The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Deep Representation-Decoupling Neural Networks for Monaural Music Mixture Separation

Zhuo Li,[1] Hongwei Wang,[2] Miao Zhao,[1] Wenjie Li,[1] Minyi Guo[2]
[1]The Hong Kong Polytechnic University, {cszhuoli, csmiaozhao, cswjli}@comp.polyu.edu.hk
[2]Shanghai Jiao Tong University, wanghongwei55@gmail.com, myguo@sjtu.edu.cn

Abstract

Monaural source separation (MSS) aims to extract and reconstruct different sources from a single-channel mixture, which could facilitate a variety of applications such as chord recognition, pitch estimation and automatic transcription. In this paper, we study the problem of separating vocals and instruments from monaural music mixture. Existing works for monaural source separation either utilize linear and shallow models (e.g., non-negative matrix factorization), or do not explicitly address the coupling and tangling of multiple sources in original input signals, hence they do not perform satisfactorily in real-world scenarios. To overcome the above limitations, we propose a novel end-to-end framework for monaural music mixture separation called *Deep Representation-Decoupling Neural Networks* (DRDNN). DRDNN takes advantages of both traditional signal processing methods and popular deep learning models. For each input of music mixture, DRDNN converts it to a two-dimensional time-frequency spectrogram using short-time Fourier transform (STFT), followed by stacked convolutional neural networks (CNN) layers and long-short term memory (LSTM) layers to extract more condensed features. Afterwards, DRDNN utilizes a decoupling component, which consists of a group of multi-layer perceptrons (MLP), to decouple the features further into different separated sources. The design of decoupling component in DRDNN produces purified single-source signals for subsequent full-size restoration, and can significantly improve the performance of final separation. Through extensive experiments on real-world dataset, we prove that DRDNN outperforms state-of-the-art baselines in the task of monaural music mixture separation and reconstruction.

Introduction

The problem of *monaural source separation* (MSS) generally refers to, given a mixture consisting of a series of acoustical signals, how to separate and recover the original sources from the combined signals. MSS is capable to extract the audio signals of interest from background noises and facilitate a wide range of real-world applications, for example, automatic recording and transcription of music and poly-instruments (Dong and Li 2015), automatic chord recognition (ACR) (Cheng et al. 2008), and pitch estimation (Signol, Barras, and Lienard 2008; Emiya, Badeau, and David 2010). Figure 1 gives an illustrative example of utilizing MSS to separate distinct audio signals (i.e., vocal, drum, bass and other sources) from a mixture, which is similar to the cocktail party problem that recognizes what one person is saying when others are speaking at the same time (McDermott 2009).

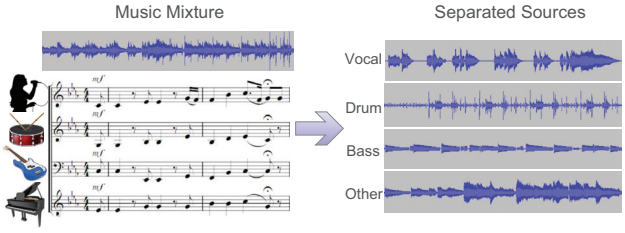


Figure 1: Illustration of monaural source separation for music mixture.

Generally speaking, MSS is a challenging problem since only one single channel is available. A common approach for MSS is to first convert the monaural music mixture into the form of spectrogram by signal processing transforms, then factorize the spectral representation by assigning each time-frequency element of the mixed spectrogram a specific source (Grais, Sen, and Erdogan 2013; Huang et al. 2014a). Conventional methods for spectrogram factorization are typically based on linear matrix factorization methods, for example, non-negative matrix factorization (NMF). NMF approximates a non-negative data matrix X by the product of non-negative basis matrices W and H, i.e., $X \approx WH$ (Ozerov and Fevotte 2010). Though NMF is effective in learning hidden representations, it still shows limitations in the field of MSS, because music mixture is generally complex signal with non-linear patterns, which cannot be well handled by such shallow representation learning models due to its linear structure. Moreover, the coupling and tangling of multiple signals in music mixture make it even harder for traditional linear methods to achieve satisfactory results for MSS.

Instead of directly utilizing linear models for spectrogram separation, some tentative efforts have considered applying non-linear mappings from the mixed spectrogram to multiple sources in MSS. Recently, deep learning based models are widely used in audio and music processing, which achieve great success due to their stacked non-linear

structures. For example, (Nugraha, Liutkus, and Vincent 2016) proposes a framework where deep neural networks (DNNs) are used to model the source spectra and combined with the classical multichannel Gaussian model to exploit the spatial information. (Huang et al. 2014a; 2014b; 2015) use deep recurrent neural networks (RNN) to separate monaural speech signal from mixtures by jointly optimizing an additional time-frequency masking layer to enforce reconstruction constraint. Moreover, by treating the spectrogram of input mixture as a 2D image, (Han and Lee 2016; Chandna et al. 2017; Han, Kim, and Lee 2017) propose to apply deep convolutional neural networks (CNN) for audio separation by passing the spectrogram through multiple convolutional and pooling layers alternately to learn the characteristics of different acoustics.

Though the aforementioned works utilize deep learning methods and apply non-linear mapping structures to address MSS problem, none of them finely considers the entangling nature of music mixture. In fact, to explicitly model the decoupling procedure in MSS is highly desirable, as it could effectively distill the "active ingredients" from original mixture for each specific component and strengthen the discrimination among different sources on the dense feature level. Therefore, in this paper, we propose an end-to-end *Deep Representation-Decoupling Neural Networks* (DRDNN) for monaural music mixture separation. The key idea in DRDNN is to jointly take advantages of traditional signal processing methods, deep learning approaches, and decoupling operation in a comprehensive manner. Specifically, DRDNN consists of five components, namely time-frequency conversion component, feature extraction component, decoupling component, feature restoration component, and frequency-time conversion component. End-to-end training is performed on DRDNN, with the purpose to minimize the overall reconstruction errors across all separated sources.

To evaluate the performance of DRDNN, we conduct extensive experiments on DSD100 dataset[1], which consists of music mixture of vocals, bass, drums and other instruments, as well as the corresponding separated audios of target sources. The experiment results demonstrate the significant improvements of DRDNN over multi-layer perceptron model (MLP), deep CNN model (DCNN), and deep RNN model (DRNN) by 1.6 dB, 0.9 dB and 0.5 dB on average on source-to-interference ratio (SIR), source-to-distortion ratio (SDR) and source-to-artifact ratio (SAR), respectively. We attribute the superiority of DRDNN to its well-designed deep architecture, especially the decoupling component that explicitly distinguishes different sources in original input.

Problem Formulation

Denote the input sequence of monaural music mixture as $x = \{x(1), ..., x(N)\}$ with sampling rate F_s, where $x(n) \in \mathbb{R}$ and $n = 1, ..., N$. The original sequence x is mixed by S different sources, including the target source sequence y, the sequences of background sources y'_s $(s = 1, ..., S-1)$, and the noise sequence e, i.e.,

$$x(n) = y(n) + \sum_{s=1}^{S-1} y'_s(n) + e(n),\ n = 1, ..., N. \quad (1)$$

Our target is to separate sequences of multiple sources from x and make estimation $\tilde{y}$ that approximates the target source sequence y as much as possible.

Deep Representation-Decoupling Neural Networks (DRDNN) Framework

In this section, we introduce the framework of DRDNN in details. Figure 2 shows the overall architecture of DRDNN, which consists of five components and each of which is described as follows: (1) Time-frequency conversion: Short-time Fourier transform (STFT) is used to convert the original input of music mixture into a two-dimensional time-frequency spectrogram representation. (2) Feature extraction: Multiple convolutional neural network (CNN) layers and long-short term memory (LSTM) are employed to extract features from the spectrogram of input mixture and help reduce the feature sizes with respect to frequency and time dimensionality. (3) Decoupling: Right next to feature extraction, a decoupling component is specifically devised in DRDNN, which contains a series of multi-layer perceptrons (MLP) with non-linear activation function. The purpose of adding decoupling component is to enhance the dense feature representation and enforce the insulation between different sources, which can be proven to greatly improve the performance of final separation. (4) Feature restoration: Upon decoupling completes, the decoupled dense features of different sources are handled by another groups of LTSM and deconvolution layers to restore the full-size spectrogram representation for each target source. (5) Frequency-time conversion: Finally, by inverse STFT (ISTFT), the full-size spectrogram representation of each source can be recovered to time-domain signal as the ultimate output.

Time-Frequency Conversion

In signal processing, it is common to convert the time-sequential signal into frequency domain, since the processing in frequency domain can facilitate the operations of denoising, signal detection or feature extraction that are difficult to complete in original time domain. In DRDNN, we choose STFT for time-frequency conversion, as it considers both time and frequency resolution (Muller et al. 2011). Formally,

$$X'(k,t) = \sum_{n=1}^{N} w(n)x(n+tH)e^{-j2\pi kn/N}, \quad (2)$$

where $w(n)$ is the selected window, x is the input sequence, H is the hop-size, $t \in \{1, 2, ..., T\}$ and $k \in \{1, ..., K\}$ indicate the index of time frame and frequency band, respectively. STFT is performed window by window along the timeline, where two consecutive windows are overlapped by hop-size H. The outputs for all the windows are overlap-added in sequential order to produce a 2-D spectrogram X'. Therefore, STFT can describe the changes of local frequency information over time, which is superior to the simple Fourier transform.

[1] https://sisec.inria.fr/home/2016-professionally-produced-music-recordings/

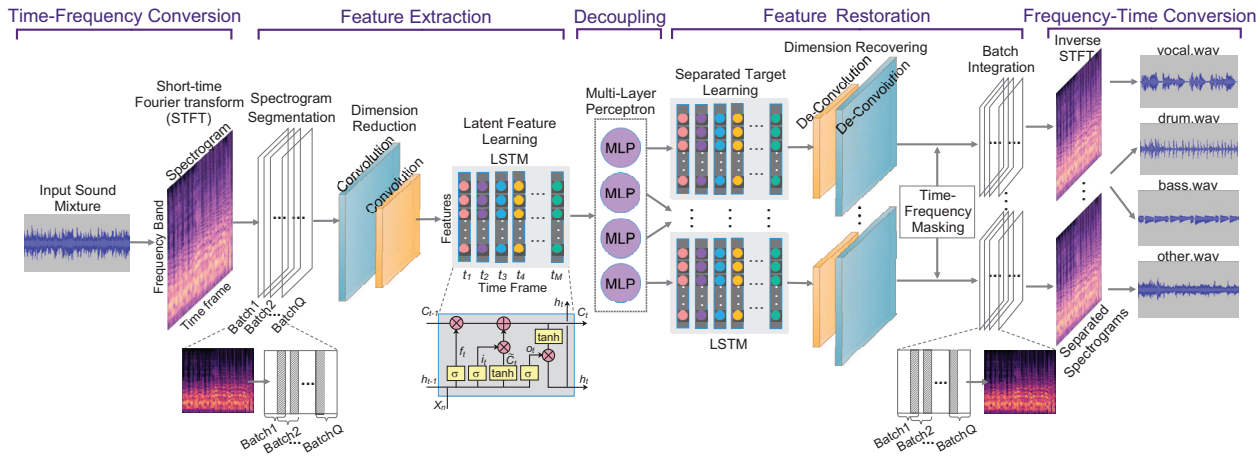


Figure 2: Illustration of architecture of Deep Representation-Decoupling Neural Networks (DRDNN).

The obtained spectrogram X' is a two-dimensional time-frequency representation, in which the time-axis t and the frequency-axis k indicate time frame and frequency band, respectively. Each element in X' represents the corresponding amplitude of the time-frequency coordinates. The spectrogram X' contains the information of harmonics[2] from vocal and various musical instruments. Besides, the distinctive timbres[3] with different playing styles can be well preserved in every local patches of the spectrogram (Han, Kim, and Lee 2017).

Feature Extraction

In feature extraction, the whole spectrogram of input mixture X' needs to be divided into a series of overlapped spectrogram batches first. Afterwards, they are processed by CNN layers for feature dimension reduction. The reduced feature maps output from CNNs are further processed by LSTM to extract the time-dependent feature representation.

Dimension Reduction on Frequency Bands Considering the original spectrogram X' is relatively large, in DRDNN, we segment X' into small batches along time axis t, with two consecutive batches partially overlapped. In this way, the whole spectrogram is reorganized as a pile of batches with the size of $Q \times K \times M$, where Q is the number of total batches, K and M are the number of frequency bands and time frames within a batch, respectively. Remind that the original spectrogram X' has T time frames. If we assume the length of overlap between two consecutive batches is P time frames, then Q can be calculated as $Q = \lfloor T/(M-P) \rfloor$, where $\lfloor \cdot \rfloor$ is the floor operation. We use X to denote the stacked 3-D spectrogram, X_n to denote the n-th batch in the stacked 3-D spectrogram, and $X_{n,k,t}$ to denote the element of the k-th frequency band and the t-th time frame in the n-th batch of X.

[2]Harmonics: when a single note is played by an instrument, the sound is in fact the composition of multiple frequencies, where the fundamental frequency of a note f_0 is the lowest frequency harmonic.

[3]Timbre: the character of a musical sound that is distinct from its pitch and intensity.

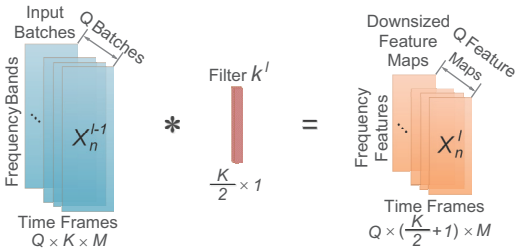


Figure 3: Convolution for frequency-dimension reduction.

Note that when computing STFT, if the FFT size of each window is N' (N'-point FFT), the number of frequency bands of the spectrogram will be $N' + 1$. To avoid parameter explosion while capturing the structural information of the two-dimensional input, we use CNN to reduce the size of frequency band before further processing, because CNN is proven to show advantages in parameter reduction and feature learning in image processing due to its stacked and alternated structure of convolutional layers and pooling layers (LeCun et al. 1998; Alex, Sutskever, and Hinton 2012). In DRDNN, we use a variant of CNN, which consists of multiple convolutional layers and activation layers without pooling layers. This is due to the consideration that either max-pooling or average-pooling leads to information loss. In CNN component, the n-th batch in the l-th convolution layer, denoted by X_n^l, can be computed as the convolution of the n-th batch in the $(l-1)$-th layer X_n^{l-1} with filter k^l:

$$\begin{aligned} X_n^1 &= f(X_n * k^1 + b^1), \\ X_n^l &= f(X_n^{l-1} * k^l + b^l),\ l = 2, ..., L_c, \end{aligned} \tag{3}$$

where $*$ represents the convolution operator, L_c is the number of convolutional layers, f is the activation function, and b^l is the bias parameter.

We discuss the effect of feature dimension reduction by CNN as follows. As illustrated in Figure 3, in DRDNN, the filter is with size of $\frac{K}{2} \times 1$ and strides of $(1, 1)$, where K is the

number of frequency bands. That is, the filter convolves with a half of the total frequency bands, but only spans across one time frame. Due to such rectangular shape filters, the frequency features are fused and reduced in dimension, but the number of time frames remains the same. For each batch, the feature dimension is approximately downsized by 1/2 after passing through a convolution layer.

Feature Extraction across Time Frames After convolution, a pile of batches X_n's are converted into the same number of reduced size feature maps $X_n^{L_c}$'s with decreasing in frequency dimension. For each feature map, though the dimension of frequency is reduced, the dimension of time remains the same as the original batch. Therefore, to capture the high-level correlations over different time frames, we use LSTM to further process the compressed feature maps, which is a variant of recurrent neural networks (RNNs) (Hochreiter and Schmidhuber 1997). LSTM specializes in addressing the long time dependency of features due to the structure of special designed hidden units, i.e., memory cells, which acts like gated leaky units controlling when and how much extent to forget the previous state (Jozefowicz, Zaremba, and Sutskever 2015).

Specifically, for the n-th input feature map $X_n^{L_c}$, we define the LSTM units at time t ($t = 1, ..., M$) as a forget gate $f_{n,t}$, an input gate $i_{n,t}$, an output gate $o_{n,t}$, a memory cell $c_{n,t}$ and a hidden state $h_{n,t}$, where $f_{n,t}, i_{n,t}, o_{n,t} \in [0, 1]$. Then LSTM transition equations are the following:

$$\begin{cases} i_{n,t} = \sigma(W_{X_i} X_{n,t}^{L_c} + W_{h_i} h_{n,t-1} + b_i), \\ f_{n,t} = \sigma(W_{X_f} X_{n,t}^{L_c} + W_{h_f} h_{n,t-1} + b_f), \\ o_{n,t} = \sigma(W_{X_o} X_{n,t}^{L_c} + W_{h_o} h_{n,t-1} + b_o), \\ \widetilde{C}_{n,t} = \tanh(W_{X_c} X_{n,t}^{L_c} + W_{h_c} h_{n,t-1} + b_c), \\ C_{n,t} = f_{n,t} \otimes C_{n,t-1} + i_{n,t} \otimes \widetilde{C}_{n,t}, \\ h_{n,t} = o_{n,t} \otimes \tanh(C_{n,t}). \end{cases} \tag{4}$$

where $X_{n,t}^{L_c}$ is the t-th strip of $X_n^{L_c}$ along the time-frame axis, $\sigma(x) = \frac{1}{1+exp(-x)}$ is the sigmoid function and $\otimes$ denotes element-wise multiplication. In DRDNN, we adopt a bi-directional LSTM to fuse the features of time, and use the hidden state in the last time frame, i.e., $h_{n,M}^{1^{st}-level}, n = 1, ..., Q$, as the final outputs of 1^{st}-level LSTM network.

Decoupling

To enhance the separating ability of DRDNN and provide effective guidances for subsequent processing, we design a decoupling component in DRDNN after the feature extraction component. Decoupling component consists of a group of multi-layer perceptrons (MLP) with non-linear activation function, each of which corresponds to a specific source at the output end. Such multi-layer structures enable the DRDNN to capture complex and highly non-linear relationships between inputs and outputs.

We discuss the decoupling component in detail as follows. After feature extraction component, we obtain the latent representations of the music mixture, i.e., $h_{n,M}^{1^{st}-level}, n = 1, ..., Q$. To decouple the above latent representations, for each feature map with index n, we design S MLPs corresponding to the S sources that we aim to separate from the original input mixture. Formally, the hidden representation for the n-th feature map in the l-th layer MLP targeting the s-th source.

$$\begin{aligned} Z_{n,s}^1 &= f\big(\langle W_{n,s}^1, h_{n,M}^{1^{st}-level} \rangle + b^1\big), \\ Z_{n,s}^l &= f\big(\langle W_{n,s}^l, Z_{n,j}^{l-1} \rangle + b^l\big),\ l = 2, ..., L_m, \end{aligned} \tag{5}$$

where W and b are weight and bias parameters, $\langle \cdot, \cdot \rangle$ denotes the inner product, L_m is the number of layers in MLP, and f is a non-linear activation function.

Previous studies (Hinton et al. 2012) reveal the effectiveness of MLP in acoustic signal modeling. Compared with directly separating the time-feature representation of music mixture into different sources, adding a decoupling layer can greatly strengthen separating ability of DRDNN and enforce the discrimination among different sources. We will further justify the design of decoupling component and demonstrate the achieved performance gains by conducting comparison test in the experiment part.

Feature Restoration

After decoupling, the entangled features of different sources are separated, and we would like to recover the spectrograms for each source. Since we utilize convolutional layer and LSTM to extract the condensed features from the input spectrogram, we need to perform the reverse operation of convolution and LSTM respectively for spectrograms restoration. Specifically, we first use another group of LSTM to restore the two-dimensional time-feature relation from the condensed feature representation of MLPs, and the time-expanded feature representation output from each LSTM is further enlarged by deconvolution to restore the original size feature map.

Similar to the 1^{st}-level LSTM network, the 2^{nd}-level LSTM in feature restoration takes the output of decoupling layer $Z_{n,s}^{L_m}$ (L_m is the number of layers in decoupling) as the input, and the corresponding output of the n-th feature map for the s-th source can be denoted as $h_{n,M,s}^{2^{nd}-level}$.

Contrary to convolution, deconvolution maps a single input activation to multiple outputs (Noh, Hong, and Han 2015), which is realized by a convolution-like operation by multiplying with a filter. Formally, for the l-th layer of deconvolution, the restored feature map $\widehat{Y}_{n,s}^l$ is obtained by convolving the $(l-1)$-th feature map $\widehat{Y}_{n,s}^{l-1}$ with filter g^l (Zeiler et al. 2010):

$$\begin{aligned} \widehat{Y}_{n,s}^1 &= f(h_{n,M,s}^{2^{nd}-level} * g^1 + b^1), \\ \widehat{Y}_{n,s}^l &= f(\widehat{Y}_{n,s}^{l-1} * g^l + b^l),\ l = 2, ..., L_c, \end{aligned} \tag{6}$$

where g^l is the counterpart filter of k^{L_c+1-l} in the (L_c+1-l)-th convolution layer (L_c is the total number of convolution or deconvolution layers), and b^l is the bias parameter.

After restoration, we obtain the output of deconvolution in the last layer $\widehat{Y}_{n,s}^{L_c}$, which, hopefully, could be used to train the whole DRDNN:

$$\mathcal{L} = \tfrac{1}{2} ||\widehat{Y}_{n,s}^{L_c} - Y_{n,s}||_2^2, \tag{7}$$

where $Y_{n,s}$ is the ground truth, i.e., the real spectrogram of the n-th feature map for the s-th source. However, we find that optimizing Eq. (7) is not satisfactory in practice, because the magnitude of separated signals learned by Eq. (7) cannot well match the ground truth in real scenarios. Being aware of this, we alternatively use soft time-frequency masking to predict the spectrogram of separated sources.

Soft Time-Frequency Masking In DRDNN, soft time-frequency masking is used to estimate the proportion of a specific source in the original mixture. Different from learning separated spectrograms directly, soft masking enforces a constraint that the sum of all prediction results equals the original mixture (Huang et al. 2014b). This can effectively reduce the artifacts introduced during source separation. Each element in the soft mask for the s-th source is calculated as

$$M_s(n,k,t) = \frac{|\widehat{Y}^{L_c}_{n,s}(k,t)|}{\sum_{i=1}^{S} |\widehat{Y}^{L_c}_{n,i}(k,t)|}, \tag{8}$$

where $\widehat{Y}^{L_c}_{n,s}(k,t)$ is the value of index (k,t) in n-th restored full size feature map for source s (Huang et al. 2014a). Once the soft time-frequency masks are obtained, the estimation of reconstructed spectrogram of the s-th source $\widetilde{Y}_s$ is computed as

$$\widetilde{Y}_s(n,k,t) = M_s(n,k,t) \otimes X_{n,k,t}, \tag{9}$$

where $\otimes$ is the element-wise multiplication operator and $X_{n,k,t}$ is the element of the n-th batch, the k-th frequency band and the t-th time frame in spectrogram (Huang et al. 2014a). Batch integration is performed to form the complete spectrogram $\widetilde{Y}_s$ by overlap-adding all $\widetilde{Y}_s(n)$ in chronical order ($n = 1, 2, \cdots, Q$, and Q is the total number of batches). The audio signal $\tilde{y}_s$ in time domain can be reconstructed from the estimated spectra $\widetilde{Y}_s$ based on inverse short-time Fourier transform (ISTFT).

Learning Objectives

In DRDNN, we propose a jointly end-to-end supervised training method, in which only one model is required to be optimized for all different sources at the same time. The goal of joint optimization is to minimize objective functions, such as mean squared error (MSE).

Given $\widetilde{Y}_s$ and Y_s, the estimation and the ground truth of the spectrogram of the s-th source, respectively. Thus the learning objectives with respect to MSE can be described as

$$J_{MSE} = \sum_{s=1}^{S} ||\widetilde{Y}_s - Y_s||_2^2, \tag{10}$$

However, one of the goals in source separation is to improve the signal-to-noise (SIR) ratio. In music mixture source extraction, all the instruments other than the desired target source are regarded as background noise. Hence, we do not only need to minimize the difference between the estimated spectrogram and the ground truth for the same source, but also maximize the difference between different sources. Hence the discriminative learning objective (DIS) (Huang et al. 2015) is introduced to enhance the SIR, i.e.,

$$J_{DIS} = \sum_{s=1}^{S} ||\widetilde{Y}_s - Y_s||_2^2 - \beta \sum_{i,j=1,i\neq j}^{S} ||\widetilde{Y}_i - Y_j||_2^2, \tag{11}$$

where β is a weighting parameter between 0 and 1.

Experiments

In this section, we present the details of experiments on DRDNN. We first introduce the dataset and evaluation metrics in experiments, and show the results of DRDNN as well as baselines for mixture separation.

Dataset

We use the public *Demixing Secrets Dataset 100* (DSD100) to evaluate our DRDNN model. DSD100 consists of 100 full-track songs of different genres, which stems from the *"Mixing Secrets" Free Multitrack Download Library*. This dataset is publicly released on the purpose of helping researchers and scholars to evaluate their source separation methods from music recordings. For each song in the dataset, it includes professional synthesized mixtures and the original sources for four tracks, namely vocals, drums, bass and other instruments.

Evaluation Metrics

In our experiments, the evaluation metrics include source to interference ratio (SIR), source to distortion ratio (SDR) and source to artifacts ratio (SAR). SIR indicates the suppression of interference, SAR reflects the artifacts introduced by the separation and SDR evaluates the overall performance. According to the blind source separation (BSS) Eval toolbox (Vincent, Gribonval, and Fevotte 2006), the evaluation metrics can be computed as

$$\begin{cases} \text{SIR} = 10\log(||S_{tar}||_2^2/||e_i||_2^2), \\ \text{SDR} = 10\log(||S_{tar}||_2^2/(||e_i + e_n + e_a||_2^2)), \\ \text{SAR} = 10\log((||S_{tar} + e_i + e_n||_2^2)/||e_a||_2^2), \end{cases} \tag{12}$$

where S_{tar} is the target separated source and e_i, e_n, e_a are the corresponding interference, noise and artifacts. SIR, SDR and SAR are all measured by dB. Higher values of SIR, SDR and SAR means better separation quality.

Experiment Setup

In the experiment, we select Blackman-Harris window to perform STFT, which has large side lobes compared to other common windows. The window size is set to 1024 samples. For each window, we use the 1024-point fast fourier transform (1024-point FFT), and the hop-size H is set to 256. In spectrogram segmentation, the whole spectrogram are split into batches, and the length of each batch is 50 time frames, with 30% overlaps between two consecutive batches.

The dataset is partitioned into training set and test set with $9:1$ ratio. We use leave-one-out policy in experiments, i.e., in each round of training, 10 sound mixtures are selected for test and the reaming 90 sound mixtures are for training, while in another round, 10 other audio mixtures are selected as the test set. The SIR, SDR and SAR are computed by averaging the values over all rounds of training.

Table 1: Comparison of SIR, SDR and SAR for different separated sources between DRDNN and baselines.

		MLP	DCNN	DRNN	DRDNN
Vocal	SIR	6.46	7.50	8.34	**9.39**
	SDR	1.62	2.05	3.02	**3.15**
	SAR	6.09	6.88	6.96	**7.66**
Bass	SIR	2.56	**3.26**	2.88	3.14
	SDR	1.33	2.25	2.81	**3.41**
	SAR	5.72	6.48	6.78	**7.17**
Drum	SIR	5.76	6.81	6.84	**7.35**
	SDR	2.12	2.18	**3.27**	3.23
	SAR	4.97	5.98	**6.62**	6.59
Other	SIR	1.17	1.38	1.37	**1.97**
	SDR	1.57	1.24	1.77	**1.87**
	SAR	3.49	5.05	5.64	**6.81**

Results

In experiments, we use the following methods as baselines:

- **MLP**: Multi-layer perceptron neural network model.
- **DCNN**: Deep convolutional neural network model.
- **DRNN**: Deep recurrent neural network model.

Case Study of Spectrograms and Latent Feature Representation We conduct a case study to gain intuitive understanding on the spectrograms and latent representations of the input mixture and separated sources, respectively

- The spectrograms are visualized in Figure 4. As shown in Figure 4a, the spectrogram of input music mixture consists of all frequency components across time frames. In Figure 4b, the spectrogram of separated vocal reserves characteristics of human voice spanning over wide range frequency bands starting from about 20 Hz up to 8000 Hz, and intermittent pauses can be observed clearly. In Figure 4c, it is evident that the spectrogram of bass mainly distributes over the low frequency band (below 256 Hz), which implies that the sound of bass is quite deep. Figure 4d describes the timbre of drums, in which the regular vertical lines indicate a strong sense of rhythm. The spectrogram of "other" source illustrated in Figure 4e is usually from piano, violin and other instruments, which mainly exists in the low and middle frequency bands.
- The visualization of latent feature representations in different stages of feature learning process is presented in Figure 5. We randomly pick a reorganized mixed batch as input shown in Figure 5a. In Figure 5b, we can see that the feature dimension is reduced and features are vertically merged after convolution. Figure 5c illustrates the time-feature representation learned by LSTM before decoupling, in which features are further fused horizontally across time axis. Figure 5d and 5e represent the condensed feature representations after decoupling and restored feature maps after deconvolution of different sources, respectively.

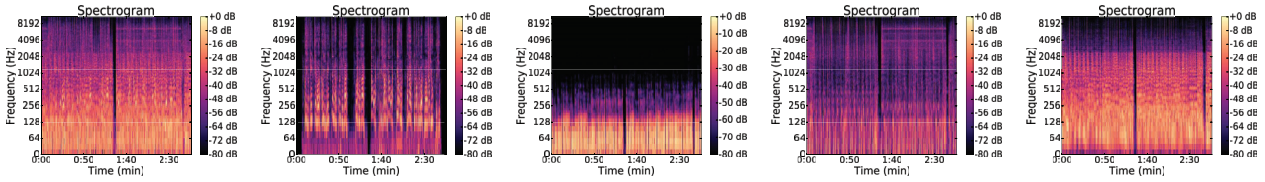
Comparison of Models We compare DRDNN and baselines with respect to the SIR, SDR and SAR of all sources in Table 1, from which we have the following observations:

- In general, among the results in Table 1, DRDNN achieves the best performance compared with all baselines, which strongly proves the effectiveness of our proposed DRDNN model. In addition, the performance ranking for baselines is DRNN $>$ DCNN $>$ MLP.
- SIR for bass is relatively low for all models because: (1) The sound of bass is inherently low and deep so that it is submerged in background instruments; (2) Bass normally serves as accompaniment in play, which is easily submerged in other instruments.
- All models perform unsatisfactorily in separating "other" component. Our explanation is that the "other" source is produced by some accompaniment of musical instruments such as piano and violin, and its ingredients may vary in different music mixtures. For example, the component of "other" could be piano for one mixture and flute for another. Hence, there is no explicit pattern in "other", and it's difficult to capture the characteristics using a small training set with only 100 music mixtures.
- The result of drum is higher than bass and "other" sources by a large margin, since the drum's timbre shows the pattern of strong rhythm (regular vertical lines demonstrated in Figure 4d), and such obvious characteristics can be easily learned by DRDNN and baselines.

Comparison of DRDNN and Its Variants To validate the efficacy of the decoupling and feature extraction component, we compare DRDNN with its two variants: DRDNN without decoupling (DRDNN-wDCP) and DRDNN without dimension reduction (DRDNN-wDR). DRDNN-wDCP drops the decoupling layer and connects the feature extraction part and restoration part directly. DRDNN-wDR removes the convolution and deconvolution layers.

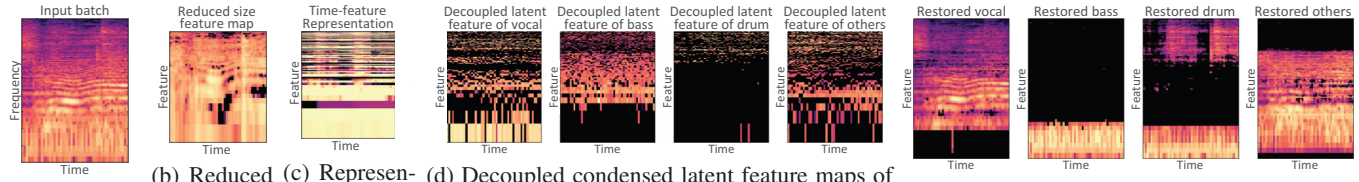
Specifically, the comparison results of SIR and training time is plotted in Figure 6, where the blue bar is the achieved SIR for separated sources, and the red bar is the training time for 3,000 iterations on DSD100 dataset:

- DRDNN in the middle achieves performance on par with DRDNN-wDR on the right, which proves the effectiveness of the feature extraction component. In fact, DRDNN not only achieves well-content performance, but also reduce the training time by around $1/4$ revealed by the red bars. This is due to processing features with much smaller dimensions by DRDNN, which could greatly accelerate the learning procedure and reduce the model complexity.
- DRDNN in the middle outperforms DRDNN-wDCP by a large margin on the left. For example, the decoupling component achieves about 3 dB SIR gain for vocal source at the cost of only 28% additional time overhead. In addition, from Figure 7 that compares the training loss between DRDNN and DRDNN-wDCP, we can observe that DRDNN-wDCP suffers underfitting that the training loss stops at around 0.6 in terms of MSE after 500 iterations. But DRDNN can further converge to and remain at the level of 0.003 after 3000 iterations.



(a) Input music mixture (b) Separated vocal (c) Separated bass (d) Separated drum (e) Separated others

Figure 4: The visualized spectrograms of the input music mixture and the separated spectrograms of different sources.



(a) Input batch (b) Reduced feature map (c) Representation before decoupling (d) Decoupled condensed latent feature maps of different sources (e) Restored feature maps of different sources

Figure 5: The visualized latent feature representation of different stages in feature learning process.

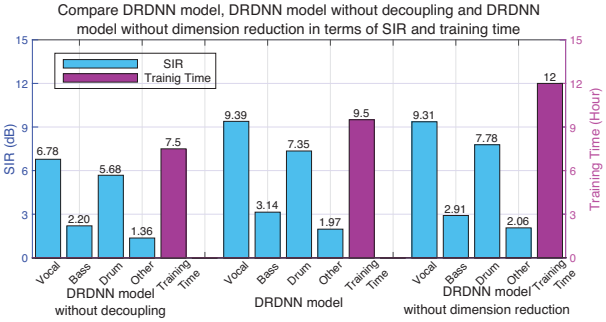


Figure 6: Comparison of DRDNN model and its variants.

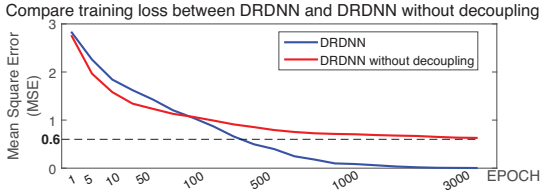


Figure 7: Comparison of training loss between DRDNN and DRDNN without decoupling.

Impact of Soft Time-Frequency Masking Furthermore, we compare the performance of DRDNN with time-frequency masking (DRDNN-Mask) and without time-frequency masking (DRDNN-noMask) and plot the results in Figure 8. Remember that soft time-frequency masking assigns each element from the mixed spectrogram different sources proportionally to their weights of magnitude in the predicted spectrograms. In contrast, DRDNN-noMask directly learns the spectrograms according to Eq. (7).

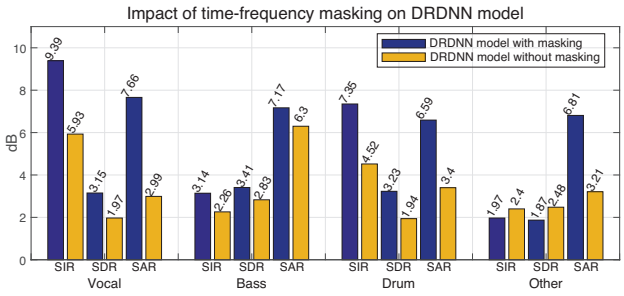


Figure 8: Impact of time-frequency masking.

From Figure 8, we can observe that DRDNN-Mask shows superior performance over DRDNN-noMask. For example, DRDNN-Mask achieves 9.39 dB and 7.35 dB SIR for vocal and drum, respectively, which is 3.46 dB and 2.83 dB higher than DRDNN-noMask. DRDNN-Mask exhibits even more advantages in terms of SAR, achieving 4.67 dB and 3.60 dB SAR gain over DRDNN-noMask for vocal and other sources, respectively. Based on the results in Figure 8, we can reach the following conclusions:

- DRDNN-Mask is more effective and adaptable than DRDNN-noMask, since timbres vary for different music mixtures and the learnt absolute values of the spectrogram for a specific source by DRDNN-noMask may not applicable to other mixtures.

- Time-frequency masking enforces an implicit constraint that the sum of all estimated sources equals to the magnitude of the original spectrogram, which can reduce the artifacts introduced during source separation.

Conclusion

In this paper, we propose Deep Representation-Decoupling Neural Networks(DRDNN), a novel end-to-end framework addressing the monaural source separation (MSS) problem for music mixture. The design of DRDNN takes advantages of both traditional signal processing method as well as popular deep learning models. More importantly, we explicitly introduce a decoupling component to extract effective information for every single source and strengthen the discriminativeness among different sources on dense feature level. Extensive experiments suggests that decoupling component is powerful to disentangle the complex and non-linear structure of music mixture.

References

Alex, K.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc. 1097–1105.

Chandna, P.; Miron, M.; Janer, J.; and Gómez, E. 2017. Monoaural audio source separation using deep convolutional neural networks. In *Proceedings of 13th International Conference on Latent Variable Analysis and Signal Separation*, LVA ICA2017, 258–266.

Cheng, H.-T.; Yang, Y.-H.; Lin, Y.-C.; Liao, I.-B.; and Chen, H. H. 2008. Automatic chord recognition for music classification and retrieval. In *2008 IEEE International Conference on Multimedia and Expo*, 1505–1508. IEEE.

Dong, Y., and Li, D. 2015. *Automatic Speech Recognition: A Deep Learning Approach*. Springer, 2015th edition.

Emiya, V.; Badeau, R.; and David, B. 2010. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Transactions on Audio, Speech, and Language Processing* 18(6):1643–1654.

Grais, E. M.; Sen, M. U.; and Erdogan, H. 2013. Deep neural networks for single channel source separation. In *Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP'13, 3734–3738. IEEE.

Han, Y., and Lee, K. 2016. Acoustic scene classification using convolutional neural network and multiple-width frequency-delta data augmentation. *CoRR* abs/1607.02383.

Han, Y.; Kim, J.; and Lee, K. 2017. Deep convolutional neural networks for predominant instrument recognition in polyphonic music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25(1):208–221.

Hinton, G.; Deng, L.; Yu, D.; Dahl, G. E.; r. Mohamed, A.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T. N.; and Kingsbury, B. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29(6):82–97.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780.

Huang, P. S.; Kim, M.; Hasegawa-Johnson, M.; and Smaragdis, P. 2014a. Deep learning for monaural speech separation. In *Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP'14, 1562–1566.

Huang, P.; Kim, M.; Hasegawa-Johnson, M.; and Smaragdis, P. 2014b. Singing-voice separation from monaural recordings using deep recurrent neural networks. In *Proceedings of the 15th International Society for Music Information Retrieval Conference*, ISMIR'14, 477–482.

Huang, P. S.; Kim, M.; Hasegawa-Johnson, M.; and Smaragdis, P. 2015. Joint optimization of masks and deep recurrent neural networks for monaural source separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23(12):2136–2147.

Jozefowicz, R.; Zaremba, W.; and Sutskever, I. 2015. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning*, ICML'15, 2342–2350. JMLR.org.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

McDermott, J. H. 2009. The cocktail party problem. *Current Biology* 19(22):R1024–R1027.

Muller, M.; Ellis, D. P. W.; Klapuri, A.; and Richard, G. 2011. Signal processing for music analysis. *IEEE Journal of Selected Topics in Signal Processing* 5(6):1088–1110.

Noh, H.; Hong, S.; and Han, B. 2015. Learning deconvolution network for semantic segmentation. In *Proceedings of the 2015 IEEE International Conference on Computer Vision*, ICCV'15, 1520–1528. IEEE Computer Society.

Nugraha, A. A.; Liutkus, A.; and Vincent, E. 2016. Multichannel audio source separation with deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24(9):1652–1664.

Ozerov, A., and Fevotte, C. 2010. Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing* 18(3):550–563.

Signol, F.; Barras, C.; and Lienard, J.-S. 2008. Evaluation of the pitch estimation algorithms in the monopitch and multipitch cases. In *Proceedings of the Acoustical Society of America 08*, 675–680.

Vincent, E.; Gribonval, R.; and Fevotte, C. 2006. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing* 14(4):1462–1469.

Zeiler, M. D.; Krishnan, D.; Taylor, G. W.; and Fergus, R. 2010. Deconvolutional networks. In *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR 2010, 2528–2535. IEEE.