MaD TwinNet: Masker-Denoiser Architecture with Twin Networks for Monaural Sound Source Separation

Konstantinos Drossos*, Stylianos Ioannis Mimilakis†, Dmitriy Serdyuk‡,
Gerald Schuller†, Tuomas Virtanen*, and Yoshua Bengio‡§
*Tampere University of Technology, Tampere, Finland
Email: {firstname.lastname}@tut.fi
†Fraunhofer IDMT – Technical University of Ilmenau, Ilmenau, Germany
Email: {mis,shl}@idmt.fraunhofer.de
‡MILA, Université de Montréal, Montreal, Canada
Email: serdyuk@iro.umontreal.ca
§CIFAR Senior Fellow

***Abstract*—Monaural singing voice separation task focuses on the prediction of the singing voice from a single channel music mixture signal. Current state of the art (SOTA) results in monaural singing voice separation are obtained with deep learning based methods. In this work we present a novel recurrent neural approach that learns long-term temporal patterns and structures of a musical piece. We build upon the recently proposed Masker-Denoiser (MaD) architecture and we enhance it with the Twin Networks, a technique to regularize a recurrent generative network using a backward running copy of the network. We evaluate our method using the Demixing Secret Dataset and we obtain an increment to signal-to-distortion ratio (SDR) of 0.37 dB and to signal-to-interference ratio (SIR) of 0.23 dB, compared to previous SOTA results.**

I. Introduction

Music source separation is an active research area in the context of audio signal processing and machine learning. The task is to estimate musical sources from observed musical mixture signals. One of the biggest challenges in music source separation is the estimation of singing voice from monaural (i.e. single channel) mixture signals [1]. That is due to the high overlap that the sources exhibit in various signal representations [2].

Most approaches for source separation use time-frequency masking [3]. Modern state of the art (SOTA) methods estimate the mask with neural networks of various architectures. Given the strong and long-term temporal patterns and structures of music (e.g. rhythm, beat/tempo, melody), architectures that can model long time dependencies, e.g. recurrent neural networks (RNNs), seem to be a great fit for the music source separation task. But, local structures are usually dominating the learning signal, because the RNN focuses on the most recent information [4]. A known issue with the RNNs, that has been pointed out in many seminal works [4], e.g. [5], [6]. As a result, the long-term temporal patterns of music (e.g. tempo/beat, melody, and rhythm) might not be modeled correctly by an RNN, because the learning signal will be heavily influenced by the local structures [4].

The Twin Network (TwinNet) [4] is an effective way to regularize generative RNNs when the generation is conditioned on some input (e.g. past content) and make the RNN also take into account the expected future content. This technique uses a second RNN which generates the same output in the backward direction and ensures that the hidden states of the two networks are close.

In this paper we are presenting a method that performs music source separation, using the TwinNet, and capable to model long-term structures of music. We evaluate the proposed method by focusing on singing voice separation (i.e. separating the singing voice from musical mixtures). This work builds upon the method for masking and denoising simultaneously [3]. The main contributions of this paper are:

i We present a method that improves the previous objective SOTA SDR and SIR by 0.37 dB and 0.23 dB, respectively. Our method is less computationally intensive comparing to the one presented in [3] which uses the recurrent inference (an iterative method which allows deep learning architectures to have stochastic depth [7]);
ii We show that the TwinNet based regularization can be used for enhancing the results obtained by the Masker-Denoiser (MaD) architecture.

The rest of the paper is organized as follows. In Section II we give a brief overview of the related work for source separation task and approaches. The proposed method is thoroughly presented in Section III. The followed experimental procedure is described in Section IV and the obtained results are reported and discussed in Section V. Section VI concludes this work.

II. Related work

A common approach to estimate individual sources from monaural mixtures is to to apply time-varying filters to the mixture signal [8]. The most straightforward way to derive and apply these filters is to treat audio signals as wide-sense stationary and compute a time-frequency representation of the signals via the short-time Fourier transformation (STFT). Then, the source estimate can be obtained by:

$$\hat{\mathbf{Y}}^j = \mathbf{Y} \odot \mathbf{M}^j, \tag{1}$$

where $\mathbf{Y}$ and $\hat{\mathbf{Y}}^j \in \mathbb{C}^{M \times N}$ are the complex-valued STFT representations of the mixture signal vector $\mathbf{x}$ and the j-th source estimated signal $\hat{\mathbf{x}}^j$ respectively, with M overlapping time frames and N frequency sub-bands. $\mathbf{M}^j \in \mathbb{R}_{\geq 0}^{M \times N}$ is the j-th source-dependent filter, which we will refer to as *mask*, and $\odot$ denotes the Hadamard product. The question that remains open is how to compute $\mathbf{M}^j$.

In the case that all the j sources are known *a priori*, the source dependent masks are computed by employing ratios of the known sources' time-frequency representations [8]–[10]. Selecting an appropriate method for mask computation when all the sources are known is outside the scope of this work. Interested readers are kindly referred to the following works [8]–[10]. However, it is important to note that the mask computation is an *open* optimization problem [11] and in many cases assumptions about the source additivity [9] and the phase dependencies [8], [10] have to be made for many mask computations.

When the sources in an observed mixture are not known *a priori*, supervised approaches relying on deep learning based optimization have yielded SOTA results [1]. Deep learning approaches for singing voice separation can be distinguished in three categories. The first category includes methods that train a deep neural network (DNN) to predict $\mathbf{M}^j$, conditioned on features computed using $\mathbf{Y}$ [12] such as the magnitude spectrogram $\mathbf{V} = |\mathbf{Y}|$, where $|\cdot|$ denotes the matrix entry-wise absolute value operator. During training (when all sources are known for a given dataset), the pre-computation of the target $\mathbf{M}^j$ can rely on the ideal ratio mask (IRM, ${\mathbf{M}^j}_{\text{IRM}} \in [0, 1]$), defined as

$$\mathbf{M}^j_{\text{IRM}} = \frac{\mathbf{V}^j}{\sum\limits_{j' \in J} \mathbf{V}^{j'}} \text{ where,} \tag{2}$$

J is the total number of sources in a mixture. As can be seen, the form of the denominator implies the strong assumption that all the sources are additive. It must be noted here that the ideal amplitude mask (IAM, ${\mathbf{M}^j}_{\text{IAM}} \in \mathbb{R}_{\geq 0}$), a usual alternative way of computing $\mathbf{M}^j$ is defined as

$$\mathbf{M}^j_{\text{IAM}} = \frac{\mathbf{V}^j}{\mathbf{V}}, \tag{3}$$

is considered to be not appropriate for deep learning approaches, due to the lack of upper limit of the mask [8]. As shown in [13] and for tasks like singing voice separation, deep learning approaches that are trained to predict masks can be outperformed by methods that are not relying on pre-computed masks. The latter methods are discussed in the following paragraphs.

The second category follows the idea that was introduced in denoising autoencoders (DAEs) [14], [15]. More specifically, DNNs are trained to recover the target source magnitude spectrogram $\mathbf{V}^j$ from a corrupted version of $\mathbf{V}^j$. The corrupted version of $\mathbf{V}^j$ is assumed to be the observed mixture magnitude spectrogram $\mathbf{V}$ [16], [17]. For such methods, it was observed that the performance of the separation is highly dependent on post-processing steps that involved either the fusion of multiple trained DNNs [18] and/or post-processing steps such as masking of the mixture signal using the outcome of DNNs [13], [16], [18]–[20].

Aiming to encapsulate the process of masking into deep learning optimization routines, the approaches of the third category introduced skip connections to the DNNs. The skip connections propagate the mixture signal $\mathbf{V}$ through two information paths. The first information path is the typical forward propagation of $\mathbf{V}$ through the layers of the DNNs and the second information path allows $\mathbf{V}$ to directly reach the output of the DNNs which is used to mask $\mathbf{V}$ using Eq. (1), yielding the final DNN estimate.

Specifically, the work [19] employs deep RNNs and trained them to yield magnitude estimates for all the sources concurrently (i.e. $\hat{\mathbf{V}}^j \; \forall j$). The magnitude estimates are then given to a deterministic function involving the computation of the ratio of the estimated magnitudes. The ratio outputs the mask that is applied to $\mathbf{V}$ and is encapsulated through the training procedure [19]. That approach does not allow the deep RNNs to learn the masking process, but rather to output magnitude estimates that can be used to compute the mask. With the main ambition to also learn the masking process, the work of [17] proposed the usage of highway networks [21] that allow $\mathbf{V}$ to be masked directly by the output of a neural network layer. An extension to temporal sequences employing gated recurrent units (GRU) [22] was presented in [13] where the term skip-filtering connections was introduced. In [23] a deep, ladder-like-structured, convolutional neural network (CNN) was presented. The output of the CNN was used to mask the input $\mathbf{V}$ to the CNN to provide magnitude estimates of the singing voice.

The limitation of the above methods is that highway networks of [17] do not compute any latent variables that can be used for denoising [14]. On the other hand, CNN architectures are prone to learn statistical irregularities of the data [24] making these two architectures not robust against data perturbations [14], [24], while the GRU encoder-decoder of [13] is not robust against interferences from other sources concurrently active in the mixture signal [3]. To tackle these problems, the *masker-denoiser* (MaD) architecture was introduced in [3]. This architecture builds upon [13] by incorporating a sparse transformation and a stochastic-depth optimization [7] step that are used to generate the mask applied to $\mathbf{V}$ (i.e. the masker). As a final step a DAE with skip-filtering connections (i.e the denoiser) is responsible for eliminating remaining interferences from other music sources [3].

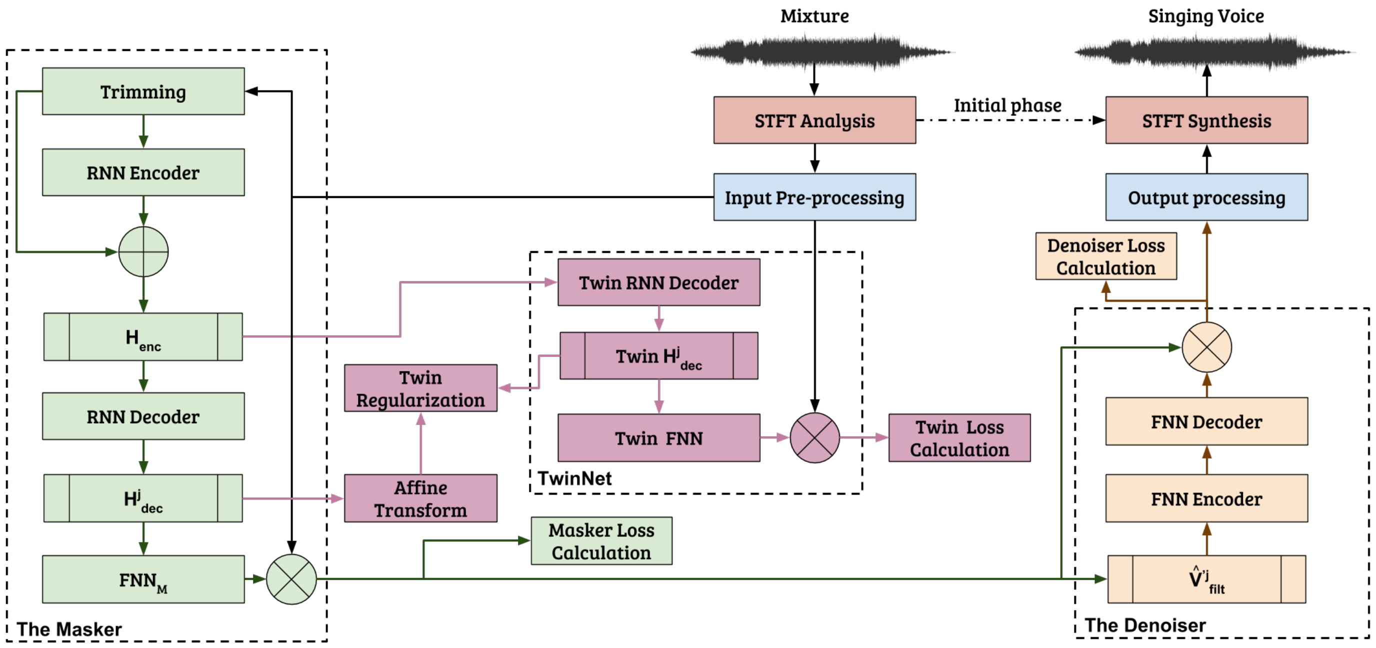

Fig. 1. Illustration of the proposed method. The parts in magenta color are used only during training.

Although the approach in [3] provided SOTA results in deep learning based monaural singing voice separation, the quality of the decoded mask is relying on stochastic optimization of a data-driven depth of the GRU decoder via recurrent inference [7]. As a consequence and as the reported results in the experimental procedure of [3] suggest, the recurrent inference imposes a computationally cumbersome optimization process of the model given the available data for singing voice separation.

To tackle that, we propose to replace the recurrent inference process by a recently developed Twin Network (TwinNet) [4]. Similarly to the bidirectional RNN, the TwinNet employs a backward running network. The bidirectional RNNs are limited to be used for the representation learning when the TwinNet is applied for generative RNN. In addition to the original RNN, the TwinNet adds a second recurrent network that is a copy of the original except that it aggregates the output in the backward direction. Second, the TwinNet adds a term to the loss function that depends on a trainable function of the backward running network. This loss function pushes together the hidden states of the forward net and the backward net for co-temporal time-steps. This cost ensures that the hidden state of the forward network encodes information stored in the state of the backward network. In other words, the TwinNet cost encourages the RNN to anticipate the future encoded in the backward running RNN, resulting in better modeling of both past and future context with the RNN.

We use the TwinNet under the hypothesis that the time-frequency mask for singing voice separation should remain the same regardless of the direction (i.e. forward or backward in time) that one chooses to traverse a sequence of frames with time-frequency representation of the mixture signal.

III. Proposed method

Our proposed method takes as an input the raw audio signal of a mixture of sources, and outputs the raw audio signal of the target source (i.e. the singing voice). The method uses a deep neural network architecture, which consists of two parts. The first part takes the mixture magnitude spectrogram of the input time-domain audio as an input, estimates the target source magnitude spectrogram from the mixture by predicting a time-frequency filter and applying it to the input, and outputs the estimated magnitude spectrogram of the target source. The second part takes the output of the first part as an input, predicts and applies a denoising mask, and outputs a filtered version of its input. Since the input to the first part is the mixture signal and the output is an estimate of the signal of the target source, we frame the time-frequency filter of the first part as a time-frequency mask. Similarly, since the input to the second part is a representation of the target source and its output is the same representation of the same source, we frame the time-frequency filter of the second part as a denoising filter. The first part of our method is denoted as *The Masker* and the second as the *The Denoiser*.

The Masker and the Denoiser are neural network architectures based on DAEs [14]. According to the initial paper of DAEs [14], DAEs try to learn stochastically the manifold of the clean data. Source separation can be understood as a process that transforms the samples of the corrupted data manifold (i.e. the mixture) into samples that reside in the manifold of clean data (i.e. the target source) [25]. This transformation for audio data can be understood using time-frequency masking [8]. We want to include in our optimization graph the generation of the mask and the denoising filter. Consequently, we setup our method in a way that the Masker will predict the mask that will be applied to the input magnitude spectrogram, and the denoiser will predict the values of the denoising filter which is applied to the output of the Masker. To do so, we employ the skip-filtering connections [13], which allow us to define the magnitude spectrogram of the target source as the target of the Masker and the Denoiser, but the output of the last neural network layer in the Masker and the Denoiser would be the mask and the denoising filter, respectively.

We claim that the direction that one chooses to view the time-frequency representation of the mixture signal (i.e. forward or backward in time), does not affect the mask that is used in order to separate the singing voice from the mixture. Additionally, we hypothesize that a reverse traversing of the time-frequency representation of the mixture, will make the Masker to learn and anticipate the strong temporal patterns and structures of music. For these reasons, we use the recently proposed TwinNet for regularizing the Masker during training [4]. Our method is illustrated in Figure 1 and thoroughly presented below.

A. Input pre-processing

The input to our method is the vector of audio samples of the mixture signal $\mathbf{x} = [x_1, x_2, \dots, x_N]$, $x_n \in [-1, 1]$, sampled at 44.1 kHz. We transform the input mixture signal into a time-frequency representation by using the STFT. For the STFT, we use overlapping frames of $N = 2049$ samples (≈ 46 milliseconds), segmented using the Hamming window function, and zero padded to $N' = 4096$ samples. The hop size is set to 384 samples (≈ 9 milliseconds). After the STFT, we retain only the first N frequency sub-bands (i.e. up to the Nyquist frequency, including the DC term), resulting in the time-frequency representation of $\mathbf{x}$, $\mathbf{Y} \in \mathbb{C}^{M\times N}$, with $\mathbf{V} \equiv |\mathbf{Y}|$ to be the magnitude of $\mathbf{Y}$. From $\mathbf{V}$, we create overlapping subsequences of length T, that overlap by a factor of $L\times 2$, in order to use context information from previous (L) and next (again L) frames. This results in having $B = \lceil M/T \rceil$ sequences of the form $\mathbf{V}_{\text{in}} \in \mathbb{R}_{\geq 0}^{T\times N}$ for the whole input signal $\mathbf{x}$, where $\lceil\cdot\rceil$ is the ceiling function. Each $\mathbf{V}_{\text{in}}$ is used as an input to the next part of our method, which is the Masker.

B. The Masker

The Masker consists of a frequency trimming process, a bi-directional recurrent neural network (Bi-RNN) encoder (RNN$_{\text{enc}}$), a forward RNN decoder (RNN$_{\text{dec}}$), a sparsifying transform which is implemented with a rectified linear unit (ReLU) and a feed-forward neural network (FNN), and the skip-filtering connections. The input to the masker is the sequence $\mathbf{V}_{\text{in}}$ and its output is the predicted magnitude of the j-th target source, $\hat{\mathbf{V}}'^{j}_{\text{filt}} \in \mathbb{R}_{\geq 0}^{T'\times N}$, with $T' = T - 2L$.

$\mathbf{V}_{\text{in}}$ is trimmed to $\mathbf{V}_{\text{tr}} \in \mathbb{R}_{\geq 0}^{T\times F}$, with $F = 744$. This is done in order to reduce the dimensionality of the RNN$_{\text{enc}}$ and thus the number of training parameters. Consequently, information up to 8 kHz is retained. Since most of the relevant information for the singing voice is up to 8 kHz, the aforementioned reduction is considered to not have a great impact to the process of the Masker.

$\mathbf{V}_{\text{tr}}$ is used as an input to the RNN$_{\text{enc}}$. The forward RNN of the RNN$_{\text{enc}}$ takes the sequence $\mathbf{V}_{\text{tr}}$ as an input, and the backward RNN the $\overleftarrow{\mathbf{V}}_{\text{tr}} = [\mathbf{v}_{\text{tr}T}, \dots, \mathbf{v}_{\text{tr}t}, \dots, \mathbf{v}_{\text{tr}1}]$. The hidden states of the forward and the backward RNNs of the RNN$_{\text{enc}}$ and at frame t, $\mathbf{h}_t$ and $\overleftarrow{\mathbf{h}}_t$ respectively, are concatenated and summed to the input (with residual connections) as

$$\mathbf{h}_{\text{enc}_t} = [(\mathbf{h}_t + \mathbf{v}_{\text{tr}t})^{\text{T}}, (\overleftarrow{\mathbf{h}}_t + \overleftarrow{\mathbf{v}}_{\text{tr}t})^{\text{T}}]^{\text{T}} \ , \tag{4}$$

leading to the output of the RNN$_{\text{enc}}$, $\mathbf{H}'_{\text{enc}} \in \mathbb{R}_{\geq -1}^{T\times 2F}$,

$$\mathbf{H}'_{\text{enc}} = [\mathbf{h}_{\text{enc}_1}, \mathbf{h}_{\text{enc}_2}, \dots, \mathbf{h}_{\text{enc}_T}] \ . \tag{5}$$

Because we want the RNN$_{\text{dec}}$ to focus only on the frames of $\mathbf{H}'_{\text{enc}}$ that are relevant to the sequence from which we want to extract the target source (i.e. the frames in the range $[1 + L, T - L]$), we drop the first and the last L $\mathbf{h}_{\text{enc}_t}$. This results to the output $\mathbf{H}_{\text{enc}} \in \mathbb{R}_{\geq -1}^{T'\times 2F}$ of the RNN$_{\text{enc}}$,

$$\mathbf{H}_{\text{enc}} = [\mathbf{h}_{\text{enc}_{1+L}}, \mathbf{h}_{\text{enc}_{2+L}}, \dots, \mathbf{h}_{\text{enc}_{T-L}}] \ . \tag{6}$$

$\mathbf{H}_{\text{enc}}$ is used as an input to the RNN$_{\text{dec}}$, which outputs the hidden states $\mathbf{H}^{j}_{\text{dec}} \in [-1, 1]^{T',F}$. Consequently, the $\mathbf{H}^{j}_{\text{dec}}$ is used as an input to the sparsifying transform (the FNN with the ReLU) to obtain the time-frequency mask for the target source j, $\tilde{\mathbf{M}}^{j} \in \mathbb{R}_{\geq 0}^{T'\times N}$, as

$$\tilde{\mathbf{M}}^{j} = g(\mathbf{H}^{j}_{\text{dec}}\mathbf{W}_{\text{FNN}_{\text{M}}} + \mathbf{b}_{\text{FNN}_{\text{M}}}) \ , \tag{7}$$

where $\mathbf{W}_{\text{FNN}_{\text{M}}}$ and $\mathbf{b}_{\text{FNN}_{\text{M}}}$ are the weight matrix and bias vector of the FNN, respectively, and g is the element wise ReLU activation. The output of the Masker is obtained by employing the skip-filtering connections as

$$\hat{\mathbf{V}}'^{j}_{\text{filt}} = \tilde{\mathbf{M}}^{j} \odot \mathbf{V}'_{\text{in}} \ , \tag{8}$$

where $\mathbf{V}'_{\text{in}} = [\mathbf{v}_{\text{in}1+L}, \mathbf{v}_{\text{in}2+L}, \dots, \mathbf{v}_{\text{in}T-L}]$.

C. TwinNet architecture and regularization

The usage of a backward RNN as a regularizer for a forward RNN during training has been proposed in [4], [26]. The target is to make the forward RNN capable to model better the long-term temporal structures and patters in the sequences that the RNN processes. To do so, the authors in [4] use the hidden states of the backward RNN as a target for the hidden states of the forward RNN in a deterministic way, as can be seen in Figure 2.

More specifically, in the original proposal of the TwinNet [4], the authors consider a sequence of inputs $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_T]$, aiming at estimating the density $p(\mathbf{S})$. To do so, they target at maximizing the log-likelihood $\log p(\mathbf{S})$, using a forward RNN to process $\mathbf{S}$ and a non-linear transformation on top of the RNN, for predicting $p_{\text{f}}(\mathbf{s}_t|\mathbf{s}_{<t}) = \Psi_{\text{f}}(\overrightarrow{\mathbf{h}}_t)$. $\overrightarrow{\mathbf{h}}_t$ is the output of the forward RNN and $\Psi_{\text{f}}(\cdot)$ is the non-linear transformation applied on top of the forward RNN (e.g. a softmax).

To encourage the forward RNN to take into account up-coming (future) inputs, the authors in [4] use a backward

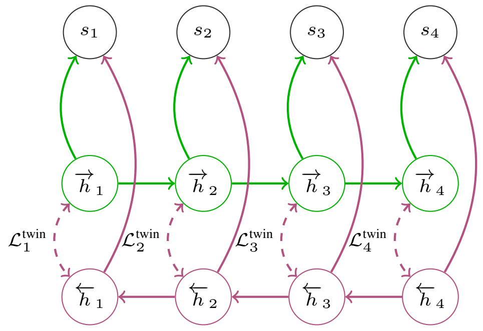

Fig. 2. Illustration of the hidden states regularization with the TwinNet. Forward RNN is depicted in green color. TwinNet (backward RNN) and TwinNet regularization are depicted in magenta color

RNN to predict $p_{\text{b}}(\mathbf{s}_t|\mathbf{s}_{<t}) = \Psi_{\text{b}}(\overleftarrow{\mathbf{h}}_t)$, where $\overleftarrow{\mathbf{h}}_t$ is the output of the backward RNN and $\Psi_{\text{b}}(\cdot)$ is a non-linear transformation applied on top of the backward RNN. Additionally, to regularize the learning process, they calculate the distance $\mathcal{L}_t^{\text{twin}} = ||f(\overrightarrow{\mathbf{h}}_t) - \overleftarrow{\mathbf{h}}_t||_2$, where f is a learned affine transformation. All the above components are jointly optimized by maximizing

$$Q = \sum_t \log p_{\text{f}}(\mathbf{s}_t|\mathbf{s}_{<t}) + \log p_{\text{b}}(\mathbf{s}_t|\mathbf{s}_{>t}) - \mathcal{L}_t^{\text{twin}}. \tag{9}$$

The $\mathcal{L}_t^{\text{twin}}$ at Eq. (9), is the term that encourages the forward RNN to take into account the future inputs.

During the evaluation/testing process, the backward RNN and the associated parts (i.e. the magenta parts in Figure 2) are not used. Finally, in the optimization graph, all the preceding parts of the network from the backward RNN are not receiving a gradient signal from the objective of the backward RNN. This means that the input to the backward RNN is disconnected from the computation graph and is used only to optimize the backward RNN.

In our work, we use the TwinNet only during training to regularize the $\mathbf{H}_{\text{dec}}^j$. We claim that the RNN_{dec} can greatly benefit from compensating for the future time frames, due to the strong temporal patterns and structures of music. We set up the TwinNet as a duplicate/twin of the RNN_{dec}, the sparsifying transform, and the skip-connections, as can be seen in Figure 1. The input to the TwinNet is the $\mathbf{H}_{\text{enc}}$ and its output is the $\mathbf{H}_{\text{twin}}^j$, calculated exactly as $\mathbf{H}_{\text{dec}}^j$. We apply

$$\mathcal{L}^{\text{twin}} = \sum_t ||f(\mathbf{h}_{\text{dec}_t}) - \mathbf{h}_{\text{twin}_t}||_2 \tag{10}$$

as the cost for the regularization of the RNN_{dec}. The affine transform f (obtained from a feed-forward network without any non-linearity applied to its output) is not used during evaluation/testing. In our work we transmit the gradient signal from the TwinNet back to the RNN_{enc}, in order to imbue the compensation for future values to the encoding part of the Masker as well. The output of the TwinNet that is used to optimize it is $\hat{\mathbf{V}}_{\text{twin}}^j$ and is obtained exactly as $\hat{\mathbf{V}}_{\text{filt}}^{\prime j}$.

D. The Denoiser

We expect that the implemented masking process by the Masker will allow interferences from other active sources to the magnitude spectrogram of the separated source. For that reason, we employ an extra learnable time-frequency filter applied to the $\hat{\mathbf{V}}_{\text{filt}}^{\prime j}$, in order to refine the latter and make it as close as possibly to the $\mathbf{V}^{j}{}_{\text{in}}^{\prime}$. We perceive this process as denoising, hence we term the module that implements it the Denoiser.

The Denoiser consists of two FNNs, the FNN_{enc} and FNN_{dec}, takes $\hat{\mathbf{V}}_{\text{filt}}^{\prime j}$ as an input, and outputs the $\hat{\mathbf{V}}_{\text{filt}}^{j} \in \mathbb{R}_{\geq 0}^{T' \times N}$. The two FNNs of the Denoiser are set up in an DAE fashion and have shared weights through time. The FNN_{enc} takes $\hat{\mathbf{V}}_{\text{filt}}^{\prime j}$ as the input and outputs $\mathbf{H}_{\text{FNN}_{\text{enc}}} \in \mathbb{R}_{\geq 0}^{T' \times N''}$, with $N'' = \lfloor N/2 \rfloor$ and $\lfloor \cdot \rfloor$ to be the floor function, as

$$\mathbf{H}_{\text{FNN}_{\text{enc}}} = g(\hat{\mathbf{V}}_{\text{filt}}^{\prime j} \mathbf{W}_{\text{FNN}_{\text{enc}}} + \mathbf{b}_{\text{FNN}_{\text{enc}}}) \ , \tag{11}$$

where $\mathbf{W}_{\text{FNN}_{\text{enc}}}$ and $\mathbf{b}_{\text{FNN}_{\text{enc}}}$ are the weights matrix and bias vector, respectively, of the FNN_{enc}. The FNN_{dec} accepts the $\mathbf{H}_{\text{FNN}_{\text{enc}}}$ and outputs the $\mathbf{H}_{\text{FNN}_{\text{dec}}} \in \mathbb{R}_{\geq 0}^{T' \times N}$, as

$$\mathbf{H}_{\text{FNN}_{\text{dec}}} = g(\mathbf{H}_{\text{FNN}_{\text{enc}}} \mathbf{W}_{\text{FNN}_{\text{dec}}} + \mathbf{b}_{\text{FNN}_{\text{dec}}}) \ , \tag{12}$$

where $\mathbf{W}_{\text{FNN}_{\text{dec}}}$ and $\mathbf{b}_{\text{FNN}_{\text{dec}}}$ are the weights matrix and bias vector, respectively, of the FNN_{dec}. The final output of the Denoiser, $\hat{\mathbf{V}}_{\text{filt}}^{j}$, is obtained as

$$\hat{\mathbf{V}}_{\text{filt}}^{j} = \mathbf{H}_{\text{FNN}_{\text{dec}}} \odot \hat{\mathbf{V}}_{\text{filt}}^{\prime j} \ . \tag{13}$$

E. Output processing

By iterating through all the overlapping subsequences of the analyzed input signal, the estimates from Eq. (13) are aggregated together and reshaped to form the magnitude spectrogram of the j-th target source $\hat{\mathbf{V}}^j \in \mathbb{R}_{\geq 0}^{M \times N}$. For the target source, we obtain the complex-valued representation by means of the Griffin-Lim algorithm (least squares error estimation from modified STFT magnitude) [27], which uses the synthesis window and mixture's phase information. Inverse STFT is then applied to compute the time-domain samples of the target source $\hat{\mathbf{x}}^{j=1}$.

F. Implementation and training details

We jointly train all components of our method. We treat $\hat{\mathbf{V}}_{\text{filt}}^{\prime j}$, $\hat{\mathbf{V}}_{\text{filt}}^{j}$, and $\mathbf{V}^{j}{}_{\text{twin}}$ as matrices with unnormalized probabilities (i.e. the values are not summing up to one), allowing us to use the generalized Kullback-Leibler divergence as cost function, and the employed objective is

$$\begin{aligned} \mathcal{L} =& \mathcal{L}_{\text{D}} + \mathcal{L}_{\text{M}} + \mathcal{L}_{\text{TW}} + 0.5\mathcal{L}^{\text{twin}} \\ & + \lambda_1 |\text{diag}\{\mathbf{W}_{\text{FNN}_{\text{M}}}\}|_1 + \lambda_2 ||\mathbf{W}_{\text{FNN}_{\text{dec}}}||_2^2, \text{ where} \end{aligned} \tag{14}$$

$$\mathcal{L}_{\text{D}} = D_{\text{KL}}(\mathbf{V}^j \ || \ \hat{\mathbf{V}}_{\text{filt}}^{j}), \tag{15}$$

$$\mathcal{L}_{\text{M}} = D_{\text{KL}}(\mathbf{V}^j \ || \ \hat{\mathbf{V}}_{\text{filt}}^{\prime j}), \tag{16}$$

$$\mathcal{L}_{\text{TW}} = D_{\text{KL}}(\mathbf{V}^j \ || \ \mathbf{V}^{j}{}_{\text{twin}}), \text{ and} \tag{17}$$

$\mathcal{L}_{\text{M}}$, $\mathcal{L}_{\text{D}}$, and $\mathcal{L}_{\text{TW}}$ are the objectives of the Masker, the Denoiser, and the TwinNet respectively, D_{KL} is the generalized Kullback-Leibler divergence, $\lambda_1 = 1 \times 10^{-2}$ and

$\lambda_2 = 1 \times 10^{-4}$ are regularization terms, $|\cdot|_1$ is the ℓ_1 vector norm, and $||\cdot||_2^2$ is the L_2 matrix norm. diag$\{\mathbf{W}_{\text{FNN}_\text{M}}\}$ is the main diagonal of the weight matrix of the FNN of the Masker (i.e. the elements w_{ij} of $\mathbf{W}_{\text{FNN}_\text{M}}$ with $i = j$). The values for λ_1 and λ_2 are after the vanilla version of the Masker-Denoiser architecture [3]. We employ the $\lambda_1|\text{diag}\{\mathbf{W}_{\text{FNN}_\text{M}}\}|_1$ in order to enforce the FNN of the Masker not to have energy in its main diagonal. We observed that high energy in the main diagonal of the $\mathbf{W}_{\text{FNN}_\text{M}}$ results to a source-dependent activity detector, rather than a source-dependent filter. We employ the $\lambda_2||\mathbf{W}_{\text{FNN}_\text{dec}}||_2^2$ as a weight decay to avoid overfitting of the Denoiser and induce a sparsity factor.

All RNNs are GRUs and are initialized using the orthogonal initialization technique from [28] and all other matrices using random samples from a normal distribution [29]. Biases are initialized with zeros. All parameters are jointly optimized using the Adam algorithm [30]. The learning rate is set equal to 10^{-4}, the batch size to 16, and a gradient L_2 norm clipping equal to 0.5 is applied. Out method is implemented using the PyTorch framework[1] and the code can be found online[2]. One complete pass of all the training data (i.e. one epoch) by the MaD TwinNet network (i.e. excluding any pre- or post-processing of the data), needs approximately 800 seconds on a NVIDIA P100 GPU. For the same dataset, one epoch, and on the same P100 GPU, the vanilla MaD (the NRI case in [3]) needs around 600 seconds and the MaD with recurrent inference (the RIS case in [3]) needs around 1600 seconds.

IV. Experimental procedure

We assess the performance of our proposed method by focusing on the task of singing voice separation. We use the development subset of Demixing Secret Dataset[3] (DSD100) and the non-bleeding/non-instrumental stems of MedleydB [31] for training our approach in a supervised fashion. Our total training set consists of 116 mixtures and their corresponding individual sources. The evaluation subset of DSD100 (50 mixtures and corresponding sources) is used for measuring the objective performance of our methods in terms of signal-to-distortion ratio (SDR) and signal-to-interference ratio (SIR), as proposed by the music source separation evaluation campaign (SiSeC) [1]. Each multi-track contained in the available data is used to generate a monaural version of each of the four sources by averaging the two available channels. The target that we used for training (i.e. $\mathbf{V}^j$) is the outcome of the ideal ratio masking process [8], scaled by a factor of 2. The masking and the scaling processes are performed to avoid the inconsistencies in time delays and/or mixing gains between the mixture signal and the singing voice, which were apparent in the stems of the MedleydB dataset. Through our experiments it was observed that inconsistencies in the mixing gains yielded target source estimates that lacked amplitude, slightly decreasing the performance of the Masker. The length of the sequences is set to $T = 60$ (approximately equal to 0.5 seconds), and the context information parameter to $L = 10$. The values for T and L are chosen after the initial proposal of the MaD architecture [3].

We compared our proposed method with established SOTA approaches that solely deal with monaural singing voice separation. These approaches and their corresponding results are listed at the on-line results page of the signal separation evaluation campaign for music signals (SiSeC-MUS)[4]. The approach denoted as GRA3 [12] is a DNN supervised approach to yield estimates for the ideal and/or IRM masks that are used to process the mixture magnitude spectrogram. The method denoted as CHA [32] is a CNN approach that yields estimates of all sources and then post-process them by using an IRM mask. STO2 is a DNN approach that operates on the common-fate signal representation [33].

MM-RINF and MIM-NINF are MaD based methods, but *none of them uses the TwinNet regularization*. The MIM-RINF approach incorporates the recurrent inference stochastic optimization for the RNN_dec, using a maximum number of 10 iterations and a termination threshold equal to 1×10^{-3}. The MIM-NINF approach does not incorporate the recurrent inference optimization procedure and it is the vanilla MaD architecture. Finally a supervised method based on robust principal component analysis (RPCA) [34] for singing voice separation is also taken into consideration for the objective assessment, and this RPCA-based method is denoted as JEO2. The results from all the above mentioned approaches were obtained from the reported results of [1] and [3], following the same evaluation data and protocol proposed by SiSeC-MUS in [1].

V. Results

In Figures 3 and 4 are the box plots for the obtained results, for the employed metrics (i.e. SDR and SIR), and compared with the previous SOTA results. In Table I are the median values for the obtained results and compared with the same previous approaches. We use the median value because that is the one proposed and used by the SiSeC. An online demo of the separated audio sequences is available.[5]

TABLE I
THE MEDIAN VALUES FOR SDR AND SIR OF THE PROPOSED METHOD AND PREVIOUS APPROACHES.

	Metric	
Approach	**SDR**	**SIR**
GRA3	-1.74	1.28
CHA	1.58	5.17
MIM-NINF	3.63	7.06
STO2	3.92	6.75
JEO2	4.07	6.09
MIM-RINF	4.20	7.94
MaDTwinNet	**4.57**	**8.17**

As can be seen in the Table I and in the Figures 3 and 4 for all the metrics, the MaD TwinNet method achieves higher

[1]http://pytorch.org/
[2]https://github.com/dr-costas/mad-twinnet
[3]http://www.sisec17.audiolabs-erlangen.de
[4]https://sisec17.audiolabs-erlangen.de/#/results/1/4/2
[5]http://arg.cs.tut.fi/demo/mad-twinnet/

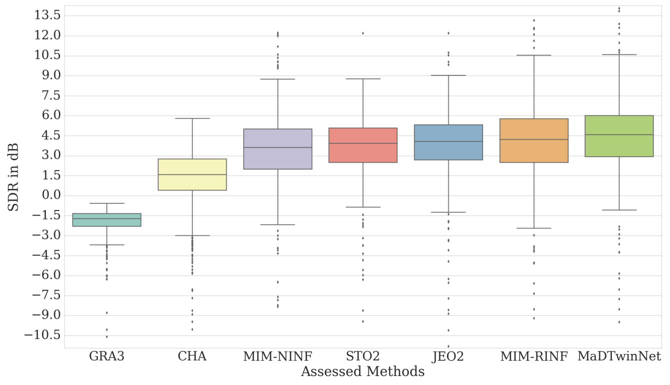

Fig. 3. Box-plots for the SDR obtained by MaDTwinNet and previous SOTA approaches.

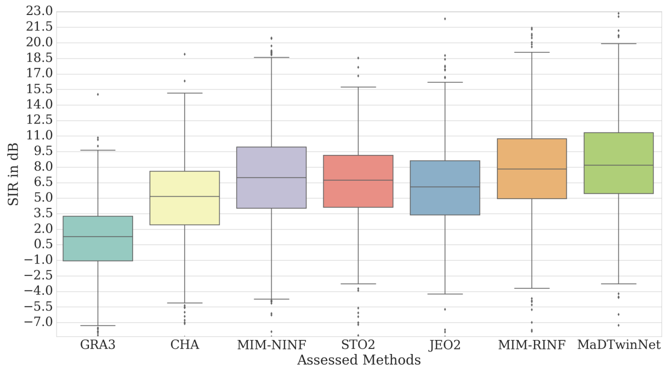

Fig. 4. Box-plots for the SIR obtained by MaDTwinNet and previous SOTA approaches.

scores than the previous approaches. Specifically, in Figure 3 can be seen that the MaD TwinNet achieves better score than the previous better approach (i.e. MIM-RINF). At the same time, the MaD TwinNet has smaller range of values from the MIM-RINF approach, indicating more consistency at the expected results than the MIM-RINF approach. Since the basic difference from the MIM-* approaches is the TwinNet and the recurrent inference, the results for SDR indicate that the usage of the TwinNet leads to more robust methods. Additionally, MaD TwinNet surpasses in all aspects all other presented approaches in Figure 3.

Almost the same trend can be observed at the results for the SIR, depicted at Figure 4. Again, the MaDTwinNet surpasses all previous monaural approaches in the terms of the achieved SIR. The MaDTwinNet approach seems to yield higher SIR values, compared to the MIM-RINF approach. Compared with the results for the SDR, it can be seen clearly the the TwinNet regularization increase the performance of the MaD architecture.

VI. Conclusions

In this paper we proposed a method for music source separation, able to model both past and future context of a musical sound source. We augmented our previously proposed MaD architecture with the recently proposed TwinNet. The Masker (of the MaD architecture) outputs a first estimate of the magnitude spectrogram of the targeted source, and the Denoiser enhances this first estimate by removing artifacts introduced by the Masker. We used the TwinNet to regularize the Masker. We evaluated our proposed method using the free DSD dataset and focusing on the singing voice separation task. The results showed an increase to the previous obtained SOTA results on the same task. Specifically, we reached to an increase of 0.37 dB and 0.23 dB to SDR and SIR, respectively. The obtained results show that the TwinNet can enhance the performance of the MaD architecture.

As following work we propose the focus towards end-to-end methods, meaning that the neural network should receive as an input and produce as an output audio samples directly. This will make the time-frequency transformation to be included in the optimization graph and, probably, yield superior results.

Acknowledgments

Part of the computations leading to these results was performed on a TITAN-X GPU donated by NVIDIA to K. Drossos. K. Drossos and T. Virtanen wish to acknowledge CSC-IT Center for Science, Finland, for computational resources. D. Serdyuk would like to acknowledge the support of the following agencies for research funding and computing support: Samsung, NSERC, Calcul Québec, Compute Canada, the Canada Research Chairs, and CIFAR. S.-I. Mimilakis is supported by the European Union's H2020 Framework Programme (H2020-MSCA-ITN-2014) under grant agreement no 642685 MacSeNet. Part of the research leading to these results has received funding from the European Research Council under the European Unions H2020 Framework Programme through ERC Grant Agreement 637422 EVERYSOUND. The authors would like to thank P. Magron and G. Naithani (TUT, Finland) for their valuable comments and feedback during the writing process.

References

[1] A. Liutkus, F.-R. Stöter, Z. Rafii, D. Kitamura, B. Rivet, N. Ito, N. Ono, and J. Fontecave, "The 2016 signal separation evaluation campaign," in *Latent Variable Analysis and Signal Separation: 13th International Conference, LVA/ICA 2017*, 2017, pp. 323–332.

[2] D. Giannoulis, D. Barchiesi, A. Klapuri, and M. D. Plumbley, "On the disjointess of sources in music using different time-frequency representations," in *2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA 2011)*, Oct. 2011, pp. 261–264.

[3] S.-I. Mimilakis, K. Drossos, J.-F. Santos, G. Schuller, T. Virtanen, and Y. Bengio, "Monaural singing voice separation with skip-filtering connections and recurrent inference of time-frequency mask," in *43rd International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018)*, 2018.

[4] D. Serdyuk, N.-R. Ke, A. Sordoni, A. Trischler, C. Pal, and Y. Bengio, "Twin Networks: Matching the future for sequence generation," *CoRR*, vol. abs/1708.06742, 2017.

[5] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, Mar 1994.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[7] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," *CoRR*, vol. abs/1603.09382, 2016.

[8] H. Erdogan, J. R. Hershey, S. Watanabe, and J. Le Roux, "Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks," in *40th International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, Apr. 2015, pp. 708–712.

[9] A. Liutkus and R. Badeau, "Generalized Wiener filtering with fractional power spectrograms," in *40th International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, Apr. 2015, pp. 266–270.

[10] S. Voran, "The selection of spectral magnitude exponents for separating two sources is dominated by phase distribution not magnitude distribution," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA 2017)*, Oct. 2017.

[11] D. FitzGerald and R. Jaiswal, "On the use of masking filters in sound source separation," in *15th International Conference on Digital Audio Effects (DAFx-12)*, Sep. 2012.

[12] E.-M. Grais, G. Roma, A.J.R. Simpson, and M.-D. Plumbley, "Single-channel audio source separation using deep neural network ensembles," in *Audio Engineering Society Convention 140*, May 2016.

[13] S.-I. Mimilakis, K. Drossos, G. Schuller, and T. Virtanen, "A recurrent encoder-decoder approach with skip-filtering connections for monaural singing voice separation," in *27th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2017.

[14] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[15] Y. Bengio, L. Yao, G. Alain, and P. Vincent, "Generalized denoising auto-encoders as generative models," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., pp. 899–907. Curran Associates, Inc., 2013.

[16] S. Uhlich, F. Giron, and Y. Mitsufuji, "Deep neural network based instrument extraction from music," in *40th International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, 2015, pp. 2135–2139.

[17] S.-I. Mimilakis, E. Cano, J. Abeßer, and G. Schuller, "New sonorities for jazz recordings: Separation and mixing using deep neural networks," in *Audio Engineering Society 2nd Workshop on Intelligent Music Production*, 2016.

[18] S. Uhlich, M. Porcu, F. Giron, M. Enenkl, T. Kemp, N. Takahashi, and Y. Mitsufuji, "Improving music source separation based on deep neural networks through data augmentation and network blending," in *42nd International Conference on Acoustics, Speech and Signal Processing (ICASSP 2017)*, 2017, pp. 261–265.

[19] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Joint optimization of masks and deep recurrent neural networks for monaural source separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 12, pp. 2136–2147, Dec. 2015.

[20] N. Takahashi and Y. Mitsufuji, "Multi-scale multi-band densenets for audio source separation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA 2017)*, Oct. 2017.

[21] R.-K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *CoRR*, vol. abs/1505.00387, 2015.

[22] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR)*, 2015.

[23] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, "Singing voice separation with deep U-Net convolutional networks," in *18th International Society for Music Information Retrieval Conferenceng*, Suzhou, China, Oct. 2017.

[24] J. Jo and Y. Bengio, "Measuring the tendency of CNNs to learn surface statistical regularities," *CoRR*, vol. abs/1711.11561, 2017.

[25] J. Särelä and H. Valpola, "Denoising source separation," *J. Mach. Learn. Res.*, vol. 6, pp. 233–272, Dec. 2005.

[26] A. Goyal, A. Sordoni, M.-A. Côté, R.-N. Ke, and Y. Bengio, "Z-Forcing: Training stochastic recurrent networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 6716–6726.

[27] D. Griffin and J. Lim, "Signal estimation from modified short-time Fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, Apr. 1984.

[28] A.-M. Saxe, J.-L. McClelland, and S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," *CoRR*, vol. abs/1312.6120, 2013.

[29] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10)*, 2010, pp. 249–256.

[30] D.-P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[31] R.-M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "Medleydb: A multitrack dataset for annotation-intensive MIR research," in *15th International Society for Music Information Retrieval (ISMIR)*, Oct. 2014, pp. 66–70.

[32] P. Chandna, M. Miron, J. Janer, and E. Gómez, "Monoaural audio source separation using deep convolutional neural networks," in *Latent Variable Analysis and Signal Separation: 13th International Conference, LVA/ICA 2017*, 2017, pp. 258–266.

[33] F.-R. Stöter, A. Liutkus, R. Badeau, B. Edler, and P. Magron, "Common fate model for unison source separation," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP 2016)*, 2016, pp. 126–130.

[34] I.-Y. Jeong and K. Lee, "Singing voice separation using rpca with weighted l_1-norm," in *Latent Variable Analysis and Signal Separation: 13th International Conference, LVA/ICA 2017*, 2017, pp. 553–562.