

SEMI-SUPERVISED MONAURAL SINGING VOICE SEPARATION WITH A MASKING NETWORK TRAINED ON SYNTHETIC MIXTURES

Michael Michaelshvili¹, Sagie Benaim¹, Lior Wolf^{1,2}

¹Tel Aviv University

²Facebook AI Research

ABSTRACT

We study the problem of semi-supervised singing voice separation, in which the training data contains a set of samples of mixed music (singing and instrumental) and an unmatched set of instrumental music. Our solution employs a single mapping function g , which, applied to a mixed sample, recovers the underlying instrumental music, and, applied to an instrumental sample, returns the same sample. The network g is trained using purely instrumental samples, as well as on synthetic mixed samples that are created by mixing reconstructed singing voices with random instrumental samples. Our results indicate that we are on a par with or better than fully supervised methods, which are also provided with training samples of unmixed singing voices, and are better than other recent semi-supervised methods.

Index Terms— Singing voice separation, Adversarial training, Semi-supervised learning

1. INTRODUCTION

The problem of separating a given mixed signal into its components without direct supervision is ubiquitous. For example, in single cell gene expression conducted in cancer research, one obtains a gene expression that contains both the cancer cell of interest and the expression of immune cells that attach to it. In what is known in biology as gene expression deconvolution [1], one would like to obtain the expression of the cancer cell itself, while only having access to a dataset of such mixed readings and another dataset containing gene expression profiles of immune cells.

In the task of singing voice separation, which is the focus of this work, examples of mixed music, which contains both singing and instrumental music, are abundant. It is also relatively easy to label parts of the song where no singing is present. However, it is much harder to separate out pure voice samples. Without such samples, one cannot use the supervised methods that were suggested for this separation task.

In this work, we propose a novel method for performing the separation. The method is based on applying a learned function twice: once on the mixtures, in order to recover estimated singing voice samples, and once on synthetic mixes, in which the reconstructed singing samples are crossed with real

instrumental samples from the training set. The advantage of these crosses over the original mixed samples is that the underlying components of these mixed samples are known, and, therefore, added losses can be applied, when training the separating function on them.

2. RELATED WORK

Single-channel source separation is a long-standing task which has been researched extensively. Classical works on blind source separation include Single-Channel ICA [2] and, specifically in singing voice separation, RPCA [3]. These methods utilize hand-crafted priors on the sources, such as a low rank assumption on the instrumental music.

The problem of singing voice separation is often studied in the supervised case, where the mixed samples are provided with the target source. Often, a simple masking model in the spectral domain is assumed and the desired source \mathbf{b} is given by a point-wise multiplication of the mixed signal \mathbf{a} and some mask \mathbf{m} , i.e., $\mathbf{b} = \mathbf{a} \odot \mathbf{m}$, where \odot is the Hadamard product. In our work, we use a network g such that $\mathbf{b} \approx \mathbf{a} - g(\mathbf{a})$, where the architecture of g includes the masking, i.e., $g(\mathbf{a}) = \mathbf{a} \odot m(\mathbf{a})$, for some subnetwork m with outputs in $[0, 1]$.

The GRA3 method [4], similarly to ours, estimates the mask \mathbf{m} directly from the mixed sample \mathbf{a} . This is done using an ensemble of four deep neural networks, trained with different losses. The architecture we use is of the type commonly used for the autoencoding of images. Similar architectures are used in other work to directly estimate all the sources from the mixtures, e.g., [5, 6].

The GRU-RIS-L method of Mimitakis et al. [7], employs RNNs of stochastic depth in order to recover the time-frequency mask. The usage of RNNs allows for efficient modeling of longer time dependencies of the input data. This is extended in [8] (MaDTwinNet) by introducing a technique called Twin Net, which regularizes the RNNs. Our analysis of long sequences is segment by segment and does not exploit long range dependencies.

Adversarial training using GANs [9] is a powerful method for unconditional image generation. GANs are composed of two parts: (i) a generator g that synthesizes realistic images, and (ii) a discriminator d that distinguishes real from fake images. The objective of the generator is to create images that

are realistic enough to fool the discriminator. The objective of the discriminator is to detect the fake images. The method was later extended to perform unsupervised image-to-image mapping [10]. In this setting, the generator is conditioned on an input image from the source domains and generates a “fake” sample in the target domain. As in the unconditional setting, the discriminator attempts to differentiate between real and generated images. Adversarial training was used for supervised source separation, where the distribution of each of the mixture components is known and modeled by a GAN, by Stoller et al. [11] and Subkhan et al. [12]. The adversarial training was motivated as being better able to deal with correlated sources.

In the setting of Semi-supervised audio source separation, in which we work, the task is to separate mixtures of two sources given mixed samples as well as samples from only one of the sources. Previous solutions were typically based on NMF [13] or the related PLCA [14].

The most similar method to ours is NES [15], which separates mixed samples into a sum of two samples: one from an observed domain and one from an unobserved domain. The method consists of an iterative process: (i) estimation of samples from the unobserved distribution; (ii) synthesis of mixed signals by combining training samples from the observed domain and the estimated samples from the unobserved one; (iii) training of a mapping from the mixed domain to the observed domain. It was demonstrated in [15] that due to its iterative nature, NES is sensitive to the initialization method. Our method, in contrast, performs a non-iterative end to end training that includes the synthetic mixtures as part of the network. This also allows us to apply additional losses, such as GAN based losses and the constraint that learned function g is idempotent ($g \circ g = g$) [16]. As can be seen in Sec. 5, our results are significantly stronger than those obtained by [15].

3. METHOD

In the problem of semi-supervised separation, the learning algorithm is provided with unlabeled datasets from two domains, a domain of mixtures A and a domain of observed components C . There also exists a target domain B , from which no samples are presented. The goal is to learn a function $g : A \rightarrow C$, which maps a sample in domain $a \in A$ to a component c in domain C such that there exists a component $b \in B$ for which the following equality holds $a = b + c$.

During training, we obtain two sets of unmatched samples: the set S_A of mixed samples in domain A , and the set S_C of samples in the observed domain C .

Due to the lack of training samples in B , we rely on the generation of a synthetic training set of samples in domain B :

$$\bar{S}_B := \{a - g(a) | a \in S_A\} \quad (1)$$

The network mixes the samples in \bar{S}_B with random samples in C , in order to create the following set of synthetic

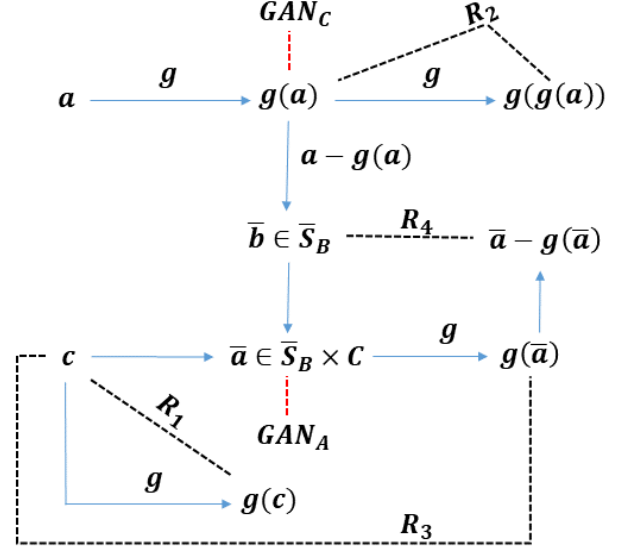


Fig. 1. The transformations and constraints of our method. Blue arrows stand for functions. Dashed lines represent losses, which are of two types: reconstruction losses (black) and GAN loss terms (red).

crosses:

$$\bar{S}_B \times C := \{\bar{b} + c | \bar{b} \in \bar{S}_B, c \in C\} \quad (2)$$

For each sample $\bar{a} \in \bar{S}_B \times C$, we memorize the underlying samples \bar{b}, c that were used to create it, and mark these samples as $b(\bar{a})$ and $c(\bar{a})$, respectively.

In addition to g , we train two discriminator networks d_C and d_A , which provide adversarial signals that enforce the distribution of the recovered samples from domain C to match the distribution of the training set S_C and the mixed synthetic samples to match the distribution of domain S_A . Specifically, d_C is applied to samples of the form $g(a)$, where $a \in S_A$; d_A is applied to samples of the form $\bar{a} \in \bar{S}_B \times C$.

The following losses are used to train the network g :

$$\mathcal{L}_{R_1} = \sum_{c \in S_C} \|g(c) - c\|_1 \quad (3)$$

$$\mathcal{L}_{R_2} = \sum_{a \in S_A} \|g(g(a)) - g(a)\|_1 \quad (4)$$

$$\mathcal{L}_{R_3} = \sum_{\bar{a} \in \bar{S}_B \times C} \|g(\bar{a}) - c(\bar{a})\|_1 \quad (5)$$

$$\mathcal{L}_{R_4} = \sum_{\bar{a} \in \bar{S}_B \times C} \|(\bar{a} - g(\bar{a})) - b(\bar{a})\|_1 \quad (6)$$

$$\mathcal{L}_{GAN_C} = \sum_{a \in S_A} -\ell(d_C(g(a)), 0) \quad (7)$$

$$\mathcal{L}_{GAN_A} = \sum_{\bar{a} \in \bar{S}_B \times C} -\ell(d_A(\bar{a}), 0), \quad (8)$$

where ℓ is the Least Squares loss, following [17]. That is, $\ell(x, y) = (x - y)^2$. Note that g appears in $\mathcal{L}_{\text{GAN}_A}$ and appears more than once in $\mathcal{L}_{R_3}, \mathcal{L}_{R_4}$, since it takes part in the formation of the set $\bar{S}_B \times C$.

The first loss requires that g , applied to samples in C , is the identity operator. The second loss enforces idempotence on g (since g maps to domain C , applying it again should be the same as applying identity), and the next two losses enforce the separation of the synthetic cross samples to result in the known components. The last two losses are GAN based losses in the domains C and A . The full objective for g is defined as:

$$\mathcal{L}_g = \mathcal{L}_{R_1} + \mathcal{L}_{R_1} + \mathcal{L}_{R_3} + \mathcal{L}_{R_4} + 0.5(\mathcal{L}_{\text{GAN}_C} + \mathcal{L}_{\text{GAN}_A})$$

The discriminators of the GAN losses, d_C and d_A , are trained with the following losses, respectively:

$$\mathcal{L}_{d_C} = \sum_{\mathbf{a} \in S_A} \ell(d_C(g(\mathbf{a})), 0) + \sum_{\mathbf{c} \in S_C} \ell(d_C(\mathbf{c}), 1) \quad (9)$$

$$\mathcal{L}_{d_A} = \sum_{\bar{\mathbf{a}} \in \bar{S}_B \times C} \ell(d_A(\bar{\mathbf{a}}), 0) + \sum_{\mathbf{a} \in S_A} \ell(d_A(\mathbf{a}), 1) \quad (10)$$

4. IMPLEMENTATION DETAILS

An Adam optimizer is used with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and a batch size of one. The learning rate is initially set to 0.0001 and is halved after 100,000 iterations.

4.1. Network architecture

The underlying network architecture adapts that used in [18]. Let $C7S1_k$ denote a 7×7 1-stride convolution with k filters. Similarly, let $C4S2_k$ denote a 4×4 2-stride convolution with k filters. Let R_k denote a residual block with two 3×3 convolutional blocks and k filters and let u_k denote a $2 \times$ nearest-neighbor upsampling layer, followed by a 5×5 convolutional block with k filters and 1 stride.

Recall that $g(\mathbf{a}) = \mathbf{a} \odot m(\mathbf{a})$. m is built as an auto-encoder. The encoder consists of two downsampling convolutional layers, $C7S1_{64}$ and $C4S2_{128}$. This is followed by four residual blocks of type R_{256} . Each convolutional layer of the encoder is followed by an Instance Normalization layer and a ReLU activation. The decoder consists of four residual blocks of type R_{256} . This is followed by two upsampling blocks, u_{128} and u_{256} , and a convolutional layer $C7S1_3$. Each convolutional layer of the decoder is followed by a Adaptive Instance Normalization [19] layer and a ReLU activation. To obtain mask values between 0 and 1, the ReLU of the last layer is replaced by a sigmoid activation function.

A multi-scale discriminator is used for d_C and d_A , as in [20], to produce both accurate low-level details, as well as capture global structure. Each discriminator consists of the following sequence of layers: $C4S2_{64}$, $C4S2_{128}$, $C4S2_{256}$ and $C4S2_{512}$. Each convolutional layer is followed by a leaky ReLU with slope parameter of 0.2.

4.2. Audio processing

To convert an audio file to an input to network g , we perform the following pre-processing: The audio file is re-sampled to 20480 Hz. It is then split into clips of duration of 0.825 seconds. We then compute the Short Time Fourier Transform (STFT) with window size of 40ms, hop size of 64 and FFT size of 512, resulting in an STFT of size 257×256 . Lastly, we take the absolute values and apply a power-law compression with $p = 0.3$, i.e. we obtain $|A|^{0.3}$, where $|A|$ is the magnitude of the STFT. The highest frequency bin is trimmed, resulting in an input audio representation of size 256×256 .

To convert the method's output $\bar{\mathbf{b}} = \mathbf{a} - g(\mathbf{a})$ back to audio, we apply ISTFT on the multiplication of the magnitude spectrogram of $\bar{\mathbf{b}}$ with the phase of the original mixture, and add back the top-frequency by padding with zeros. To process an entire audio file, we simply process each non overlapping segment individually, and concatenate the results.

5. EVALUATION

We perform a comparison to other semi-supervised methods, using the evaluation protocol used by [15]. In addition, we compare our semi-supervised method to the state of the art supervised methods, following the protocol used in [8]. Finally, ablation experiments are run to study the relative importance of the various losses.

5.1. Comparison to semi-supervised methods

For semi-supervised methods, our evaluation protocol follows closely the one of [15]. We evaluated our method against the five methods reported there: (1) *Semi-supervised Non-negative Matrix Factorization (NMF)* [14]: The method learns a set of $l = 3$ bases from the samples in S_C by Sparse NMF [21, 22] as $S_C = H_c * W_c$, with mixture components H_c and basis vectors W_c , where the two matrices are non-negative, using the fast Non-negative Least Squares solver of [23]. Then, the mixture S_A is decomposed with $2l$ bases, where the first l bases are simply W_c : $S_A = H_{ac} * W_c + H_{ab} * W_b$. The estimated components from domain B are then given by: $\bar{S}_B = h_{ab} * W_b$. (2) *GAN*: A masking function m is learned so that after masking, the training mixtures are indistinguishable from the source samples by a discriminator d , similar to our $\mathcal{L}_{\text{GAN}_C}$ loss. (3) *GLO Masking (GLOM)*: This method learns an explicit generative GLO [24] model to both domain A and domain C and fits the parameters to each given sample \mathbf{a} , followed by approximating the solution by a mask between 0 and 1 that is multiplied by the mixed signal \mathbf{a} . (4) *Neural Egg Separation (NES)*: The iterative method of [15], which is initialized by taking the mixture components to be each half of the mixed signal \mathbf{a} . (5) *Fine-tuned NES (NES-FT)*: Initializing NES with the GLOM solution above.

The semi-supervised experiments are performed on the MUSDB18 [25] dataset, which consists of 150 music tracks, 100 of which in the train set and 50 in the test set. Each music track is comprised of separate signal streams of the mixture, drums, bass, the accompaniment, and the vocals. In our method, samples are preprocessed as described in Sec. 4.2 and then trained using the method of Sec. 3. We compare the performance of our method, using the signal-to-distortion ratio (SDR) in Tab. 1. We can observe that NMF, GAN, GLOM and NES perform much worse than NES-FT and our method. There is also a significant gap between NES-FT and our method (2.1dB vs 3.2dB) as well.

5.2. Comparison to fully supervised methods

We next compare with fully supervised methods that solely deal with singing voice separation. For this comparison, our evaluation protocol follows closely the one of [8], except that our method does not employ the training samples of the singing voices and is unaware of the matching pairs (\mathbf{a} , \mathbf{c}). Baseline results are shown for *GRA3* [4], *GRU-RIS-L* [7] and *MaDTwinNet* [8], which are discussed in Sec. 2, and for the following methods: *CHA* [6], which uses CNN to estimate time-frequency soft masks; *STO2* [26], which is based on signal representation that divides the complex spectrogram into a grid of patches of arbitrary sizes; and *JEO2* [3], which is based on robust principal component analysis (RPCA). The results for all of the above approaches are obtained from [8].

The development subset of of Demixing Secret Dataset (DSD100) [27] and the non-bleeding/non-instrumental stems of MedleyDB [28] are used for training. Baseline approaches here are trained in a supervised fashion, while our method is trained in a semi-supervised manner. For evaluation, the evaluation subset of DSD100, which consists of 50 samples, is used. For these methods, the literature reports both the signal-to-distortion ratio (SDR) and the signal to-interference ratio (SIR), and we report both, using the `mir_eval` Python library.

The comparison is shown in Tab. 2. As can be seen, SDR values for our method are better than those of *GRA3* and *CHA*, but worse than *STO2*, *JEO2*, *GRU-RIS-L* and *MaDTwinNet*. Our SIR value is significantly higher than all baselines, achieving a gap of 7.0 to the second best method. This is consistent with our observation: The network seems to filter out all the instrumental music very well for most samples. However, for some samples, there is a slight distortion of the voice generated. Samples, in comparison to those published by [7], are available at <https://sagiebenaim.github.io/Singing/>.

5.3. Ablation study

We perform an ablation analysis to understand the relative contribution of the different losses in our method. This is done by removing various losses from the training objective and retraining.

Table 1. Median SDR (dB) for our method and previous semi-supervised approaches evaluated on the MUSDB18 [25] dataset. Baselines are from [15], which did not report SIR.

Approach	SDR	SIR	Approach	SDR	SIR
NMF	0.0	-	NES	0.3	-
GAN	0.3	-	NES-FT	2.1	-
GLOM	0.6	-	Ours	3.2	14.2

Table 2. Median SDR and SIR (dB) values for the proposed method and previous supervised approaches, which solely deal with singing voice separation, evaluated on the evaluation subset of DSD100 [27] dataset.

Approach	Supervision	SDR	SIR
GRA3 [4]	supervised	-1.7	1.3
CHA [6]	supervised	1.6	5.2
STO2 [26]	supervised	3.9	6.7
JEO2 [3]	supervised	4.1	6.1
GRU-RIS-L [7]	supervised	4.2	7.9
MaDTwinNet [8]	supervised	4.6	8.2
Ours	semi-supervised	3.5	15.2

Table 3. Ablation study: Median SDR and SIR values for the proposed method without (w/o) selected losses evaluated on the evaluation subset of DSD100 [27].

Losses	SDR	SIR	Losses	SDR	SIR
All losses	3.5	15.2	w/o \mathcal{L}_{R_4}	-6.3	-4.7
w/o \mathcal{L}_{R_1}	-0.9	3.4	w/o \mathcal{L}_{GAN_A}	-6.3	-4.2
w/o \mathcal{L}_{R_2}	2.3	9.7	w/o \mathcal{L}_{GAN_C}	-4.1	-2.4
w/o \mathcal{L}_{R_3}	-4.3	13.3	w/o $\mathcal{L}_{GAN_A} \& \mathcal{L}_{GAN_C}$	-17.0	-3.6

As can be seen in Tab. 3, \mathcal{L}_{R_2} has a smaller significance than other losses. The most significant losses are \mathcal{L}_{R_4} , and the GAN losses \mathcal{L}_{GAN_C} and \mathcal{L}_{GAN_A} , without even one of these, the two metrics drop considerably. \mathcal{L}_{R_3} is also very significant and without it the SDR is greatly diminished.

6. CONCLUSIONS

We present a new method for semi-supervised singing voice separation that is competitive with some of the state of the art supervised methods and all of the literature semi-supervised ones. The crux of the method is the use of compound losses, applied to synthetic mixes, and the application of two GANs. The method is applied sequentially to fixed-length audio clips and, as future work, we would like to employ overlapping segments and even incorporate longer-term dependencies.

7. REFERENCES

- [1] Yingdong Zhao and Richard Simon, “Gene expression deconvolution in clinical samples,” *Genome medicine*, vol. 2, no. 12, pp. 93, 2010.
- [2] Mike Davies and Christopher James, “Source separation using single channel ICA,” *Signal Processing*, 2007.
- [3] Il-Young Jeong and Kyogu Lee, “Singing voice separation using rpca with weighted l_1 -norm,” in *Int. Conf. on Latent Variable Analysis and Signal Separation*, 2017.
- [4] Emad M Grais, Gerard Roma, Andrew JR Simpson, and Mark D Plumbley, “Single-channel audio source separation using deep neural network ensembles,” in *Audio Engineering Society Convention 140*, 2016.
- [5] S. Uhlich, F. Giron, and Y. Mitsufuji, “Deep neural network based instrument extraction from music,” in *ICASSP*, 2015.
- [6] Pritish Chandna, Marius Miron, Jordi Janer, and Emilia Gómez, “Monoaural audio source separation using deep convolutional neural networks,” in *Int. Conf. on Latent Variable Analysis and Signal Separation*, 2017.
- [7] Stylianos I. Mimilakis, Konstantinos Drossos, João F Santos, Gerald Schuller, Tuomas Virtanen, and Yoshua Bengio, “Monoaural singing voice separation with skip-filtering connections and recurrent inference of time-frequency mask,” in *ICASSP*, 2018.
- [8] Konstantinos Drossos, Stylianos Ioannis Mimilakis, et al., “MaD TwinNet: Masker-denoiser architecture with twin networks for monaural sound source separation,” in *Int. Joint Conf. on Neural Networks*, 2018.
- [9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al., “Generative adversarial nets,” in *NIPS*, 2014.
- [10] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *ICCV*, 2017.
- [11] Daniel Stoller, Sebastian Ewert, and Simon Dixon, “Adversarial semi-supervised audio source separation applied to singing voice extraction,” *arXiv preprint arXiv:1711.00048*, 2017.
- [12] Cem Subakan and Paris Smaragdis, “Generative adversarial source separation,” *arXiv preprint arXiv:1710.10779*, 2017.
- [13] Tom Barker and Tuomas Virtanen, “Semi-supervised non-negative tensor factorisation of modulation spectrograms for monaural speech separation,” in *Int. Joint Conf. Neural Networks*, 2014.
- [14] Paris Smaragdis, Bhiksha Raj, et al., “Supervised and semi-supervised separation of sounds from single-channel mixtures,” in *Int. Conf. Independent Component Analysis and Signal Separation*, 2007.
- [15] Yedid Hoshen, Tavi Halperin, and Ariel Ephrat, “Neural separation of observed and unobserved distributions,” in *Submitted to Int. Conf. Learning Representations*, 2019.
- [16] Tomer Galanti and Lior Wolf, “A theory of output-side unsupervised domain adaptation,” *arXiv preprint arXiv:1703.01606*, 2017.
- [17] Xudong Mao, Qing Li, Haoran Xie, et al., “Least squares generative adversarial networks,” in *Int. Conf. Computer Vision (ICCV)*, 2017.
- [18] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz, “Multimodal unsupervised image-to-image translation,” *arXiv preprint arXiv:1804.04732*, 2018.
- [19] Xun Huang and Serge J. Belongie, “Arbitrary style transfer in real-time with adaptive instance normalization,” *Int. Conf. on Computer Vision (ICCV)*, 2017.
- [20] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, et al., “High-resolution image synthesis and semantic manipulation with conditional GANs,” in *CVPR*, 2018.
- [21] Patrik O Hoyer, “Non-negative matrix factorization with sparseness constraints,” *JMLR*, 2004.
- [22] Hyunsoo Kim and Haesun Park, “Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares,” *Bioinformatics*, 2007.
- [23] Jingu Kim and Haesun Park, “Fast nonnegative matrix factorization: An active-set-like method and comparisons,” *SIAM J. Scientific Computing*, 2011.
- [24] Piotr Bojanowski, Armand Joulin, et al., “Optimizing the latent space of generative networks,” in *ICML*, 2018.
- [25] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner, “The MUSDB18 corpus for music separation,” Dec. 2017.
- [26] Fabian-Robert Stöter, Antoine Liutkus, Roland Badeau, Bernd Edler, and Paul Magron, “Common Fate Model for Unison source Separation,” in *ICASSP*, 2016.
- [27] Antoine Liutkus, Fabian-Robert Stöter, et al., “The 2016 signal separation evaluation campaign,” in *Latent Variable Analysis and Signal Separation*, 2015.
- [28] Rachel Bittner, Justin Salamon, Mike Tierney, et al., “Medleydb: A multitrack dataset for annotation-intensive mir research,” in *ISMIR*, 2014.