systems IIIa and IIIb. System IIIa used a fixed value of regularization coefficient δ_1 of 10^{-5}. System IIIb used a set of regularization coefficients δ_1, namely $10^{-10}, 10^{-9}, \ldots, 10^{-5}$, and tried to use the smallest possible value for each song for which the matrix was numerically invertible.

Fitting step: spatial updates

(a) $\widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \leftarrow \widehat{\mathbf{c}}_j(f,n)\widehat{\mathbf{c}}_j(f,n)^H$

(b) $\widehat{\mathbf{R}}_j(f) \leftarrow \sum_{n=1}^{N} \widehat{\mathbf{R}}_{\mathbf{c}_j}(f,n) \left(\sum_{n=1}^{N} \widehat{v}_j(f,n)\right)^{-1}$

Fitting step: spectral updates None.

3.3 Results

As mentioned above, the organizers provided a script for computing the performance metrics. Each song is divided into several chunks and the performance metrics are calculated for each chunk. We were not sure how the organizers would analyze these in order to compare the evaluation results submitted by the participant. Therefore, we simply computed the average for each performance metric and present the summary in Tables 1 and 2. The evaluation was done both for development ('Dev') and evaluation ('Eval') datasets. All the metrics are presented in decibels (dB). In Table 1, the evaluation was done for separation of all sources, including vocal ('voc'), bass ('bas'), drum ('dru'), and other ('oth'). In Table 2, the evaluation was done for separation of vocal and accompaniment, where the accompaniment ('acc') is obtained by summing the non-vocal tracks, including bass ('bas'), drum ('dru'), and other ('oth'). The average ('avg') is presented in italics. The best value for each dataset, performance metric, and track type is shown in boldface.

Table 1: Independent performance evaluation for separation of four sources.

	Dev				**Eval**				**Full (Dev+Test)**			
	SDR	ISR	SIR	SAR	SDR	ISR	SIR	SAR	SDR	ISR	SIR	SAR
System I												
voc	2.32	**7.06**	8.70	2.58	0.31	**5.35**	4.68	1.57	1.32	**6.20**	6.69	2.07
bas	7.59	16.59	10.69	11.16	5.29	14.02	**7.46**	10.24	6.44	15.30	9.08	10.70
dru	1.84	5.83	10.02	0.53	0.27	2.35	4.91	-3.29	1.05	4.09	7.47	-1.38
oth	3.17	8.21	6.70	5.02	1.92	6.85	**3.71**	4.75	2.54	7.53	5.20	4.88
avg	*3.73*	*9.42*	*9.03*	*4.82*	*1.95*	***7.14***	*5.19*	*3.31*	*2.84*	*8.28*	*7.11*	*4.07*
System II												
voc	2.89	**7.06**	10.01	2.90	0.47	4.64	4.82	1.01	1.68	5.85	7.41	1.96
bas	**7.99**	17.28	**10.85**	11.53	**5.43**	14.01	7.42	10.43	**6.71**	15.64	**9.14**	10.98
dru	**2.61**	**6.57**	10.85	1.73	**0.63**	**2.78**	5.47	-2.00	**1.62**	**4.67**	8.16	-0.14
oth	3.86	8.64	**7.93**	5.65	**1.97**	7.07	3.35	5.28	2.91	7.86	**5.64**	5.46
avg	*4.34*	***9.88***	*9.91*	*5.45*	*2.13*	*7.12*	*5.26*	*3.68*	*3.23*	***8.50***	*7.59*	*4.57*
System IIIa												
voc	3.85	6.05	12.61	**6.23**	**1.17**	3.69	6.10	**3.69**	2.51	4.87	9.35	**4.96**
bas	7.45	17.96	9.06	**14.19**	5.11	15.63	6.14	**13.04**	6.28	16.79	7.60	**13.61**
dru	2.07	2.96	**13.87**	3.00	0.52	0.90	7.75	-0.88	1.29	1.93	**10.81**	1.06
oth	3.88	8.74	6.60	8.28	1.90	7.04	3.10	7.75	2.89	7.89	4.85	8.01
avg	*4.31*	*8.93*	***10.53***	*7.92*	*2.17*	*6.81*	*5.77*	*5.90*	*3.24*	*7.87*	*8.15*	*6.91*
System IIIb												
voc	**3.88**	6.14	**12.64**	6.21	1.16	3.73	**6.52**	3.65	**2.52**	4.94	**9.58**	4.93
bas	7.45	**18.07**	9.05	14.18	5.12	**15.65**	6.57	13.03	6.28	**16.86**	7.81	13.60
dru	2.24	3.34	13.67	**3.96**	0.62	1.19	**7.92**	**0.61**	1.43	2.27	10.79	**2.29**
oth	**3.95**	**9.11**	6.64	**8.39**	1.91	**7.22**	3.11	**7.84**	**2.93**	**8.17**	4.88	**8.12**
avg	***4.38***	*9.17*	*10.50*	***8.19***	***2.20***	*6.95*	***6.03***	***6.28***	***3.29***	*8.06*	***8.26***	***7.23***

3.4 Discussion

Overall, System IIIb is the best among the systems used in our experiments. However, if we observe the average performance on the evaluation dataset only, the performance differences are not really significant, except for the SAR. Still considering the evaluation dataset only, generally Systems I and II yielded higher SIR and SDR, respectively, while System IIIb yielded higher ISR and SAR. System I and II could not be the best overall for SIR and SDR mainly because System III(a and b) performed much better for the vocal track.

The difference of regularization method between Systems IIIa and IIIb is reflected in ISR metric because the regularization mainly affects spatial filtering. We can observe an improvement of 1 dB on average and almost 4 dB for the accompaniment track when we used smaller regularization values. As an additional information, listening tests comparing the results of these two systems showed that we tend to lose higher frequencies in System IIIa which is not favorable for tracks characterized by higher frequencies, such as drums.

Table 2: Independent performance evaluation for separation of vocals and accompaniment.

	Dev				**Eval**				**Full (Dev+Test)**			
	SDR	ISR	SIR	SAR	SDR	ISR	SIR	SAR	SDR	ISR	SIR	SAR
System I												
voc	2.32	**7.06**	8.70	2.58	0.31	**5.35**	4.68	1.57	1.32	**6.20**	6.69	2.07
acc	13.22	25.78	17.18	17.00	11.11	22.15	**14.76**	15.20	12.16	23.96	**15.97**	16.10
avg	*7.77*	*16.42*	*12.94*	*9.79*	*5.71*	*13.75*	*9.72*	*8.38*	*6.74*	*15.08*	*11.33*	*9.09*
System II												
voc	2.89	**7.06**	10.01	2.90	0.47	4.64	4.82	1.01	1.68	5.85	7.41	1.96
acc	13.94	27.23	**17.41**	17.92	11.27	23.70	14.21	16.28	12.61	25.46	15.81	17.10
avg	*8.41*	*17.14*	*13.71*	*10.41*	*5.87*	*14.17*	*9.51*	*8.64*	*7.14*	*15.66*	*11.61*	*9.53*
System IIIa												
voc	3.85	6.05	12.61	**6.23**	**1.17**	3.69	6.10	**3.69**	2.51	4.87	9.35	**4.96**
acc	14.21	24.03	16.56	20.33	11.54	21.68	13.56	18.69	12.88	22.86	15.06	19.51
avg	*9.03*	*15.04*	*14.58*	*13.28*	*6.35*	*12.68*	*9.83*	*11.19*	*7.69*	*13.86*	*12.21*	*12.23*
System IIIb												
voc	**3.88**	6.14	**12.64**	6.21	1.16	3.73	**6.52**	3.65	**2.52**	4.94	**9.58**	4.93
acc	**14.94**	**29.35**	16.83	**21.46**	**11.96**	**25.26**	13.70	**19.47**	**13.45**	**27.31**	15.26	**20.46**
avg	***9.41***	***17.75***	***14.73***	***13.83***	***6.56***	***14.50***	***10.11***	***11.56***	***7.98***	***16.12***	***12.42***	***12.70***

4 Conclusion

Our experimental results show that the current systems performed reasonably well in the context of music separation. These systems could be used as the baseline for our future experiments. Many aspects should be explored further in the proposed framework, including the features and the hyper-parameters. In addition, the framework should also be tested in the context of speech separation.

References

[1] 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, Jul. 2010.

[2] 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.

[3] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends Signal Process.*, vol. 7, no. 3-4, pp. 197–387, Jun. 2014.

[4] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. 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.

[5] A. Narayanan and D. Wang, "Ideal ratio mask estimation using deep neural networks for robust speech recognition," in *Proc. IEEE Int'l Conf. Acoust. Speech Signal Process. (ICASSP)*, Vancouver, Canada, May 2013, pp. 7092–7096.

[6] 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 and Information Process. (GlobalSIP)*, Dec. 2014, pp. 577–581.

[7] 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'l. Symp. Chinese Spoken Language Process. (ISCSLP)*, Sept 2014, pp. 250–254.

[8] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Deep learning for monaural speech separation," in *Proc. IEEE Int'l Conf. Acoust. Speech Signal Process. (ICASSP)*, Florence, Italy, May 2014, pp. 1562–1566.

[9] ——, "Singing-voice separation from monaural recordings using deep recurrent neural networks," in *Proc. Int'l. Soc. for Music Inf. Retrieval (ISMIR)*, Taipei, Taiwan, Oct. 2014, pp. 477–482.

[10] ——, "Joint optimization of masks and deep recurrent neural networks for monaural source separation," *ArXiv e-prints*, Feb. 2015. [Online]. Available: http://arxiv.org/abs/1502.04149

[11] S. Uhlich, F. Giron, and Y. Mitsufuji, "Deep neural network based instrument extraction from music," in *Proc. IEEE Int'l Conf. Acoust. Speech Signal Process. (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 2135–2139.

[12] 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'l Conf. Acoust. Speech Signal Process. (ICASSP)*, Kyoto, Japan, Mar. 2012, pp. 57–60.

[13] A. Liutkus, D. Fitzgerald, and Z. Rafii, "Scalable audio separation with light kernel additive modelling," in *Proc. IEEE Int'l Conf. Acoust. Speech Signal Process. (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 76–80.

[14] 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.

[15] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier networks," in *Proc. Int'l. Conf. Artificial Intelligence and Statistics (AISTATS)*, vol. 15, Fort Lauderdale, USA, Apr. 2011, pp. 315–323.

[16] 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.

[17] X. Jaureguiberry, E. Vincent, and G. Richard, "Fusion methods for audio source separation," Dec. 2014. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01120685

[18] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Conf. on Neural Information Processing Systems (NIPS)*, Vancouver, Canada, Dec. 2006, pp. 153–160.

[19] S. Duffner and C. Garcia, "An online backpropagation algorithm with validation error-based adaptive learning rate," in *Proc. Int'l. Conf. Artificial Neural Networks (ICANN)*, Porto, Portugal, Sep. 2007, pp. 249–258.

[20] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int'l. Conf. Machine Learning (ICML)*, Atlanta, USA, Jun. 2013, pp. 1139–1147.

[21] 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

[22] E. Vincent, R. Gribonval, and C. Fevotte, "Performance measurement in blind audio source separation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 4, pp. 1462–1469, Jul. 2006.

[23] N. Ono, D. Kitamura, Z. Rafii, N. Ito, and A. Liutkus, "The 2015 signal separation evaluation campaign," in *Proc. Int'l. Conf. Latent Variable Analysis and Signal Separation*, Liberec, Czech Republic, Aug. 2015, to appear.

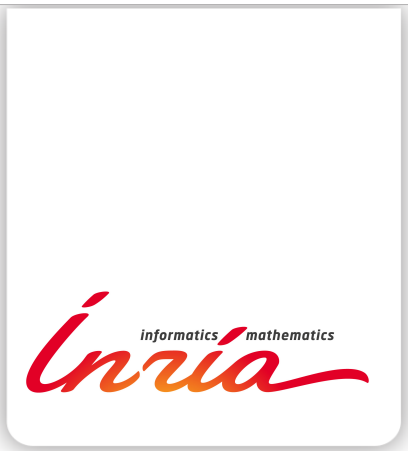
Contents

1 Introduction **3**

2 Proposed framework **4**
2.1 Iterative procedure 4
2.2 Deep neural network spectral model 5
2.2.1 DNN architecture 5
2.2.2 DNN input and output 6
2.2.3 DNN training 6

3 Experiments **7**
3.1 Task and dataset 7
3.2 Algorithm settings 7
3.2.1 System I 8
3.2.2 System II 8
3.2.3 System III 9
3.3 Results 9
3.4 Discussion 10

4 Conclusion **11**



RESEARCH CENTRE
NANCY – GRAND EST

615 rue du Jardin Botanique
CS20101
54603 Villers-lès-Nancy Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399