Fusion Methods for Audio Source Separation

Xabier Jaureguiberry, Emmanuel M. Vincent, Gael Richard

Fusion Methods for Audio Source Separation

Xabier Jaureguiberry, Emmanuel Vincent and Gaël Richard

***Abstract*—A wide variety of audio source separation techniques exists and can already tackle many challenging industrial issues. However, by contrast to other application domains, fusion principles were rarely investigated in audio source separation despite their demonstrated potential in classification tasks. In this paper, we propose a general fusion framework which takes advantage of the diversity of existing separation techniques in order to improve separation quality. Our approaches aim at obtaining a new source estimate by summing the individual estimates given by different separation techniques weighted by a set of fusion coefficients. We investigate three alternative fusion methods which are based on standard non-linear optimization, Bayesian model averaging or deep neural networks. Experiments conducted on both speech enhancement and singing-voice extraction demonstrate that the proposed methods lead to diverse separation performance, yet all outperform traditional model selection. The use of deep neural networks for the estimation of time-varying coefficients notably leads to great quality improvements, up to +3.3 dB in terms of signal-to-distortion ratio (SDR) compared to model selection. As such, our fusion framework is a practical and efficient way to get rid of the need to choose and carefully tune a separation system and it further allows the adaptation of existing techniques to given separation problems and objectives.**

***Index Terms*—Audio Source Separation, Fusion, Aggregation, Ensemble, Deep Neural Networks, Deep Learning, Variational Bayes, Model Averaging, Non-Negative Matrix Factorization, Speech Enhancement, Singing Voice Extraction**

I. Introduction

Blind audio source separation aims at recovering the audio signals, called *sources*, that compose a given mixture. The most challenging situation is at stake when the number of sources is greater than the number of observable channels in the mixture. As such, the problem becomes underdetermined. Numerous approaches have been proposed in the literature [1], [2]. The sources can be modeled based on their sparsity [3], [4], their redundancy [5], [6], their spatial diversity [7], their morphological characteristics [8]–[10] or according to perceptual grouping criteria [11]. Amongst the existing source models, Non-negative Matrix Factorization (NMF) is one of the most popular [12]–[15]. For example, it has achieved great performance in the latest CHiME contest [15], [16] dedicated to speech enhancement. Apart from these model-based methods, source separation can also be handled thanks to more traditional data analysis techniques such as Independent Component Analysis (ICA) [1] or Principal Component Analysis (PCA) [17].

Faced with a given source separation problem to be solved, one will tipically either develop his own approach or choose an existing technique and adapt it to the problem at play. This choice is guided by the type of mixture and sources to be separated and it often leads to a compromise between separation quality and difficulty of implementation. Once a technique has been chosen, the quality of separation also depends on the tuning of its parameters which is often driven by experience. For instance, the order of an NMF model is known to have a great influence on separation quality [18]. Automatic tuning based on model-order selection principles derived from information theory [19] or specific selection criteria [20], [21] might be applied but with limited success on real data [22]. Furthermore, since two distinct separation methods may have complementary strengths and weaknesses, selecting one method rather than another is expected to be suboptimal.

Fusion techniques [23], also named ensemble or aggregation techniques, refer to the combination of several methods in order to better solve a given problem. Transposed to the context of source separation, fusion is opposed to selection as it consists in using several separation methods that differ either by the technique itself or by its tuning and in combining their results into a new solution. Fusion principles have been particularly popular in classification [24] and have led to efficient concepts such as bagging and boosting [25]. Despite being similar to a classification problem [26], audio source separation has barely benefited from fusion principles so far. Recently, the concept of bagging for convolutive blind source separation was introduced in [27] while the authors in [26], [28] proposed to combine time-frequency masks in a way similar to classification.

Following our previous works [22], [26], [29], we here propose a general framework dedicated to fusion in audio source separation which only assumes that all considered separation techniques lead to time-domain estimated signals, thus allowing the combination of heterogeneous separation techniques as well as identical techniques with different parameter settings. In [26], we proposed a preliminary framework in which the fusion rule consisted in summing the estimated time-domain source signals weighted by static time-invariant fusion coefficients. In [29], we proposed to adapt the fusion coefficients to the signal by Bayesian model averaging, thus turning the previous static rule into an adaptive one.

In the following, we further extend these works to a novel adaptive time-varying fusion rule in which the fusion weights are adapted to each frame of the mixture to be separated. As such, this paper establishes a general framework which handles all the aforementioned fusion rules. Although it is presented for single channel signals, it can be easily extended to the multichannel case. In addition, we propose improved learning methods for time-invariant fusion and we introduce a variational Bayesian (VB) approach and a deep neural network (DNN) based approach for time-varying fusion. We evaluate performance using two distinct objective functions, namely

This work was partly supported under the research programme EDiSon3D (ANR-13-CORD-0008-01) funded by ANR, the French State agency for research.

the Mean Square Error (MSE) and the Signal-to-Distortion Ratio (SDR) of the source estimates. We compare all the proposed methods on a speech enhancement task for the fusion of source estimates obtained by NMFs with different numbers of components. We also conduct additional experiments on professionally-produced music signals to get new insight on the combination of heterogeneous separation techniques.

The structure of the rest of the paper is as follows. The general framework is introduced in Section II. The estimation of time-invariant static fusion coefficients is investigated in Section III. A VB algorithm for the estimation of adaptive time-invariant and time-varying fusion coefficients is presented in Section IV for the specific case of NMF. In Section V, we exploit DNNs to determine time-varying adaptive fusion coefficients. The proposed fusion rules are compared in Section VI in the context of speech enhancement using NMF models while the fusion of heterogeneous separation techniques is studied in Section VII. Finally, conclusions are drawn in Section VIII.

II. General framework

A. Single-channel source separation

Throughout this paper, we will assume that the source separation problem we wish to address consists in estimating the J sources $s_j(t)$ that compose an observable linear mixture $x(t)$. The mixing equation can be written in the Short-Time Fourier Transform (STFT) domain as

$$x_{fn} = \sum_{j=1}^{J} s_{j,fn} + \epsilon_{fn} \tag{1}$$

in which f and n respectively denote the frequency bin and the time frame. In the following, we refer to $\mathbf{s}_{fn} = [s_{1,fn} ..., s_{J,fn}]^T$ as the source vector and to ϵ_{fn} as the sensor noise.

B. Fusion of different source estimates

Let us suppose that M distinct models and/or algorithms can be used to estimate each of the J sources. We define a new estimate of each source through a simple weighted sum of the M estimated sources $\tilde{s}_{jm,fn}$ indexed by model m:

$$\forall j, f, n,\ \tilde{s}_{j,fn} = \sum_{m=1}^{M} \alpha_{m,fn}\, \tilde{s}_{jm,fn}, \tag{2}$$

in which $\forall m, f, n, \alpha_{m,fn} \geq 0$ and $\sum_{m=1}^{M} \alpha_{m,fn} = 1$. In the following, we refer to $\boldsymbol{\alpha} = \{\alpha_{m,fn}\}_{m=1..M}$ as the set of *fusion coefficients*, or simply the *fusion vector*.

C. Time-invariant vs. time-varying fusion

Several special cases of the above general fusion rule can be considered. One such special case, called *time-invariant fusion*, is to assume that the fusion coefficients α_m remain independent of the time-frequency bin (f, n). This assumption leads to a simplified expression of (2) which turns out to be equivalent to a weighted sum of the estimated time-domain source signals:

$$\forall j, t,\ \tilde{s}_j(t) = \sum_{m=1}^{M} \alpha_m \tilde{s}_{jm}(t). \tag{3}$$

To go further, we propose in this work to investigate another special case of the time-frequency fusion rule (2) in which the fusion coefficients $\alpha_{m,n}$ depend on time only:

$$\forall j, f, n,\ \tilde{s}_{j,fn} = \sum_{m=1}^{M} \alpha_{m,n}\, \tilde{s}_{jm,fn}. \tag{4}$$

Similarly to above, this fusion rule, called herein *time-varying fusion*, can be rewritten in the time domain. Denoting $\tilde{s}^n_{jm}(t)$ the m^{th} estimation of source j within time frame n, that is the inverse STFT of $\{\tilde{s}_{jm,fn}\}_{f=1..F}$, the resulting framed source signal is expressed as

$$\tilde{s}^n_j(t) = \sum_{m=1}^{M} \alpha_{m,n} \tilde{s}^n_{jm}(t). \tag{5}$$

Contrary to (3), the fusion coefficients now depend on the frame n. The full estimated source $\tilde{s}_j(t)$ is then recovered by summing $\tilde{s}^n_j(t)$ over n in a traditional overlap-add manner.

D. Static vs. adaptive fusion

Two distinct study cases can already been derived from (3) and (5) according to whether the fusion coefficients α_m depend on the observed mixture $x(t)$ or not. In the following, we will refer to *static fusion* when the fusion coefficients do not depend on the mixture (see Section III) and to *adaptive fusion* when the fusion coefficients are estimated according to the mixture to be separated (see Sections IV and V).

E. Oracle estimation

In audio source separation, the quality of separation is often measured by the Signal-to-Distortion Ratio (SDR) expressed in decibels (dB) [30]. For instance, the SDR of the source estimate $\tilde{s}_j(t)$ is given by

$$\text{SDR}\left[\tilde{s}_j\right] = 10\log_{10} \frac{\sum_t \|s_j(t)\|^2}{\sum_t \|s_j(t) - \tilde{s}_j(t)\|^2} \tag{6}$$

where $s_j(t)$ denotes the true source signal.

As the true sources are not available in practice, fusion results which rely on their knowledge will be called *oracle* [31], so as to emphasize that they do not account for achievable results in practical situations but that they give instead an upper bound on the performance that can be expected from a given fusion rule.

As such, for a given mixture $x(t)$, we define the oracle time-invariant fusion coefficients as the coefficients α_m that maximize the SDR of the estimated source $\tilde{s}_j(t)$. They are obtained by solving the following maximization problem under linear equality and inequality constraints:

$$\begin{array}{ll} \underset{\{\alpha_m\}_{m=1..M}}{\text{argmax}} & 10\log_{10} \frac{\sum_t \|s_j(t)\|^2}{\sum_t \left\|s_j(t) - \sum_{m=1}^{M} \alpha_m \tilde{s}_{jm}(t)\right\|^2} \\ \text{subject to} & \left\{ \begin{array}{l} \forall m,\ \alpha_m \geq 0 \\ \sum_{m=1}^{M} \alpha_m = 1 \end{array} \right. . \end{array} \tag{7}$$

This turns out to be equivalent to the minimization of the mean square error (MSE) between the true source $s_j(t)$ and its fused estimate $\sum_{m=1}^{M} \alpha_m \tilde{s}_{jm}(t)$ which can be formulated as a standard Quadratic Programming (QP) [32] problem in matrix form

$$\begin{array}{ll} \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} & c + \boldsymbol{\alpha}^T \widetilde{\mathbf{G}}\, \boldsymbol{\alpha} - 2\, \tilde{\mathbf{d}}^T \boldsymbol{\alpha} \\ \text{subject to} & \left\{ \begin{array}{l} \forall m,\, \alpha_m \geq 0 \\ \sum_{m=1}^{M} \alpha_m = 1 \end{array} \right. \end{array} \tag{8}$$

in which $\boldsymbol{\alpha}$ denotes the vector of fusion coefficients and $\boldsymbol{\alpha}^T$ its transpose. The matrix $\widetilde{\mathbf{G}}$ of size $M \times M$ is the so-called Gram matrix whose elements are the scalar products between the estimated signals, *i.e.*,

$$\forall m_1, m_2,\, \widetilde{G}_{m_1 m_2} = \sum_t \tilde{s}_{jm_1}(t) \tilde{s}_{jm_2}(t). \tag{9}$$

Similarly, the vector $\tilde{\mathbf{d}}$ of length M is composed of the scalar products between the estimated signals and the true source signal and the scalar c is the squared norm of the true source signal:

$$\begin{array}{rl} \forall m, & \tilde{d}_m = \sum_t s_j(t) \tilde{s}_{jm}(t) \\ & c = \sum_t \left\| s_j(t) \right\|^2 . \end{array} \tag{10}$$

Oracle results for time-varying fusion can be similarly computed by replacing the estimated and the true sources by their framed versions $\tilde{s}_{jm}^n(t)$ and $s_j^n(t)$. The components $\widetilde{\mathbf{G}}$, $\tilde{\mathbf{d}}$ and c are thus to be computed on each frame n and not on the whole signal.

III. Static fusion

Assuming that we have defined a subset of M separation systems that are relevant for a given source separation problem, static fusion aims at estimating a unique vector of fusion coefficients for the whole signal, each coefficient being independent of the mixture $x(t)$ to be separated. In this context, the time-invariant rule (3) and the time-varying rule (5) are strictly equivalent. In this section, we propose three distinct methods to estimate static fusion coefficients.

A. Static fusion by mean

The first, simplest method consists in taking the mean of the M estimated signals $\tilde{s}_{jm}(t)$, which is equivalent to setting $\forall m, \alpha_m = 1/M$ in (3). In the following, we will refer to this approach as *static fusion by mean*.

B. Learned static fusion

As an alternative, we propose a learning method to determine the static fusion coefficients from a representative training dataset. To do so, we proposed in [26] to solve a QP problem similar to (8) which was equivalent to minimizing the Mean Square Error (MSE) between the true and estimated sources on the training dataset. Supposing that our training dataset is composed of L mixtures $x^{(l)}(t)$ together with their true sources $s_j^{(l)}(t)$, we thus wish to solve the following minimization problem

$$\begin{array}{ll} \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} & \sum_l c_l + \boldsymbol{\alpha}^T \left(\sum_l \widetilde{\mathbf{G}}_l\right) \boldsymbol{\alpha} - 2 \left(\sum_l \tilde{\mathbf{d}}_l^T\right) \boldsymbol{\alpha} \\ \text{subject to} & \left\{ \begin{array}{l} \forall m,\, \alpha_m \geq 0 \\ \sum_{m=1}^{M} \alpha_m = 1 \end{array} \right. , \end{array} \tag{11}$$

in which $\widetilde{\mathbf{G}}_l$, $\tilde{\mathbf{d}}_l$ and c_l are defined as in (9) and (10) but for each example l. In the following, the coefficients thus obtained will be referred to as *MSE-based static fusion* coefficients.

Here, to go further, we propose to optimize the coefficients α_m in order to maximize the average SDR on the training dataset. This turns out to be equivalent to the following minimization problem :

$$\begin{array}{ll} \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} & \sum_l 10 \log_{10} \left(c_l + \boldsymbol{\alpha}^T \widetilde{\mathbf{G}}_l\, \boldsymbol{\alpha} - 2\, \tilde{\mathbf{d}}_l^T\, \boldsymbol{\alpha} \right) \\ \text{subject to} & \left\{ \begin{array}{l} \forall m,\, \alpha_m \geq 0 \\ \sum_{m=1}^{M} \alpha_m = 1 \end{array} \right. . \end{array} \tag{12}$$

In the following, the fusion coefficients thus obtained will be called *SDR-based static fusion* coefficients.

Both MSE-based and SDR-based static coefficients α_m can be used to separate any other mixture $x(t)$ which is not present in the training dataset. Note that alternative choices for the objective function could also be considered such as the Signal-to-Interference Ratio (SIR), the Signal-to-Artifacts Ratio (SAR), a combination of these measures [30], or any other objective function relevant to a given source separation problem.

IV. Adaptive fusion using variational Bayesian averaging

The quality of separation can be improved by adapting the fusion coefficients to the mixture to be separated. In that context, we propose to exploit the principle of Bayesian model averaging [33] and apply it to NMF in order to derive adaptive fusion coefficients.

A. Non-negative matrix factorization

1) Maximum-likelihood formulation: Historically, NMF has been proposed as an optimization problem aiming at decomposing a non-negative observation matrix into the product of two other non-negative matrices [34]. When using NMF for audio source separation [35], it is usually assumed that the Power Spectral Density (PSD) $v_{j,fn}$ of each source at time n and frequency f can be modeled as the result of NMF

$$v_{j,fn} = \sum_{k=1}^{K_j} w_{j,fk} h_{j,kn}, \tag{13}$$

in which the coefficients $\{w_{j,fk}\}_{f=1..F}^{k=1..K_j}$ form the *dictionary* of spectral templates characterizing the spectral content of the j^{th} source and $\{h_{j,kn}\}_{k=1..K_j}^{n=1..N}$ are the so-called *activation* coefficients which indicate the amplitude of activation of each spectral template across time. K_j is the *number of components*, *i.e.*, the number of spectral templates chosen to model source j.

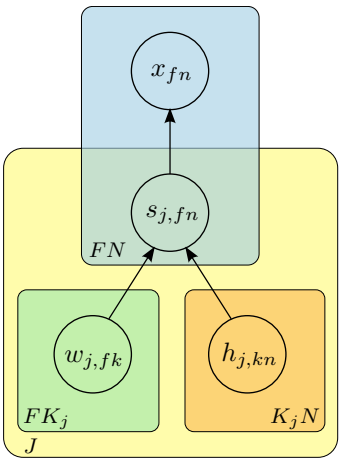

Fig. 1. Graphical model for Bayesian NMF

Parameter estimation is performed by iterative multiplicative update rules which minimize a certain divergence d between the mixture PSD $|x_{fn}|^2$ on the one hand, and the sum of the J source PSDs $v_{j,fn}$ and the noise PSD σ^2 on the other hand:

$$\underset{w,h}{\operatorname{argmin}} \sum_{fn} d\left(|x_{fn}|^2 \middle| \sum_{j=1}^{J} v_{j,fn} + \sigma^2\right). \tag{14}$$

In the following, we consider the Itakura-Saito (IS) divergence:

$$d_{IS}(x|y) = \frac{x}{y} - \log\frac{x}{y} - 1. \tag{15}$$

Once the model parameters have been estimated, the source STFT coefficients are estimated as:

$$\tilde{s}_{j,fn} = \frac{v_{j,fn}}{\sum_{j'=1}^{J} v_{j',fn} + \sigma^2}\, x_{fn}. \tag{16}$$

This original formulation of NMF can also be seen as a maximum likelihood (ML) problem [21]. In that case, each source $s_{j,fn}$ is assumed to follow a circularly-symmetric complex normal distribution

$$s_{j,fn} \sim \mathcal{N}(0, v_{j,fn}) \tag{17}$$

whose variance $v_{j,fn}$ is the result of NMF as defined in (13). ϵ_{fn} is supposed to follow a Gaussian distribution of zero mean and of variance σ^2 so that $\epsilon_{fn} \sim \mathcal{N}(0, \sigma^2)$. The source estimate given in (16) is thus justified as being the Minimum Mean Square Error (MMSE) estimate of source j [36].

The results of separations obtained with NMF highly depend on the number of components K_j that is chosen for each source [18], [26]. Assuming that the separation process has been conducted for M different numbers of components K_{jm} leading to M estimates $\tilde{s}_{jm}(t)$ of each source j, a new source estimate can be obtained using (2).

2) Variational Bayesian formulation: To go further, we propose to consider the generative model of NMF depicted in Fig. 1, which allows full Bayesian treatment. Both the dictionary and the activation coefficients are seen as random variables and assumed to follow Gamma priors [37]:

$$w_{j,fk} \sim \Gamma(a, a) \;\text{ and }\; h_{j,kn} \sim \Gamma(b, b). \tag{18}$$

The goal of Bayesian inference is to estimate the posterior probability over all model parameters. As this is generally intractable, some approximation is required. VB is a practical inference algorithm which has been successfully applied to this generative model [37], [38] in particular.

In this context, the posterior probability of the source vector $\mathbf{s}_{fn}$ is identified as a multivariate complex Gaussian distribution denoted as $q(\mathbf{s}_{fn}) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{s},fn}, \boldsymbol{\Sigma}_{\mathbf{s},fn})$. The STFT coefficients of the sources are then given by the expectation $\boldsymbol{\mu}_{\mathbf{s},fn}$ of this posterior distribution, which can be expressed in a similar form as (16) with

$$\tilde{s}_{j,fn} = \mu_{s_{j,fn}} = \frac{C_{j,fn}}{\sum_{j'=1}^{J} C_{j',fn} + \sigma^2}\, x_{fn}. \tag{19}$$

The term $C_{j,fn}$ depends on the NMF parameters of source j through the expectation

$$C_{j,fn} = \sum_{k=1}^{K_j} \mathbb{E}\left[\frac{1}{w_{j,fk} h_{j,kn}}\right]^{-1} \tag{20}$$

in which $\mathbb{E}$ denotes the expectation over the variational posteriors $q(w_{j,fk})$ and $q(h_{j,kn})$ which turn out to be generalized inverse Gaussian (GIG) distributions. For the detailed expressions of $\boldsymbol{\Sigma}_{\mathbf{s},fn}$, $q(w_{j,fk})$, $q(h_{j,kn})$ and $C_{j,fn}$, see [29].

B. Adaptive fusion as variational Bayesian averaging

Practically, VB proposes to approximate the marginal likelihood of a model by the so-called *free energy*, which can then be used to select the most likely model [39], [40] amongst a set of possible models. VB has been notably applied to NMF in order to infer the best number of components [37], [41]. To go further, we propose here to use the free energy to achieve fusion instead of simple selection.

Indeed, the above VB formulation of NMF gives a straightforward interpretation of the fusion rule expressed in (5). We here assume that the separation process has been independently conducted for M different NMF models, each model being defined by the set of numbers of components chosen to model the J sources and denoted as $\mathbf{K}_m = \{K_{1m}, ..., K_{Jm}\}$. K_{jm} refers to the number of components chosen in model m for source j. Bayesian model averaging [33] proposes to average the M source posterior distributions $q_m(\mathbf{s}_{fn})$ as

$$q(\mathbf{s}_{fn}) = \sum_{m=1}^{M} p(\mathbf{K}_m|x)\, q_m(\mathbf{s}_{fn}), \tag{21}$$

in which $p(\mathbf{K}_m|x)$ is the posterior probability of model m. Taking the expectation of (21) leads to the following expression of the fused source

$$\tilde{s}_{j,fn} = \sum_{m=1}^{M} p(\mathbf{K}_m|x)\, \tilde{s}_{jm,fn}. \tag{22}$$

Thanks to Bayes rule, the posterior over $\mathbf{K}_m$ can be expressed as the product of its prior probability π_m and its likelihood $p(x|\mathbf{K}_m)$, up to a normalization constant. As this

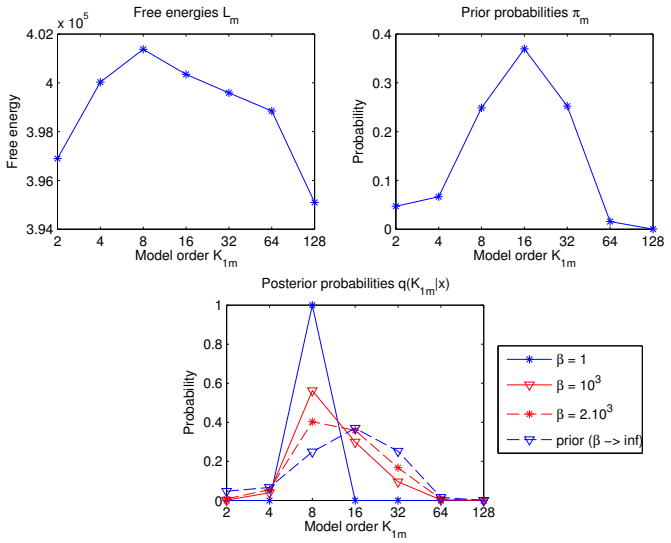

Fig. 2. Shape of the order posterior as a function of the shape parameter β, for one example of the CHiME corpus

likelihood is intractable in practice, we replace it with the free energy given by VB inference and denoted as $\mathcal{L}_m$:

$$\begin{aligned}\mathcal{L}_m &= \sum_{fn} \mathbb{E}\left[\log p(x|\mathbf{s}_{fn})\right] \\ &+ \sum_{j,fn} \left(\mathbb{E}\left[\log p(s_{j,fn}|w_j,h_j)\right] - \mathbb{E}\left[\log q(s_{j,fn})\right]\right) \\ &+ \sum_{j,fk} \left(\mathbb{E}\left[\log p(w_{j,fk})\right] - \mathbb{E}\left[\log q(w_{j,fk})\right]\right) \\ &+ \sum_{j,kn} \left(\mathbb{E}\left[\log p(h_{j,kn})\right] - \mathbb{E}\left[\log q(h_{j,kn})\right]\right).\end{aligned} \tag{23}$$

Here, the operator $\mathbb{E}$ denotes the expectation over the variational distributions of each parameter, *i.e.*, $q(s_{j,fn})$, $q(w_{j,fk})$ and $q(h_{j,kn})$. For details on their computation, refer to [38]. Once the free energy has been computed for each model m, the fusion coefficients introduced in (2) are identified as the posterior probability of $\mathbf{K}_m$:

$$\alpha_m \propto \pi_m \exp^{\mathcal{L}_m}. \tag{24}$$

Due to the approximate inference strategy and because the data do not strictly adhere to any of the M models, we have observed in preliminary experiments that the fusion coefficients estimated according to (24) practically result in a selection instead of a fusion, *i.e.*, one fusion coefficient is equal to 1 and the others are equal to 0. Moreover, it turns out that the selected model is not always the one which gives the best separation quality. As a consequence, we propose to introduce a parameter $\beta \geq 1$ that aims at controlling the shape of the posterior of $\mathbf{K}_m$ by penalizing its entropy [42]. The fusion coefficients are now given by

$$\alpha_m \propto \pi_m \exp^{\mathcal{L}_m/\beta}. \tag{25}$$

As shown in Fig. 2, when $\beta = 1$, the proposed fusion turns out to select the model with the highest free energy $\mathcal{L}_m$. On the contrary, when β tends towards infinity, the posterior distribution tends towards the prior probabilities π_m. A value of β between these two extreme values allows us to achieve a suitable compromise between the prior and the likelihood of each model.

C. Learning the priors and the shape parameter

The prior probabilities π_m as well as the shape parameter β must be learned. Similarly to the learned static fusion approaches introduced in Section III-B, we propose to learn them by optimizing the MSE or the SDR on a representative training dataset. This time, the optimization problem can be written in the MSE case as

$$\begin{aligned}\underset{\boldsymbol{\pi},\beta}{\text{argmin}} \quad & \sum_l \Big(c_l - 2\,\tilde{\mathbf{d}}_l^T \left(\boldsymbol{\pi} \circ \mathrm{e}^{\boldsymbol{\mathcal{L}}_l/\beta}\right) \\ & + \left(\boldsymbol{\pi} \circ \mathrm{e}^{\boldsymbol{\mathcal{L}}_l/\beta}\right)^T \widetilde{\mathbf{G}}_l \left(\boldsymbol{\pi} \circ \mathrm{e}^{\boldsymbol{\mathcal{L}}_l/\beta}\right) \Big) \\ \text{subject to} \quad & \begin{cases} \forall l, m,\ \pi_m\, \mathrm{e}^{\mathcal{L}_m^{(l)}/\beta} \geq 0 \\ \forall l,\ \sum_m \pi_m\, \mathrm{e}^{\mathcal{L}_m^{(l)}/\beta} = 1 \end{cases},\end{aligned} \tag{26}$$

or in the SDR case as

$$\begin{aligned}\underset{\boldsymbol{\pi},\beta}{\text{argmin}} \quad & \sum_l 10\log_{10} \Big(c_l - 2\,\tilde{\mathbf{d}}_l^T \left(\boldsymbol{\pi} \circ \mathrm{e}^{\boldsymbol{\mathcal{L}}_l/\beta}\right) \\ & + \left(\boldsymbol{\pi} \circ \mathrm{e}^{\boldsymbol{\mathcal{L}}_l/\beta}\right)^T \widetilde{\mathbf{G}}_l \left(\boldsymbol{\pi} \circ \mathrm{e}^{\boldsymbol{\mathcal{L}}_l/\beta}\right) \Big) \\ \text{subject to} \quad & \begin{cases} \forall l, m,\ \pi_m\, \mathrm{e}^{\mathcal{L}_m^{(l)}/\beta} \geq 0 \\ \forall l,\ \sum_m \pi_m\, \mathrm{e}^{\mathcal{L}_m^{(l)}/\beta} = 1 \end{cases}.\end{aligned} \tag{27}$$

In both cases, $\boldsymbol{\pi}$ denotes the vector of priors π_m, $\boldsymbol{\mathcal{L}}_l$ the vector composed of the M free energies $\mathcal{L}_m^{(l)}$ for example l and $\circ$ the Hadamard (or element-wise) product operator. Due to the introduction of the shape parameter β, solving (26) and (27) on a given training database is much more complex than for static fusion as the related optimization problems become non-linear under non-linear constraints. However, trust region algorithms [32], [43] may reach satisfactory local minima.

D. Extension to time-varying fusion

The approach of Section IV-B can be extended to the estimation of time-varying fusion coefficients. To that aim, the free energy for frame n is defined as

$$\begin{aligned}\mathcal{L}_{m,n} &= \sum_f \mathbb{E}\left[\log p(x|\mathbf{s}_{fn})\right] \\ &+ \sum_{j,f} \left(\mathbb{E}\left[\log p(s_{j,fn}|w_j,h_j)\right] - \mathbb{E}\left[\log q(s_{j,fn})\right]\right) \\ &+ \sum_{j,fk} \left(\mathbb{E}\left[\log p(w_{j,fk})\right] - \mathbb{E}\left[\log q(w_{j,fk})\right]\right) \\ &+ \sum_{j,k} \left(\mathbb{E}\left[\log p(h_{j,kn})\right] - \mathbb{E}\left[\log q(h_{j,kn})\right]\right).\end{aligned} \tag{28}$$

Time-varying fusion coefficients are then estimated for each frame n as

$$\alpha_{m,n} \propto \pi_m \exp^{\mathcal{L}_{m,n}/\beta} \tag{29}$$

in which the priors π_m and the shape parameter β can be learned on a representative dataset by replacing the summation on l in (26) and (27) by a summation on both l and n. Moreover, $\widetilde{\mathbf{G}}_l$, $\tilde{\mathbf{d}}_l$ and c_l must be replaced by their framed counterparts $\widetilde{\mathbf{G}}_{l,n}$, $\tilde{\mathbf{d}}_{l,n}$ and $c_{l,n}$.

V. Adaptive fusion using neural networks

The adaptive fusion scheme presented in Section IV requires a Bayesian treatment of source separation which may not be available for other models than NMF. Moreover, as it will be demonstrated in Section VI, adaptive time-varying averaging as exposed in Section IV-B leads to unsatisfactory results compared to oracle fusion. As a consequence, we propose in this section to resort to DNNs in order to determine time-varying fusion coefficients and get closer to oracle performance.

A. Problem formulation

Given a representative training dataset composed of several mixtures with the corresponding true and estimated sources, we wish to estimate the fusion coefficients from the knowledge of the mixture and the estimated sources only. Traditionally, such an estimation is conducted in two steps. The first step consists in computing some features of the inputs, namely the mixture signal $x(t)$ and the M estimated signals $\tilde{s}_{jm}(t)$. The second step aims at mapping these features to the desired output, here the oracle vector of fusion coefficients $\{\alpha_{m,n}\}$.

For the feature extraction step, the set of potential features is extremely large and the selection of an appropriate subset varies with respect to the type of mixture and sources at play. For instance, we can name Mel-Frequency Cepstrum Coefficients (MFCCs), Linear Prediction Coefficients (LPC), chroma and so on [44].

For the mapping step, Gaussian Mixture Models (GMMs) have been widely used [45]. However, DNNs have now outperformed GMMs in many fields [46]. The main advantage of DNNs is their ability to automatically learn features from the input and map them to the desired output. The recent introduction of *rectified linear units* [47] replacing traditional hyperbolic tangent and logistic activations has brought even more advantages such as faster and more accurate convergence. Neural networks thus seem to be a good solution to handle both feature and mapping steps jointly.

B. Network architecture

Some considerations about the architecture of the neural network required for such a task can already be discussed, without any experimental context. Concerning the dimensionality of the problem, we can expect to have a small output layer of size M (*i.e.*, one neuron per fusion coefficient) but a much larger input layer. In the following, we will consider that the input is composed of short-term power spectra of the available signals.

More precisely, in our experiments detailed in Sections VI and VII, we propose that the input relative to frame n is defined as follows:

$$\begin{bmatrix} |\mathbf{x}_{n-C}|^2 & \cdots & |\mathbf{x}_n|^2 & \cdots & |\mathbf{x}_{n+C}|^2 \\ |\tilde{\mathbf{s}}_{j1,n-C}|^2 & \cdots & |\tilde{\mathbf{s}}_{j1,n}|^2 & \cdots & |\tilde{\mathbf{s}}_{j1,n+C}|^2 \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ |\tilde{\mathbf{s}}_{jm,n-C}|^2 & \cdots & |\tilde{\mathbf{s}}_{jm,n}|^2 & \cdots & |\tilde{\mathbf{s}}_{jm,n+C}|^2 \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ |\tilde{\mathbf{s}}_{jM,n-C}|^2 & \cdots & |\tilde{\mathbf{s}}_{jM,n}|^2 & \cdots & |\tilde{\mathbf{s}}_{jM,n+C}|^2 \end{bmatrix} \tag{30}$$

in which the first line refers to the mixture with $\mathbf{x}_n = [x_{1n}, ..., x_{fn}, ..., x_{Fn}]$ and the next M lines refer to the M estimated sources with $\tilde{\mathbf{s}}_{jm,n} = [\tilde{s}_{jm,1n}, ..., \tilde{s}_{jm,fn}, ..., \tilde{s}_{jm,Fn}]$. Each line is composed of the power spectra of the current frame, the C preceding frames and the C following frames. We hence take advantage of the context of the central frame n. Each frame being a vector of F frequency bins, the final input which results from the flattening of matrix (30) into one dimension is a vector of length $F(2C+1)(M+1)$.

In order to decrease the size of the neural network, a reduction of the input dimensionality might thus be needed. For this purpose, Singular Value Decomposition (SVD) or Principal Component Analysis (PCA) are both commonly used to reduce high-dimensional data while retaining relevant components of the input [48]. For our experiments, we chose to use PCA so as to keep a certain amount of data variance. Moreover, the training data have been standardized, *i.e.*, centered and normalized to unit variance, before and after PCA.

The output layer, which corresponds to the M estimated time-varying fusion coefficients $\tilde{\alpha}_{m,n}$, is composed of M neurons, one for each coefficient. Each output neuron uses a *softmax* function as activation in order to ensure that $\forall n, \sum_m \tilde{\alpha}_{m,n} = 1$. Other layers are made of Rectified Linear Units (ReLUs).

C. Training cost functions

A neural network is usually trained by gradient descent together with backpropagation of errors with respect to a given cost function to be minimized [49]. The cost function is often defined as a function of the estimated output, here the estimated coefficients $\{\tilde{\alpha}_{m,n}\}$ for time frame n, and of the desired output, here the oracle coefficients $\{\alpha_{m,n}\}$ as defined in Section II-E. A common choice is to minimize the mean square error. For a given frame n, the MSE is defined as

$$\varphi_n^{\text{OMSE}} = \sum_{m=1}^{M} \left(\alpha_{m,n} - \tilde{\alpha}_{m,n}\right)^2. \tag{31}$$

In the following, we will refer to this cost as the Oracle MSE (OMSE) to distinguish it from the source MSE cost already introduced in (8), (11) and (26). Another common choice is to use the Cross-Entropy (CE) generalized to non-binary multiclass problems which is defined in our context as

$$\varphi_n^{\text{CE}} = -\sum_{m=1}^{M} \alpha_{m,n} \log \frac{\tilde{\alpha}_{m,n}}{\alpha_{m,n}}. \tag{32}$$

Both these cost functions require the oracle fusion coefficients to be known in order to estimate the errors.

Hereafter, we propose two other cost functions that do not require the knowledge of oracle fusion coefficients. Following the MSE optimization formulations of (11) and (26), the cost function for training can be defined as the MSE between the n^{th} frame of the true source $s_j^n(t)$ and of its estimate $\tilde{s}_j^n(t)$ fused with its corresponding estimated fusion coefficients, *i.e.*, the outputs of the network $\tilde{\alpha}_{m,n}$. For frame n, the cost function is defined as

$$\varphi_n^{\text{SMSE}} = c_n + \tilde{\boldsymbol{\alpha}}_n^T \, \widetilde{\mathbf{G}}_n \, \tilde{\boldsymbol{\alpha}}_n - 2\, \tilde{\mathbf{d}}_n^T \, \tilde{\boldsymbol{\alpha}}_n \tag{33}$$

where c_n, $\tilde{\mathbf{d}}_n$ and $\widetilde{\mathbf{G}}_n$ are defined in (9) and (10). Following (12) and (27), the cost function can also be defined as the SDR of the n^{th} frame of source estimate $\tilde{s}_j^n(t)$ fused with estimated fusion coefficients $\tilde{\alpha}_{m,n}$. The cost function thus becomes :

$$\varphi_n^{\text{SDR}} = 10 \log_{10} \left(c_n + \tilde{\boldsymbol{\alpha}}_n^T \, \widetilde{\mathbf{G}}_n \, \tilde{\boldsymbol{\alpha}}_n - 2\, \tilde{\mathbf{d}}_n^T \, \tilde{\boldsymbol{\alpha}}_n \right). \tag{34}$$

Note that in order to make the training as efficient as with the two other cost functions, c_n, $\widetilde{\mathbf{G}}_n$ and $\tilde{\mathbf{d}}_n$ can be precomputed for all frames n of the training dataset.

Finally, we shall remark that neural networks are usually trained by iterative *mini-batch* gradient descent so that each iteration aims at minimizing the mean cost function over a small sample $\mathcal{B}$ of the training dataset, called a *mini-batch*. The total cost function to be minimized at each iteration is thus

$$\varphi = \frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \varphi_n \tag{35}$$

in which $|\mathcal{B}|$ is the mini-batch size. As a consequence, in an on-line setting, *i.e.*, when $|\mathcal{B}| = 1$, the source MSE cost (33) and the source SDR cost (34) are strictly equivalent. In the following however, the mini-batch size has been fixed to $|\mathcal{B}| = 50$.

D. Other settings

At each iteration, the frames which compose a mini-batch are randomly picked from the training dataset. At the end of each epoch (*i.e.*, when all training frames have been presented exactly once), the performance of the current network is evaluated on a validation dataset. According to the average validation score, the learning rate is adapted for next epoch and *early stopping* may be performed following the method proposed in [50]. The training is stopped either when the validation score does not evolve anymore or when a predefined maximum number of epochs is attained. The performance of the final network is then evaluated on a test dataset. For our experiments, the neural networks have been implemented thanks to the Python library *Theano* [51], [52] which enables to compile mathematical expressions for optimized computation either on a Central Processing Unit (CPU) or on a Graphics Processing Unit (GPU).

VI. Experimental evaluation on a single-channel speech enhancement task

In this section, we propose to evaluate and compare all the fusion techniques described in Sections III, IV and V. As the probabilistic fusion framework of Section IV has been introduced in the context of NMF models, we restrain our study to a speech enhancement scenario in which the sources will be exclusively modelled by NMF.

A. CHiME corpus

For this experiment, we rely on the CHiME corpus [53]. The signals are composed of speech from 34 distinct speakers overlapped with noise signals recorded in a real domestic environment.

We divided the data into four disjoint datasets :

- a *clean training dataset* which features 500 utterances in clean conditions (*i.e.*, reverberated but without background noise) for each speaker,
- a *training dataset* composed of 600 utterances per speaker, each mixed with background noise at six different SNRs,
- a *validation dataset* composed of 34 utterances (one for each speaker), each mixed with background noise at six different SNRs,
- and a *test dataset* also composed of 34 utterances, each mixed with background noise at six different SNRs.

The background noise has been randomly chosen in order to reach the six different SNRs, namely $\{-6$ dB, -3 dB, 0 dB, 3 dB, 6 dB, 9 dB$\}$. The clean training dataset has been used to learn speaker-dependent dictionaries W_{1m} for each number of components $K_{1m} = 2^m$. The training dataset has been used to train the neural networks (see Section IV), to learn static fusion coefficients (see Section III) and to learn the priors π_m and the shape parameter β (see Section IV). The validation set has been used in the deep learning approach in order to perform *early stopping* and learning rate adaptation according to [50]. Finally, the test set aims at evaluating and comparing the proposed techniques. Note that the test set has never been used either to learn the fusion coefficients or to optimize the architecture of neural networks.

B. Algorithm settings

Single-channel speech enhancement aims at cleaning up a speech signal s_1 from a background noise s_2. Both the speech and the noise signals are modelled by NMF either by using the ML formulation of Section IV-A1 or the VB formulation of Section IV-A2.

In this context, the signal of interest is the speech signal s_1. As such, we propose in the following to keep the number of components of the background model to a constant value $K_2 = 32$ and to vary the number of components of the speech model K_1 only. We consider $M = 7$ possible numbers of components $K_{1m} = 2^m$ with $m = 1..M$. The time-frequency representation used in this experiment is the Quadratic Equivalent Rectangular Bandwidth (QERB) transform [54] with half-overlapping windows of 1024 samples and $F = 350$ frequency bins, in place of the traditional STFT.

C. Evaluation measure

As aforementioned, the SDR is the most used measure in audio source separation. Methods will thus be compared with respect to the SDR of the estimated speech signals. The performance on a given example is measured as the global SDR on the whole example signal as defined in (6). It is then averaged across all examples of the test (resp. validation) set.

D. ML vs. VB inference for NMF

As a baseline, the SDRs of the speech signals estimated with both ML-NMF and VB-NMF and are drawn on Fig. 3, as a function of the M number of components. The highest performance is obtained using VB-NMF with 16 components (5.85 dB in average). The graph also suggests that VB-NMF better accommodates with low numbers of components whereas ML-NMF performs better for higher numbers of components. However, as it has more parameters to estimate, note that VB-NMF is less computationally efficient than ML-NMF (14.5 seconds vs. 12.8 seconds in average per excerpt).

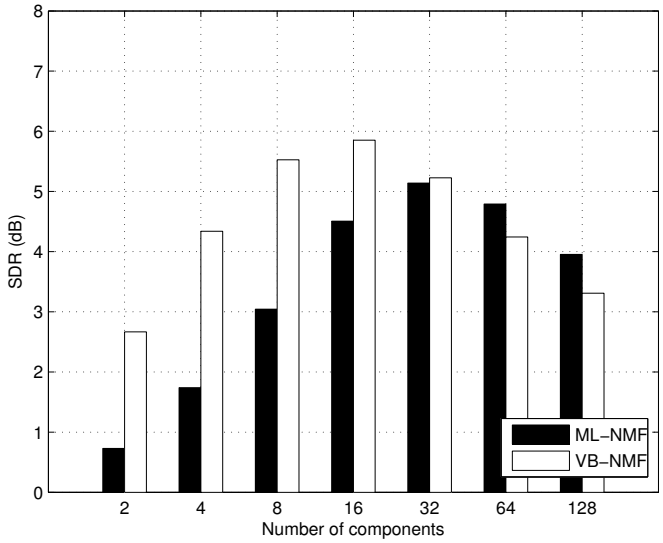

Fig. 3. Separation performance of ML-NMF and VB-NMF as a function of the number of components

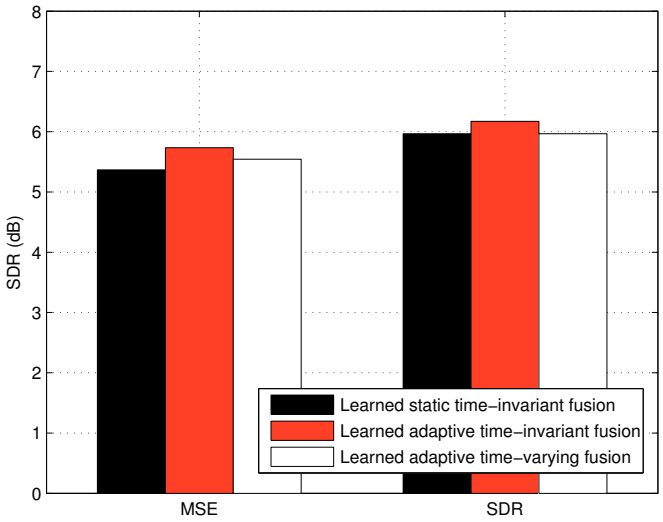

Fig. 5. Separation performance of adaptive time-invariant and time-varying fusions using variational Bayesian averaging, for both MSE and SDR optimization

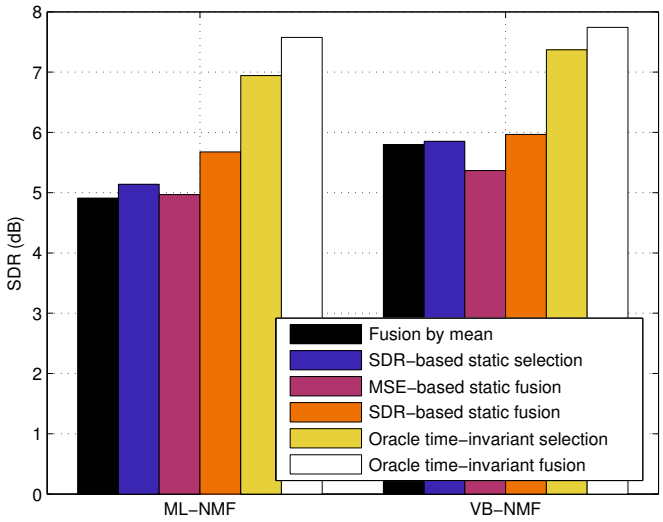

Fig. 4. Separation performance of static time-invariant fusion compared to oracle time-invariant fusion, for both ML-NMF and VB-NMF

E. Static fusion

Hereafter, the results of static fusion methods introduced in Section III are presented and compared to traditional selection results.

1) Selection vs. fusion: Fig. 4 depicts the results given by *oracle time-invariant fusion*. These are to be compared with *oracle time-invariant selection* which is also shown on Fig. 4 and which consists in selecting for each example the best performing model instead of combining the M models as in fusion. In both ML and VB cases, oracle time-invariant fusion outperforms oracle time-invariant selection (+0.54 dB for ML-NMF and +0.31 dB for VB-NMF), thus demonstrating the interest of fusion over simple selection.

2) Learned fusion and fusion by mean: This is confirmed by the results of practical learning-based techniques also depicted in Fig. 4.

Usually, only one model is used for source separation and this model is chosen by an experimental assessment. *SDR-based static selection* simulates such a choice by retaining the individual model that best performs on the training dataset in terms of SDR. *MSE-based static fusion* is obtained by solving the minimization problem (11) whereas *SDR-based static fusion* is obtained by solving (12). We notice that optimizing the MSE on the training set fails to determine static fusion coefficients that improve separation quality in terms of SDR compared to standard SDR-based static selection. However, optimizing the average SDR on the training set allows us to improve SDR-based static selection by 0.54 dB and 0.74 dB with ML-NMF and VB-NMF respectively.

Finally, it is worth noting that simple fusion by mean gives interesting performance. Indeed, contrary to the others, this fusion approach can be applied in situations where a training dataset is not available. Despite this, it outperforms SDR-based static selection by 0.57 dB in the VB-NMF scheme.

F. Adaptive fusion using variational Bayesian averaging

The performance of static fusion methods are satisfactory compared to traditional selection techniques but the adaptation of the fusion coefficients to the mixture at play could allow us to get closer to oracle time-invariant performance. Fig. 5 depicts the results obtained by adaptive time-invariant and time-varying fusions using variational Bayesian averaging, as exposed in Section IV, for both MSE (26) and SDR (27) optimizations. For comparison, static time-invariant fusion is also depicted. Once again, SDR optimization leads to better results than MSE optimization. *SDR-based time-invariant adaptive fusion* allows an improvement of 0.2 dB SDR with respect to its static counterpart. However, learned adaptive time-varying fusion which allows to adapt the fusion at the time frame level is not as efficient as we expect. It is even outperformed by adaptive time-invariant fusion, which is probably due to better minima found for (26) and (27) in the time-invariant

case than in the time-varying case. Furthermore, the overall mixed performance of these methods based on variational Bayesian averaging can be explained on the one hand by the approximations needed to estimate the free energy (23) of each model, and on the other hand by the complexity of the functions (26) and (27) to be optimized which does not guarantee a good local minimum.

G. Adaptive time-varying fusion using DNNs

Adaptive time-varying fusion based on variational Bayesian averaging has failed to improve separation performance compared to its time-invariant counterpart. Yet, oracle time-varying fusion results show that it has a great potential as the average SDR that could be reached on our test set equals 10.35 dB with ML-NMF and 9.99 dB with VB-NMF. As we will show in this subsection, the adaptive time-varying fusion framework based on DNNs, exposed in Section V, allows us to outperform adaptive fusion based on variational Bayesian averaging.

The search for the best DNN architecture for our problem has been conducted through the testing of several DNN architectures, from single-layer networks to deeper networks. Notably, the number of hidden layers has been varied from one to four layers and the number of units per layer has been defined as multiples of the output layer size M. We have tested 11 layer sizes, namely $\{7, 14, 28, 56, 112, 224, 448, 896, 1792, 3584, 7168\}$. We here used the ML-NMF formulation as the resulting oracle time-varying performance is higher than for VB-NMF. These architectures will be reviewed in the next subsections. To start with, let us introduce the architecture with a unique hidden layer that performs best.

1) Best single-layer architecture: The best architecture has been selected according to its performance on the validation dataset. The input was composed as exposed in Sections V-B and VI-B. According to (30), we chose a context of size $C = 2$, which means that we considered the two frames that precede and the two frames that follow the central frame n for each input signal. We computed a PCA on the whole input in order to keep 85 % of the variance, which allows us to reduce the input dimensionality from 14000 to 154 input units. Remember that the data have also been standardized, i.e., centered and normalized to unit variance, before and after PCA. The best results have been obtained using the SDR cost as defined in Sec. V-C.

The results for neural networks with a unique hidden layer have been plotted in Fig. 6 with respect to the number of hidden units for both the validation and the test datasets. The best performing network on the validation set has 896 hidden units and leads to a SDR of 8.46 dB on the test set. Adaptive fusion using DNNs thus outperforms all aforementioned fusion techniques and allows a gain of 3.3 dB SDR compared to SDR-based static selection. It even outperforms oracle time-invariant fusion by almost 0.9 dB.

We might notice that DNNs with half and twice as many hidden units as the best architecture perform equivalently. From 7 to 896 hidden units, the performance monotonically increases with the number of hidden units. Starting from 1792 hidden units, the performance then drastically decreases. In these cases, the network tends to overfit the training data and thus loses its ability to generalize well. This tendency to overfitting is confirmed when comparing the number of network parameters to be estimated, given in Table I, and the size of our training dataset which is made of $200,424$ frames. Indeed, we can notice that the performance starts to decrease when the number of network parameters becomes larger than the number of training examples.

Number of hidden layers	Number of units per layer	Number of parameters	SDR (dB)
1	224	36,295	8.32
	448	72,583	8.46
	896	145,159	8.46
	1792	290,311	8.47
	3584	580,615	8.41
2	224	86,695	8.29
	448	273,735	8.32
3	224	137,095	8.22
	448	474,887	8.24
4	224	187,495	5.66
	448	676,039	5.66

TABLE I
NUMBER OF NETWORK PARAMETERS AND PERFORMANCE ON TEST SET OF SOME ARCHITECTURES

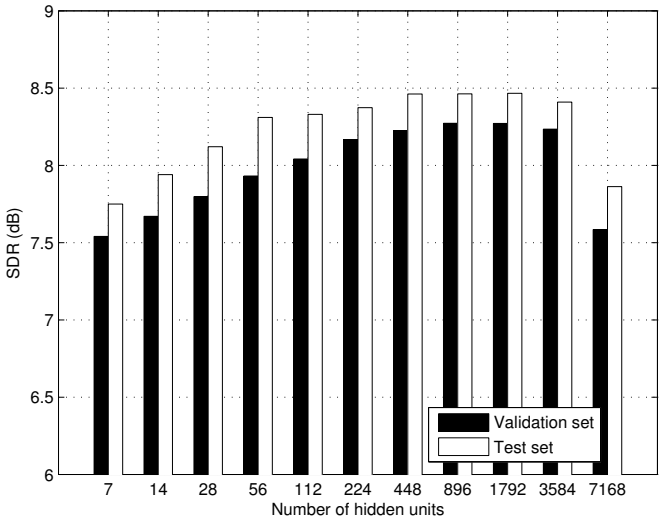

Fig. 6. SDRs obtained with different number of units with the best architecture on both validation and test sets

2) Influence of the number of layers: Deeper networks, *i.e.*, with more than one hidden layer, have also been tested. Some selected results are given in Table I. In particular, it emphasizes that networks with two and three hidden layers give slightly worst performance than the single-layer best architecture for comparable numbers of parameters. It also shows that the tested architectures comprising four hidden layers are suffering from overfitting.

3) Influence of the other parameters: Fig. 7 compares other architectures with the best one defined in the first subsection, for both the validation and the test set. SDRs are only represented for networks of one hidden layer composed of 896 units. For both validation and test, each bar of the graph refers to a neural network in which only one parameter has

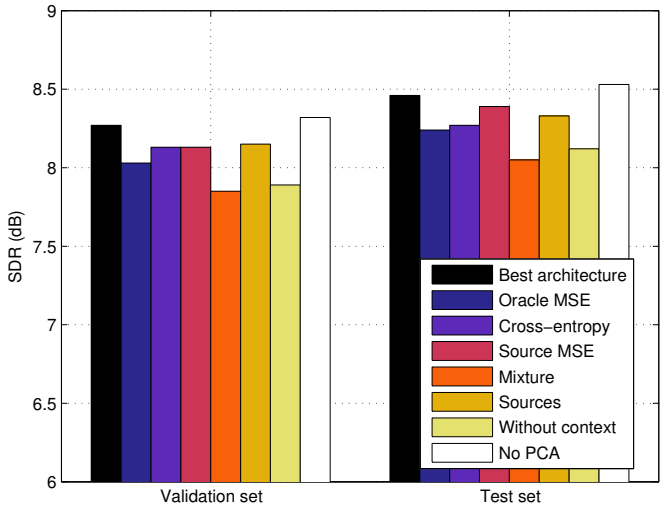

Fig. 7. Influence on the test SDR of other architecture parameters

been changed with respect to the best architecture.

The second, third and fourth bars, which respectively refer to networks using the oracle MSE, the cross-entropy and the source MSE as cost functions, show that choosing the SDR by frame as our objective function brought an appreciable improvement of approximately 0.1 to 0.2 dB SDR on the test set.

The remaining bars focus on the influence of the type of input. The fifth bar refers to a neural network whose input is composed of the mixture only, *i.e.*, the very first line of matrix (30), whereas the sixth bar refers to a network whose input is composed of the estimated sources only, *i.e.*, the M last lines of matrix (30). Both these results are outperformed by our best architecture. Similarly, the seventh bar which refers to a network in which no context is taken into account ($C = 0$) demonstrates that the handling of neighbouring frames can help improving fusion results by almost 0.35 dB SDR. Finally, the last bar shows that keeping all the variance of the input, *i.e.*, not reducing the input dimensionality by PCA, brings an improvement of 0.07 dB only. Note that this tiny improvement is obtained at the expense of a much longer training (3 days, 9 hours and 50 minutes without PCA against 1 hour and 30 minutes with PCA on a 64-bit Linux machine with an *NVIDIA Quadro 600* GPU and a quad-core *Intel Xeon* CPU).

4) Other attempts: The implementation of the dropout technique [55] has allowed to avoid overfitting for architectures with large numbers of parameters but at the expense of a higher training time and without outperforming single-layered networks. Recurrent neural networks composed of Bidirectional Long Short Term Memory (BLSTM) units have also been tested thanks to the *CURRENNT* toolkit [56]. However, their performance on the test set never exceeded 7.93 dB SDR.

VII. Experimental evaluation on music

The previous experimental section was dedicated to the evaluation of our fusion techniques on a speech enhancement task in which the models to be fused where all NMF-based. In this section, we finally propose to evaluate these techniques for the fusion of heterogeneous separation methods. We here focus on a singing voice extraction task which aims at separating the main vocal signal from its musical accompaniment.

A. Music database

In [6], a musical database has been gathered from the community music remixing website *ccMixter*[1]. It features 49 full-length stereo tracks from diverse musical genres. For our experiments, the tracks have been randomly divided into 5 groups of similar size in order to evaluate our fusion techniques by cross-validation. Furthermore, each of the 49 tracks has been cut into non-overlapping chunks of length comprised between 20 and 30 seconds, which results in a total of 308 excerpts.

For learning purposes, three of the five groups form the learning dataset whereas the two remaining groups respectively account for the validation and test sets. In order to evaluate our fusion techniques on all tracks, all experiments have been repeated five times so that each group of tracks has been used as a validation test and a test set once.

B. Separation techniques

Each excerpt has been processed with four different separation techniques, namely : the Instantaneous Mixture Model (IMM) proposed in [57], Robust Principal Component Analysis (RPCA) presented in [17], the Repeating Pattern Extraction Technique based on similarity (REPETsim) of [58] and the Kernel Additive Model (KAM) described in [6].

C. Fusion techniques

We here propose to compare static time-invariant fusion introduced in Section III to adaptive time-varying fusion using neural networks exposed in Section V, as the considered separation techniques do not fit in a common probabilistic framework in order to apply adaptive fusion as presented in Section IV.

D. Results

All results are gathered in Table II. They are evaluated in terms of SDR and averaged across all excerpts of each group and, in the last column, across all excerpts of the database. The four first lines present the results obtained with the four considered separation methods. The IMM outperforms other methods for each group as well as in average. Amongst the static fusion methods presented in the next three lines, MSE-based time-invariant fusion is the only one to outperform IMM, by 0.57 dB in average.

The results of adaptive time-varying fusion using DNN have been obtained with a single-layer network composed of 512 hidden units. As in Section VI, the input data was composed as defined in (30) with a context of size $C = 2$. We used the QERB transform as well with half-overlapping windows of 2048 samples and $F = 350$ frequency bins. The data have been furthermore standardized and the input dimensionality

[1]http://www.ccmixter.org

Method	Group 1	Group 2	Group 3	Group 4	Group 5	Average
IMM	3.49	4.33	3.16	2.55	2.83	3.30
RPCA	-0.92	-1.69	-3.65	-2.18	-1.22	-1.90
KAM	2.17	2.03	0.07	0.11	1.57	1.24
REPETsim	3.19	2.44	1.12	1.78	2.38	2.21
Fusion by mean	3.62	3.47	1.94	2.34	3.08	2.92
SDR-based time-invariant fusion	3.99	3.59	2.36	3.11	2.67	3.17
MSE-based time-invariant fusion	4.44	4.61	3.31	3.15	3.70	3.87
Oracle time-invariant fusion	*5.07*	*5.18*	*4.03*	*3.53*	*4.22*	*4.44*
Adaptive time-varying fusion using DNN	4.27	5.06	3.68	3.15	3.71	4.01
Oracle time-varying fusion	*6.88*	*7.07*	*5.89*	*5.28*	*6.08*	*6.28*

TABLE II
PERFORMANCE OF SEPARATION AND FUSION METHODS ON CCMIXTER TEST SETS

has been reduced via PCA. The training cost function was the source SDR as defined in (34).

Here again, adaptive time-varying fusion using DNN outperforms all other proposed fusion techniques, namely fusion by mean, SDR-based time-invariant fusion and MSE-based time-invariant fusion by 1.09, 0.84 and 0.15 dB respectively. Compared to the experiments in Section VI, the gain in terms of SDR is less important, which can be explained by the task complexity. Indeed, even if the size of the training set (approximately 200,000 frames per group) is comparable to the size of the training set in the speech enhancement task, we must highlight that the ccMixter database features much more variability in its contents because it mixes music excerpts from very heterogeneous genres. As such, we might thus improve fusion performance either by growing the database with more diverse examples or by reducing its variability.

VIII. CONCLUSION

In this paper, we introduced a general fusion framework that aims at combining several source estimates obtained either from a single separation technique with different parameter settings or from several distinct separation techniques. The proposed fusion rules is expressed in the time-domain and allows us to fuse a wide variety of separation techniques. We proposed three different ways to determine the fusion coefficients so that they can be adapted to the signal to be separated, according to a criterion expressed on the whole signal or at the time frame level. Our fusion techniques have been compared to standard selection in two different experiments. On a speech enhancement task, the method based on neural networks allowed us to gain more than 3 dB SDR by fusing several NMFs of different orders instead of selecting a unique one. The experiments conducted on a singing-voice extraction task furthermore showed that this learning method is also viable for fusing heterogeneous separation techniques.

To go further, we propose in the future to study other objective functions for the determination of fusion coefficients. Moreover, we plan to extend our fusion framework to the handling of frequency-varying and time-frequency-varying fusion rules.

REFERENCES

[1] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent component analysis and applications*. Academic press, 2010.

[2] 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 Processing Magazine*, vol. 31, no. 3, pp. 107–115, May 2014. [Online]. Available: https://hal.inria.fr/hal-00922378

[3] P. D. O'Grady, B. A. Pearlmutter, and S. T. Rickard, "Survey of sparse and non-sparse methods in source separation," *International Journal of Imaging Systems and Technology*, vol. 15, no. 1, pp. 18–33, 2005.

[4] O. Yılmaz and S. Rickard, "Blind separation of speech mixtures via time-frequency masking," *IEEE Transactions on Signal Processing*, vol. 52, no. 7, pp. 1830–1847, 2004.

[5] A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[6] A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet, "Kernel additive models for source separation," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4298–4310, 2014.

[7] A. Jourjine, S. Rickard, and O. Yı lmaz, "Blind separation of disjoint orthogonal signals: Demixing N sources from 2 mixtures," in *Proc. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, vol. 5, 2000, pp. 2985–2988.

[8] D. P. W. Ellis and R. J. Weiss, "Model-based monaural source separation using a vector-quantized phase-vocoder representation," in *Proc. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, vol. 5, 2006, pp. V–957–960.

[9] J.-L. Durrieu, G. Richard, B. David, and C. Févotte, "Source/filter model for unsupervised main melody extraction from polyphonic audio signals," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 18, no. 3, pp. 564–575, 2010.

[10] G. Mysore, P. Smaragdis, and B. Raj, "Non-negative hidden Markov modeling of audio with application to source separation," in *Proc. of Int. Conf. on Latent Variable Analysis and Signal Separation*. Springer, 2010, pp. 140–148.

[11] D. F. Rosenthal and H. G. Okuno, *Computational auditory scene analysis*. L. Erlbaum Associates Inc., 1998.

[12] B. Raj, R. Singh, and T. Virtanen, "Phoneme-dependent NMF for speech enhancement in monaural mixtures," in *Proc. of Interspeech*, 2011, pp. 1217–1220.

[13] N. Mohammadiha, J. Taghia, and A. Leijon, "Single channel speech enhancement using Bayesian NMF with recursive temporal updates of prior distributions," in *Proc. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, 2012, pp. 4561–4564.

[14] N. Moritz, M. R. Schädler, K. Adilŏglu, B. T. Meyer, T. Jürgens, T. Gerkmann, B. Kollmeier, S. Doclo, and S. Goetze, "Noise robust distant automatic speech recognition utilizing NMF based source separation and auditory feature extraction," *Proc. of CHiME-2013*, pp. 1–6, 2013.

[15] J. T. Geiger, F. Weninger, A. Hurmalainen, J. F. Gemmeke, M. Wöllmer, B. Schuller, G. Rigoll, and T. Virtanen, "The TUM + TUT + KUL approach to the 2nd CHiME challenge: Multi-stream ASR exploiting BLSTM networks and sparse NMF," *Proc. of CHiME-2013*, pp. 25–30, 2013.

[16] E. Vincent, J. Barker, S. Watanabe, J. Le Roux, F. Nesta, and M. Matassoni, "The second CHiME speech separation and recognition challenge: An overview of challenge systems and outcomes," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2013, pp. 162–167.

[17] 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. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, 2012, pp. 57–60.

[18] N. Bertin, R. Badeau, and G. Richard, "Blind signal decompositions for automatic transcription of polyphonic music: NMF and K-SVD on the benchmark," in *Proc. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, vol. 1, 2007, pp. I–65–68.

[19] A. Rabinovich, S. Belongie, T. Lange, and J. M. Buhmann, "Model order selection and cue combination for image segmentation," in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 1130–1137.

[20] V. Y. F. Tan and C. Févotte, "Automatic relevance determination in nonnegative matrix factorization," in *Proc. of Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, 2009.

[21] T. Virtanen, A. T. Cemgil, and S. Godsill, "Bayesian extensions to nonnegative matrix factorisation for audio signal modelling," in *Proc. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, 2008, pp. 1825–1828.

[22] X. Jaureguiberry, E. Vincent, and G. Richard, "Multiple-order nonnegative matrix factorization for speech enhancement," in *Proc. of Interspeech*, 2014, p. 4.

[23] I. Bloch, A. Hunter, A. Appriou, A. Ayoun *et al.*, "Fusion: General concepts and characteristics," *International Journal of Intelligent Systems*, vol. 16, no. 10, pp. 1107–1134, 2001.

[24] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[25] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of online learning and an application to boosting," in *Computational learning theory*. Springer, 1995, pp. 23–37.

[26] X. Jaureguiberry, G. Richard, P. Leveau, R. Hennequin, and E. Vincent, "Introducing a simple fusion framework for audio source separation," in *Proc. of IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2013, pp. 1–6.

[27] S. Chandna and W. Wenwu, "Improving model-based convolutive blind source separation techniques via bootstrap," in *Proc. of IEEE Statistical Signal Processing Workshop (SSP)*, 2014, pp. 424–427.

[28] J. Le Roux, S. Watanabe, and J. R. Hershey, "Ensemble learning for speech enhancement," in *Proc. of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, pp. 1–4.

[29] X. Jaureguiberry, E. Vincent, and G. Richard, "Variational Bayesian model averaging for audio source separation," in *Proc. of IEEE Statistical Signal Processing Workshop (SSP)*, 2014, pp. 33–36.

[30] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[31] E. Vincent, R. Gribonval, and M. D. Plumbley, "Oracle estimators for the benchmarking of source separation algorithms," *Signal Processing*, vol. 87, no. 8, pp. 1933–1950, 2007.

[32] D. P. Bertsekas, *Nonlinear programming*. Athena Scientific, 1999.

[33] J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky, "Bayesian model averaging: a tutorial," *Statistical science*, pp. 382–401, 1999.

[34] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Advances in Neural Information Processing Systems*, vol. 13, pp. 556–562, 2001.

[35] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence. with application to music analysis," *Neural Computation*, vol. 21, pp. 793–830, 2009.

[36] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural computation*, vol. 21, no. 3, pp. 793–830, 2009.

[37] M. Hoffman, D. M. Blei, and P. R. Cook, "Bayesian nonparametric matrix factorization for recorded music," in *Proc. of International Conference on Machine Learning (ICML)*, 2010, pp. 439–446.

[38] K. Adilŏglu and E. Vincent, "Variational Bayesian inference for source separation and robust feature extraction," Inria, Tech. Rep. RT-0428, 2012.

[39] A. Corduneanu and C. M. Bishop, "Variational Bayesian model selection for mixture distributions," in *Proc. of the 8th International Workshop on Artificial Intelligence and Statistics*, 2001, pp. 27–34.

[40] J. M. Bernardo *et al.*, "The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures," in *Proc. of Valencia International Meeting on Bayesian Statistics*, 2002, pp. 453–462.

[41] A. T. Cemgil, "Bayesian inference for nonnegative matrix factorisation models," *Computational Intelligence and Neuroscience*, 2009, article ID 785152.

[42] K. Katahira, K. Watanabe, and M. Okada, "Deterministic annealing variant of variational Bayes method," *Journal of Physics: Conference Series*, vol. 95, no. 1, 2008.

[43] T. F. Coleman and Y. Li, "An interior trust region approach for nonlinear minimization subject to bounds," *SIAM Journal on Optimization*, vol. 6, no. 2, pp. 418–445, 1996.

[44] B. Mathieu, S. Essid, T. Fillon, J. Prado, and G. Richard, "YAAFE, an easy to use and efficient audio feature extraction software," in *Proc. of ISMIR*, 2010, pp. 441–446.

[45] L. R. Rabiner and B.-H. Juang, *Fundamentals of speech recognition*. PTR Prentice Hall Englewood Cliffs, 1993, vol. 14.

[46] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[47] M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean *et al.*, "On rectified linear units for speech processing," in *Proc. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, 2013, pp. 3517–3521.

[48] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006, vol. 1.

[49] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[50] S. Duffner and C. Garcia, "An online backpropagation algorithm with validation error-based adaptive learning rate," *Artificial Neural Networks*, pp. 249–258, 2007.

[51] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proc. of the Python for Scientific Computing Conference (SciPy)*, 2010.

[52] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio, "Theano: new features and speed improvements," in *Proc. of Workshop on Deep Learning and Unsupervised Feature Learning (NIPS)*, 2012.

[53] E. Vincent, J. Barker, S. Watanabe, J. Le Roux, F. Nesta, and M. Matassoni, "The second 'CHiME' speech separation and recognition challenge: datasets, tasks and baselines," in *Proc. of IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, 2013, pp. 126–130. [Online]. Available: http://hal.inria.fr/hal-00796625

[54] E. Vincent, "Musical source separation using time-frequency source priors," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 91–98, 2006.

[55] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[56] F. Weninger, J. Bergmann, and B. Schuller, "Introducing CURRENNT–the Munich open-source CUDA RecurREnt Neural Network Toolkit," *Journal of Machine Learning Research*, vol. 15, p. 5, 2014.

[57] J.-L. Durrieu, B. David, and G. Richard, "A musically motivated mid-level representation for pitch estimation and musical audio source separation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1180–1191, 2011.

[58] Z. Rafii and B. Pardo, "Music/voice separation using the similarity matrix," in *Proc. of International Conference on Music Information Retrieval (ISMIR)*, 2012, pp. 583–588.