Unsupervised Deep Unfolded Representation Learning for Singing Voice Separation

Weitao Yuan, Shengbei Wang, Jianming Wang, Masashi Unoki, *Member, IEEE* and Wenwu Wang, *Senior Member, IEEE*

***Abstract*—Learning effective vocal representations from a waveform mixture is a crucial but challenging task for deep neural network (DNN)-based singing voice separation (SVS). Successful representation learning (RL) depends heavily on well-designed neural architectures and effective general priors. However, DNNs for RL in SVS are mostly built on generic architectures without general priors being systematically considered. To address these issues, we introduce deep unfolding to RL and propose two RL-based models for SVS, deep unfolded representation learning (DURL) and optimal transport DURL (OT-DURL). In both models, we formulate RL as a sequence of optimization problems for signal reconstruction, where three general priors, synthesis, non-negative, and our novel analysis, are incorporated. In DURL and OT-DURL, we take different approaches in penalizing the analysis prior. DURL uses the Euclidean distance as its penalty, while OT-DURL uses a more sophisticated penalty known as the OT distance. We address the optimization problems in DURL and OT-DURL with the first-order operator splitting algorithm and unfold the obtained iterative algorithms to novel encoders, by mapping the synthesis/analysis/non-negative priors to different interpretable sublayers of the encoders. We evaluated these DURL and OT-DURL encoders in the unsupervised informed SVS and supervised Open-Unmix frameworks. Experimental results indicate that (1) the OT-DURL encoder is better than the DURL encoder and (2) both encoders can considerably improve the vocal-signal-separation performance compared with those of the baseline model.**

***Index Terms*—Deep unfolding, representation learning, analysis prior, singing voice separation**

I. Introduction

The field of singing voice separation (SVS) has garnered considerable attention and research interest [1]–[8]. Traditionally, the SVS task has been addressed using model-based methods [2], [9]–[12], which rely heavily on the prior knowledge (assumptions) about the characteristics of the singing voice. In contrast to model-based methods, modern 'deep' methods, e.g., using deep neural networks (DNNs), operate in a data-driven manner. These methods use deep networks of neurons to learn feature representations, which have demonstrated superior effectiveness for SVS compared with traditional methods [3], [5], [8], [13]–[15].

Weitao Yuan and Shengbei Wang are with the Tianjin Key Laboratory of Autonomous Intelligence Technology and Systems, School of Software, Tiangong University, Tianjin, China, e-mail: weitaoyuan@hotmail.com and wangshengbei@tiangong.edu.cn.

Jianming Wang is with the Centre for Engineering Faculty, Tiangong University, Tianjin, China, e-mail: wangjianming@tiangong.edu.cn.

Masashi Unoki is with the School of Information Science, Japan Advanced Institute of Science and Technology, Nomi, Japan, e-mail: unoki@jaist.ac.jp.

Wenwu Wang is with the Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, UK, e-mail: w.wang@surrey.ac.uk.

Fig. 1: Typical DNN-based SVS framework, where $\mathbf{x}$ is input mixture signal and $\hat{\mathbf{x}}_0$ is estimated vocal signal.

An ordinary DNN-based SVS framework typically comprises an encoder, separator, and decoder. As shown in Fig. 1, the encoder transforms a time-domain mixture signal (or a noisy vocal signal) into a latent representation. The separator takes this latent representation as input and produces an estimated representation of the target vocal signal. Finally, the decoder converts the estimated representation back into the time-domain vocal signal. The separator relies heavily on the encoder and latent representation, as its goal is to estimate the representation of the target vocal signal using these components. To the best of our knowledge, while there has been significant research dedicated to designing the separator [2], comparatively less focus has been given to the encoder and latent representation [16].

In SVS systems, encoders can be broadly classified into two types. The first type relies on traditional time-frequency analysis tools such as short-time Fourier transform (STFT), wavelet transform, and similar techniques. The second type uses deep-learning-based approaches, specifically DNN-based representation learning (RL) methods [16], [17]. The latent representations produced using traditional time-frequency tools, which are called pre-computed representations, have only limited expression ability as they are mainly computed with linear transformation [13], [14], [18], [19]. Even more problematic, these analysis tools are neither trainable nor data-driven. This limitation prevents them from adapting to the characteristics of the training data. In contrast, the representations generated by DNN-based RL, known as learned representations, are derived from real data using a nonlinear and data-driven approach. Therefore, these learned representations have the capability to dynamically capture the essential features of vocal signals. Moreover, DNN-based RL has the advantage of directly learning latent representations from the time-domain mixture. This enables the utilization of crucial phase information [20] for separation, which is frequently overlooked with traditional analysis tools [13], [14], [18], [19]. Due to these advantages, DNN-based RL is considered more effective than traditional tools for SVS. However, there has been no solid experimental evidence to support this point [13], [16], [21], [22]. In our view, there are two key issues that are closely linked to the performance of DNN-based RL and need to be addressed to enhance its effectiveness.

Interpretability. The interpretability of sublayers within DNNs is important in SVS, as it aids in comprehending the structure of the encoder and identifying any inherent model limitations. Traditional time-frequency analysis tools such as STFT are highly interpretable as their bases/frames have well-defined mathematical expressions and clear physical interpretations [23]. In contrast, DNN-based RL is usually difficult to interpret as it is mostly constructed on generic DNN architectures [24]. According to Monga et al. [24], generic DNNs such as convolutional/recurrent neural networks (CNNs/RNNs) only include basic convolution/recurrent operations, and it is therefore difficult to discover what is learned inside the networks by examining network parameters. In current SVS systems, the encoder and decoder are usually trained together with the separator, making it challenging to independently investigate the mechanisms behind the learning and functionality of the latent representation for SVS [16], [25]. To enhance interpretability, it is necessary to re-design or customize current DNNs for RL specifically in SVS. This would involve making the functionalities and structures of the encoder more interpretable.

General priors. According to Bengio et al. [26], general priors are essential for RL and should be used when solving artificial intelligence (AI) tasks. Such general priors in common AI tasks include smoothness, multiple explanatory factors, and sparsity [26]. However, it is still unclear what are useful general priors for RL in SVS. In our view, the general priors in SVS should have a strong connection to signal reconstruction, as the ultimate goal of SVS is to reconstruct the target vocal signal from the mixture. When designing the encoder, it is crucial to take into account these reconstruction priors. By doing so, we can obtain latent representations that are effective in achieving the desired signal reconstruction. To the best of our knowledge, there is currently no research that has thoroughly investigated the systematic integration of general priors of SVS into the encoder architecture.

To address the aforementioned challenges, we propose using deep unfolding [24] [27] in the context of RL for SVS. We introduce two models based on deep unfolding: deep unfolded representation learning (DURL) and optimal transport DURL (OT-DURL). These models aim to enhance RL methods for SVS. In both models, we formulate the task of RL as a sequence of optimization problems for signal reconstruction. To address this, we introduce three general priors for SVS: synthesis, non-negative, and our novel analysis priors. These priors are designed and integrated into the optimization problems as constraints or penalties, effectively modeling various aspects of the RL process. In DURL, the analysis prior is enforced with the Euclidean distance as a penalty. In contrast, OT-DURL uses the OT distance, which has proven to be a more effective measure for the analysis prior, to evaluate the distance between distributions in the latent representation space. Both DURL and OT-DURL address the constrained optimization problems using the first-order operator splitting algorithm. The resulting iterative algorithms are then unfolded to create novel encoders for each model. We tested the DURL and OT-DURL encoders in the unsupervised informed SVS (ISVS) framework [16], [17] and supervised Open-Unmix (UMX) framework [13]. The ISVS framework uses the ground-truth vocal signal and accompaniment signal, instead of relying on a specific separator, to compute the binary mask for SVS. Thus, this approach mitigates the impact of a specific separator and enables a more accurate evaluation of the encoder and RL performance. Experimental results indicate that the DURL and OT-DURL encoders considerably enhance the performance of vocal signal separation compared with those of other models [16], [28]–[31]. The contributions of this work are summarized as follows:

- We introduce deep unfolding into RL and two interpretable deep-unfolding-based encoders specifically tailored for SVS.
- We designed a novel analysis prior for RL and further extended it to an OT-based version, which enables better measurement of distances between different latent representations compared with the traditional Euclidean distance.
- Different from current approaches which use OT as a loss function, we use OT as a form of the analysis prior. Specifically, we unfold OT into a sublayer of the encoder, allowing the different components of the OT sublayer to be optimized through back-propagation. This sublayer can be considered a 'differentiable OT-prior' within our encoders.

II. Related Work

Enhancing the interpretability of neural structures has emerged as an intriguing issue in SVS. As a pioneering work in this field, Mimilakis et al. [16] proposed an unsupervised ISVS framework. In this framework, the separator is eliminated, allowing the encoder and decoder to be trained independently of any specific separator. To further improve interpretability, they designed a re-parameterized decoder based on a previous study [32]. This decoder has the capability to learn amplitude information from data, leading to an interpretable representation similar to STFT. However, the encoder in their framework [16] was not interpretable as it was simply composed of two-layer generic convolutional networks. The authors [16] focused solely on using the rectified linear unit (ReLU) as a non-negative prior for the encoder to promote the non-negativity and sparsity of the representation in accordance with [33], [34]. In contrast, we provide a systematic approach to designing an interpretable encoder for SVS. We also explicitly incorporate a wider range of general priors of RL into the encoder.

Deep unfolding, also known as the unrolling algorithm, has primarily been applied to image and video processing [24], [35]. We, however, introduce deep unfolding into audio/music source separation and constructed DURL and OT-DURL specifically for SVS. The use of deep unfolding enables us to systematically unfold the constrained optimization problems of SVS, incorporating general priors, into interpretable encoders. We also introduce a novel prior called analysis prior for our models and present the first work using both analysis and synthesis priors for SVS. Virtanen also explored the incorporation of priors related to temporal continuity and sparsity for non-negative-matrix factorization algorithms [36]. Our approach differs from Virtanen's in that we introduce an

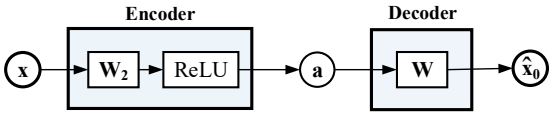


Fig. 2: Unsupervised baseline framework [16].

optimization problem that encompasses three general priors specific to SVS. We also unfold the obtained iterative algorithm to create a DNN-based encoder.

In DURL, we use the Euclidean distance to penalize the disparity between two latent representations for the analysis prior. However, the Euclidean distance is not an optimal measure for distribution distance. To improve DURL, we extend the analysis prior to an OT version, i.e., the OT-based analysis prior, within OT-DURL. The OT is a geometric measure that defines the Wasserstein distance between different distributions [37]. It is typically used as a loss function to compare various probability distributions in deep/machine-learning contexts [17], [38]–[41]. In contrast to previous research, we use OT as a general prior and integrate it as a mapping within a sublayer of the encoder. This introduces a novel trainable OT layer that effectively measures different latent representations. To the best of our knowledge, there has been no study on deep unfolding that has unfolded the OT penalty to a substructure of the neural network for SVS.

III. Baseline Framework

Fig. 2 shows the baseline SVS framework [16] that we used. Compared with the typical DNN-based SVS framework (Fig. 1), this framework does not rely on the use of a separator, enabling the encoder and decoder to be trained independently without any interference from a separator.

The input of the baseline framework is a noisy vocal signal $\mathbf{x} \in \mathbb{R}^N$. The encoder maps $\mathbf{x}$ to its latent time-frequency representation $\mathbf{a} \in \mathbb{R}^{CM}$, which consists of C channels with M frames for each channel, where C denotes the number of frequency bins and M the number of time frames. The decoder reverses this mapping and outputs an estimated vocal signal $\hat{\mathbf{x}}_0 \in \mathbb{R}^N$.

With these notations, we formulate the baseline framework as

$$\mathbf{a} = \text{Encoder}(\mathbf{x}), \qquad \text{Encoder} : \mathbb{R}^N \to \mathbb{R}^{CM}_{\geq 0}, \quad (1)$$

$$\hat{\mathbf{x}}_0 = \text{Decoder}(\mathbf{a}), \qquad \text{Decoder} : \mathbb{R}^{CM}_{\geq 0} \to \mathbb{R}^N. \quad (2)$$

A closer examination of this framework reveals that the encoder is composed of a linear transform $\mathbf{W}_2 \in \mathbb{R}^{CM \times N}$ and a nonlinear activation ReLU, i.e.,

$$\text{Encoder}(\mathbf{x}) \stackrel{\text{def}}{=} \text{ReLU}(\mathbf{W}_2\mathbf{x}), \quad (3)$$

where $\mathbf{W}_2$ is a matrix representing two one-dimensional (1D) strided convolutions[1]. More details about $\mathbf{W}_2$ can be found in a previous study [16].

The decoder in Fig. 2 models a vocal signal as a sum of C modulated sinusoidal components:

$$\text{Decoder}(\mathbf{a}) \stackrel{\text{def}}{=} \mathbf{W}\mathbf{a}, \quad (4)$$

[1]The convolution operation can be represented by circular matrix multiplication. The theory behind this can be found in Lemma 1 of [42].

where $\mathbf{W} \in \mathbb{R}^{N \times CM}$ represents the 1D transposed convolutions by C kernels. Each kernel, denoted as $\mathbf{w}_c \in \mathbb{R}^L$ $(c = 1, ..., C)$, is re-parameterized by amplitude modulated sinusoidal functions [16]:

$$\mathbf{w}_{c,l} = \cos(2\pi f_c^2\, l + \rho_c)\, \mathrm{m}_{c,l}, \quad l = [0, \ldots, L-1], \quad (5)$$

where l is the time index and c is the channel index. The trainable parameters in Eq. (5) are the sampling-rate-normalized carrier frequency $f_c \in \mathbb{R}$, phase $\rho_c \in \mathbb{R}$, and modulating signal $\mathbf{m}_c = [\mathrm{m}_{c,l}]_l \in \mathbb{R}^L$.

In the following sections, we present the DURL and OT-DURL encoders. These encoders include the baseline encoder as a special case.

IV. Deep Unfolded Representation Learning (DURL)

A. DURL

1) Synthesis prior: The input signal of DURL is the noisy input $\mathbf{x}$, assuming $\mathbf{x} = \mathbf{x}_0 + \mathbf{n}$, where $\mathbf{x}_0$ is the clean vocal signal and $\mathbf{n}$ is the noise (or accompanying music).

The encoder in SVS should output a representation $\mathbf{a}$ from $\mathbf{x}$. A general prior (assumption) is that the estimated vocal signal $\hat{\mathbf{x}}_0$ computed with $\mathbf{a}$, where $\hat{\mathbf{x}}_0 = \mathbf{W}\mathbf{a}$, should not be too far from $\mathbf{x}$, i.e., $\hat{\mathbf{x}}_0 \approx \mathbf{x}$. We then have

$$\mathbf{x} \approx \mathbf{W}\mathbf{a}, \quad (6)$$

which we call the synthesis prior. We use the standard l_2 norm (Euclidean distance) to measure the reconstruction error, i.e.,

$$\frac{1}{2}\|\mathbf{x} - \mathbf{W}\mathbf{a}\|_2^2. \quad (7)$$

By minimizing the l_2 error (7) with respect to $\mathbf{a}$, the basic RL process is formulated as the following optimization problem,

$$\underset{\mathbf{a} \in \mathbb{R}^{CM}}{\arg\min} \quad \frac{1}{2}\|\mathbf{x} - \mathbf{W}\mathbf{a}\|_2^2. \quad (8)$$

There are usually many solutions to the optimization problem (8). We use the regularizer $\|\mathbf{a}\|_2^2$ to select a low-energy latent representation $\mathbf{a}$,

$$\underset{\mathbf{a} \in \mathbb{R}^{CM}}{\arg\min} \quad \frac{1}{2}\|\mathbf{x} - \mathbf{W}\mathbf{a}\|_2^2 + \frac{\beta}{2}\|\mathbf{a}\|_2^2, \quad (9)$$

where β controls the balance between the regularizer $\frac{1}{2}\|\mathbf{a}\|_2^2$ and the l_2 error (7).

2) Non-negative prior: A general but particularly useful prior (assumption) for audio signal modeling [43] is non-negativity. The non-negative prior for the latent representation $\mathbf{a}$ is

$$\mathbf{a} \in \mathbb{R}^{CM}_{\geq 0}. \quad (10)$$

This prior promotes both non-negative and sparsity of the latent representation [16] [44].

The optimization problem (9) is extended when including the non-negative prior as

$$\underset{\mathbf{a} \in \mathbb{R}^{CM}_{\geq 0}}{\arg\min} \quad \frac{1}{2}\|\mathbf{x} - \mathbf{W}\mathbf{a}\|_2^2 + \frac{\beta}{2}\|\mathbf{a}\|_2^2. \quad (11)$$

3) Analysis prior: In a similar fashion to the synthesis prior, we introduce an analysis prior (assumption) that governs the mapping from $\mathbf{x}$ to $\mathbf{a}$, as follows

$$\mathbf{a} \approx \mathbf{W}_2\mathbf{x}. \tag{12}$$

The analysis prior is designed to complement and work in conjunction with the synthesis prior. By incorporating both priors, we achieve the following expression,

$$\hat{\mathbf{x}}_0 = \mathbf{W}\mathbf{a} \approx \mathbf{W}\mathbf{W}_2\mathbf{x}, \tag{13}$$

which can be considered as a denoising process.

We use the l_2 norm to penalize the analysis prior in (12), i.e.,

$$\frac{1}{2}\|\mathbf{a} - \mathbf{W}_2\mathbf{x}\|_2^2. \tag{14}$$

Using this penalty, we can augment the optimization problem (11) and obtain a new optimization problem,

$$\underset{\mathbf{a}\in\mathbb{R}_{\geq 0}^{CM}}{\arg\min} \quad \frac{1}{2}\|\mathbf{x} - \mathbf{W}\mathbf{a}\|_2^2 + \frac{\beta}{2}\|\mathbf{a}\|_2^2 + \frac{\rho}{2}\|\mathbf{a} - \mathbf{W}_2\mathbf{x}\|_2^2, \tag{15}$$

where ρ controls the penalty of the analysis prior. The proposed optimization problem (15) represents the full DURL with all three general priors included.

B. Solving DURL

To solve the full DURL represented by the optimization problem (15), we use the forward-backward operator splitting algorithm [45]. We first define two functions $f(\mathbf{a})$ and $g(\mathbf{a})$ to respectively represent the objective function and constraint of the full DURL as follows:

$$f(\mathbf{a}) = \frac{1}{2}\|\mathbf{x} - \mathbf{W}\mathbf{a}\|_2^2 + \frac{\beta}{2}\|\mathbf{a}\|_2^2 + \frac{\rho}{2}\|\mathbf{a} - \mathbf{W}_2\mathbf{x}\|_2^2, \tag{16}$$

$$g(\mathbf{a}) = \begin{cases} 0 & \mathbf{a} \in \mathbb{R}_{\geq 0}^{CM}, \\ +\infty & \mathbf{a} \notin \mathbb{R}_{\geq 0}^{CM}. \end{cases} \tag{17}$$

The optimization problem (15) can then be written as

$$\underset{\mathbf{a}\in\mathbb{R}^{CM}}{\arg\min} \quad f(\mathbf{a}) + g(\mathbf{a}). \tag{18}$$

With the forward-backward operator splitting method, the optimization problem (18) can be solved using the following iterative algorithm,

$$\mathbf{a}^{(k+1)} = (1-\lambda)\mathbf{a}^{(k)} + \lambda \text{Prox}_{\gamma g}\big(\mathbf{a}^{(k)} - \gamma\nabla f(\mathbf{a}^{(k)})\big), \tag{19}$$

where $0 < \lambda \leq 1$, $0 < \gamma \leq 1$, $0 \leq k \leq T$ (k denotes the iteration index), and

$$\text{Prox}_{\gamma g}(x) = \underset{u}{\arg\min} \quad \gamma g(u) + \frac{1}{2}\|u - x\|_2^2. \tag{20}$$

To compute Eq. (19), we calculate the gradient of Eq. (16):

$$\nabla f(\mathbf{a}) = (\mathbf{W}^\intercal(\mathbf{W}\mathbf{a} - \mathbf{x}) + \beta\mathbf{a}) + \rho(\mathbf{a} - \mathbf{W}_2\mathbf{x}). \tag{21}$$

Combining Eqs. (19) and (21), we obtain

$$\begin{aligned} \mathbf{a}^{(k+1)} \quad = \quad & (1-\lambda)\mathbf{a}^{(k)} + \lambda\text{Prox}_{\gamma g}\bigg(\mathbf{a}^{(k)} + \\ & \gamma\Big(\big(\mathbf{W}^\intercal(\mathbf{x} - \mathbf{W}\mathbf{a}^{(k)}) - \beta\mathbf{a}^{(k)}\big) \\ & +\rho(\mathbf{W}_2\mathbf{x} - \mathbf{a}^{(k)})\Big)\bigg). \end{aligned} \tag{22}$$

The proximal operator of g in Eq. (17) is the ReLU, i.e.,

$$\text{Prox}_{\gamma g} = \text{ReLU}. \tag{23}$$

Equation (22) can then be rewritten as

$$\begin{aligned} \mathbf{a}^{(k+1)} \quad = \quad & (1-\lambda)\mathbf{a}^{(k)} + \lambda\text{ReLU}\Big(\mathbf{a}^{(k)} + \\ & \gamma\Big(\big(\mathbf{W}^\intercal(\mathbf{x} - \mathbf{W}\mathbf{a}^{(k)}) - \beta\mathbf{a}^{(k)}\big) \\ & +\rho(\mathbf{W}_2\mathbf{x} - \mathbf{a}^{(k)})\Big)\Big), \end{aligned} \tag{24}$$

which is the iterative algorithm derived from the optimization problem (15).

The hyperparameters in Eq. (24) are $\lambda, \gamma, \beta, \rho$, and T. We can obtain different variations of this iterative algorithm by setting these hyperparameters differently. For example, when we set $\rho = 0$ and $\beta = 1$, the full DURL represented by the optimization problem Eq. (15) becomes

$$\underset{\mathbf{a}\in\mathbb{R}_{\geq 0}^{CM}}{\arg\min} \quad \frac{1}{2}\|\mathbf{x} - \mathbf{W}\mathbf{a}\|_2^2 + \frac{1}{2}\|\mathbf{a}\|_2^2, \tag{25}$$

and the corresponding iterative algorithm (Eq. (24)) becomes

$$\begin{aligned} \mathbf{a}^{(k+1)} \quad = \quad & (1-\lambda)\mathbf{a}^{(k)} + \lambda\text{ReLU}\big((1-\gamma)\mathbf{a}^{(k)} \\ & +\gamma\mathbf{W}^\intercal(\mathbf{x} - \mathbf{W}\mathbf{a}^{(k)})\big). \end{aligned} \tag{26}$$

In particular, if we set $\lambda = \gamma = 1$ and the initial state $\mathbf{a}^{(0)} = 0$, Eq. (26) becomes

$$\mathbf{a}^{(1)} = \text{ReLU}(\mathbf{W}^\intercal\mathbf{x}). \tag{27}$$

It is clear that the baseline encoder (Eq. (3)) can be obtained as one iteration of Eq. (26) when we set $\mathbf{W}^\intercal = \mathbf{W}_2$. Therefore, to include the baseline encoder as a special case, we set $\mathbf{W}^\intercal = \mathbf{W}_2$ for Eq. (24), and the DURL iterative algorithm becomes

$$\mathbf{a}^{(1)} = \text{ReLU}(\mathbf{W}_2\mathbf{x}), \tag{28}$$

$$\begin{aligned} \mathbf{a}^{(k+1)} \quad = \quad & (1-\lambda)\mathbf{a}^{(k)} + \lambda\text{ReLU}\bigg(\mathbf{a}^{(k)} + \\ & \gamma\Big(\big(\mathbf{W}_2(\mathbf{x} - \mathbf{W}\mathbf{a}^{(k)}) - \beta\mathbf{a}^{(k)}\big) \\ & +\rho(\mathbf{W}_2\mathbf{x} - \mathbf{a}^{(k)})\Big)\bigg), \end{aligned} \tag{29}$$

where $\mathbf{a}^{(k+1)}$ is the output of the k-th iteration ($1 \leq k \leq T$) and the final output is $\mathbf{a}^{(T+1)}$. We call this iterative algorithm the full DURL iterative algorithm.

C. Interpretable encoder unfolded from full DURL

1) Functional components: The DURL iterative algorithm (Eq. (29)) can be unfolded to an encoder. The main functional components in Eq. (29) are labeled in Eq. (30) (shown at the top of this page), where three general priors are mapped to corresponding sublayers: synthesis, activation, and analysis, respectively. There are also two skip connections (SCs) labeled in Eq. (30): outside and inside. The outside SC between $\mathbf{a}^{(k)}$ and the ReLU sublayer is controlled by λ. The inside SC is between $\mathbf{a}^{(k)}$ and the γ-weighted synthesis and analysis sublayers.

$$\mathbf{a}^{(k+1)} = \underbrace{(1-\lambda)\mathbf{a}^{(k)}+\lambda}_{\text{Outside SC}} \underbrace{\text{ReLU}}_{\textbf{Activation sublayer}} \Big(\underbrace{\mathbf{a}^{(k)}+\gamma\Big(\underbrace{\mathbf{W}_2(\overbrace{\mathbf{x}-\mathbf{W}\mathbf{a}^{(k)}}^{\text{Synthesis residue } \widetilde{\mathbf{x}}^{(k)}})-\beta\mathbf{a}^{(k)}}_{\textbf{Synthesis sublayer}} + \underbrace{\rho(\overbrace{\mathbf{W}_2\mathbf{x}-\mathbf{a}^{(k)}}^{\text{Analysis residue } \widetilde{\mathbf{a}}^{(k)}})}_{\textbf{Analysis sublayer}}\Big)}_{\text{Inside SC}} \Big) \quad (30)$$

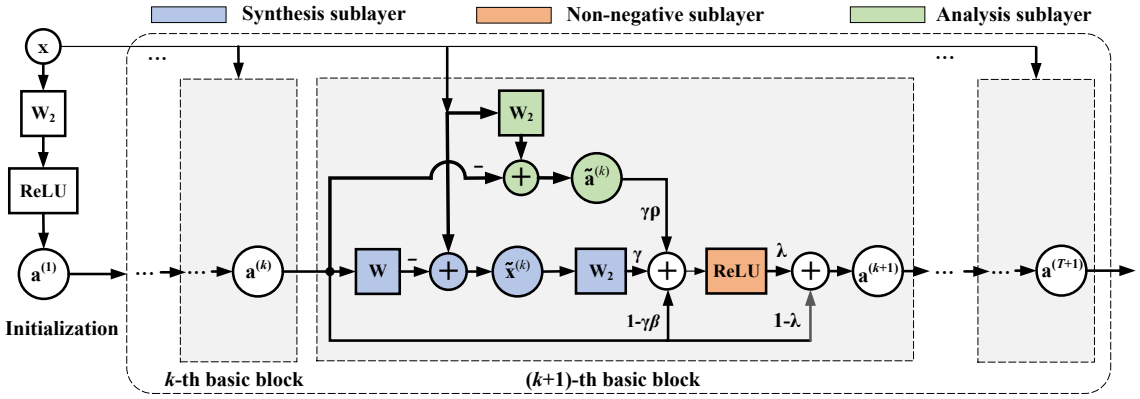


Fig. 3: Encoder unfolded from DURL iterative algorithm.

(i) Synthesis sublayer: The synthesis sublayer originates from the synthesis prior. It consists of a subtraction step of $\mathbf{W}_2\widetilde{\mathbf{x}}^{(k)}-\beta\mathbf{a}^{(k)}$, where $\mathbf{a}^{(k)}$ is the latent representation from the previous sublayer, and $\widetilde{\mathbf{x}}^{(k)}$ is the synthesis residue between the $\mathbf{x}$ and reconstructed signal $\mathbf{W}\mathbf{a}^{(k)}$, i.e.,

$$\widetilde{\mathbf{x}}^{(k)} = \mathbf{x}-\mathbf{W}\mathbf{a}^{(k)}. \quad (31)$$

The procedure $\mathbf{W}_2\widetilde{\mathbf{x}}^{(k)}$ transforms the synthesis residue $\widetilde{\mathbf{x}}^{(k)}$ to the latent representation space. It can be considered as the residue representation caused by $\widetilde{\mathbf{x}}^{(k)}$. The $\beta\mathbf{a}^{(k)}$ is from the l_2 norm regularizer in Eq. (9), where β controls the weight of the l_2 norm regularizer.

(ii) Analysis sublayer: The analysis sublayer originates from the analysis prior. It consists of a weighted analysis residue $\rho\widetilde{\mathbf{a}}^{(k)}$, where the analysis residue $\widetilde{\mathbf{a}}^{(k)}$ is defined as

$$\widetilde{\mathbf{a}}^{(k)} = \mathbf{W}_2\mathbf{x}-\mathbf{a}^{(k)}. \quad (32)$$

The $\widetilde{\mathbf{a}}^{(k)}$ iteratively computes the difference between the latent representation $\mathbf{a}^{(k)}$ and latent representation from $\mathbf{x}$, i.e., $\mathbf{W}_2\mathbf{x}$. Hyperparameter ρ controls the weight of the analysis sublayer.

(iii) Non-negative activation sublayer: The non-negative activation sublayer originates from the non-negative prior. It applies the ReLU to the summation of the inside SC and synthesis&analysis sublayers.

2) Unfolded encoder: It is easy to directly unfold Eq. (29) (i.e., Eq. (30)) to an encoder. However, to make the unfolded encoder more compact, we unfold the following equivalent equation of Eq. (29):

$$\mathbf{a}^{(k+1)} = (1-\lambda)\mathbf{a}^{(k)}+\lambda\text{ReLU}\Big((1-\gamma\beta)\mathbf{a}^{(k)}+\gamma\big(\mathbf{W}_2(\mathbf{x}-\mathbf{W}\mathbf{a}^{(k)})+\rho(\mathbf{W}_2\mathbf{x}-\mathbf{a}^{(k)})\big)\Big). \quad (33)$$

The unfolded encoder is shown in Fig. 3. The input of the encoder is $\mathbf{x}$, the initialization layer $\text{ReLU}(\mathbf{W}_2\mathbf{x})$ is from the baseline encoder, and the $(k+1)$-th layer (consisting of a synthesis sublayer, non-negative sublayer, analysis sublayer, and two SCs) is unfolded from the k-th iteration of the DURL iterative algorithm. The output of the encoder is $\mathbf{a}^{(T+1)}$. This encoder is structure-interpretable as its three sublayers are derived from the optimization problem characterized by the three general priors of SVS.

It can be challenging to ensure the cooperative functioning of these sublayers within a single neural network, given that they originate from different general priors. However, the proposed DURL tackles this issue by formulating all general priors into a unified optimization problem (15). Through the use of the forward-backward operator splitting algorithm during the optimization process, all priors and their associated penalties are coordinated in each iteration step. This coordinated approach allows the sublayers to work collaboratively and achieve their intended objectives.

V. Optimal Transport-deep Unfolded Representation Learning (OT-DURL)

The representations in the latent space can be viewed as distributions, especially when the non-negative prior is enforced. In DURL, we use Euclidean distance to penalize the disparity between two latent representations ($\mathbf{a}$ and $\mathbf{W}_2\mathbf{x}$) for the analysis prior (i.e. (14)). However, it is widely acknowledged that the Euclidean distance is not an optimal measure for comparing distributions [37], [41], [46]. To address this issue, we propose using the OT distance [37]. The OT distance defines a geometrically meaningful Wasserstein distance [37], which quantifies the distance between two distributions as the minimum cost of transporting all the mass from one distribution to another. This approach offers favorable properties for measuring distance in the latent space [41], [46], [47].

The OT distance is defined as follows. Assume two non-negative latent representations $\mathbf{p},\mathbf{q}\in\mathbb{R}_+^{CM}$ and a cost matrix $D=[d_{i,j}]\in\mathbb{R}_+^{CM\times CM}$, where D denotes the pairwise distances between two distributions, $d_{i,j}$ is the cost (weight)

of moving from $\mathbf{p}_i$ to $\mathbf{q}_j$, and "+" means a non-negative value. The OT distance between $\mathbf{p}$ and $\mathbf{q}$ is defined as

$$\mathrm{OT}(\mathbf{p},\mathbf{q}) = \min_{X\in U(\mathbf{p},\mathbf{q})} \langle D,\, X\rangle := \mathrm{Tr}(D^\intercal X), \tag{34}$$

$$U(\mathbf{p},\mathbf{q}) = \left\{ X\in\mathbb{R}_+^{CM\times CM} \middle| X\mathbb{1}_{CM}=\mathbf{p},\, X^\intercal \mathbb{1}_{CM}=\mathbf{q}\right\}, \tag{35}$$

where $\langle D,\, X\rangle$ is the inner product of D and X, Tr is the trace of a matrix, and $\mathbb{1}_{CM}$ is a vector of CM elements with each element equal to one. We approximate the OT penalty of Eq. (34) with an entropy regularized OT penalty [41], [47]:

$$\mathrm{OT}_\sigma(\mathbf{p},\mathbf{q}) = \min_{X\in U(\mathbf{p},\mathbf{q})} \langle D,\, X\rangle - \sigma E(X), \tag{36}$$

$$E(X) = -\sum_{ij} h(X_{ij}), \tag{37}$$

where $\sigma \geq 0$ and

$$h(X_{ij}) = \begin{cases} X_{ij}\log X_{ij}, & \text{if } X_{ij} > 0 \\ 0, & \text{otherwise} \end{cases}. \tag{38}$$

With the notation of Eq. (34), we define the OT-based analysis prior between $\mathbf{a}$ and $\mathbf{W}_2\mathbf{x}$ as

$$\mathrm{OT}(\mathbf{a}, \mathbf{W}_2\mathbf{x}). \tag{39}$$

Then the full DURL represented by the optimization problem (15) can be extended to the OT-DURL model[2] by replacing the Euclidean distance based analysis prior (14) with the OT-based analysis prior (39), i.e.,

$$\underset{\mathbf{a}\in\mathbb{R}^{CM}_{\geq 0}}{\arg\min} \quad \frac{1}{2}\|\mathbf{x}-\mathbf{W}\mathbf{a}\|_2^2 + \frac{\beta}{2}\|\mathbf{a}\|_2^2 + \rho\mathrm{OT}(\mathbf{a}, \mathbf{W}_2\mathbf{x}). \tag{40}$$

We use the entropy regularized OT penalty defined in Eq. (36) to approximate the OT-DURL model represented by the optimization problem (40),

$$\underset{\mathbf{a}\in\mathbb{R}^{CM}_{\geq 0}}{\arg\min} \quad \frac{1}{2}\|\mathbf{x}-\mathbf{W}\mathbf{a}\|_2^2 + \frac{\beta}{2}\|\mathbf{a}\|_2^2 + \rho\mathrm{OT}_\sigma(\mathbf{a}, \mathbf{W}_2\mathbf{x}), \tag{41}$$

which is called entropy regularized OT-DURL. This model can be easily solved by convex optimization since the entropy regularized penalty OT_σ is differentiable and has a simple form of a convex conjugate [41], [47].

A. Solving entropy regularized OT-DURL

Following a previous study [41], we solve the optimization problem (41) by first deriving its dual problem then applying a primal-dual operator splitting algorithm [45] to that dual problem.

[2]Note that we do not apply the OT distance to the synthesis prior in OT-DURL. This is because the penalty in the synthesis prior is the distance between the time-domain waveforms (i.e., $\mathbf{x}-\mathbf{W}\mathbf{a}$), and the Euclidean distance is more suitable to measure this distance.

1) Derivation of dual problem:

Proposition 1. *Given* $\mathbf{h}_1 \in \mathbf{R}^N, \mathbf{h}_2 \in \mathbf{R}^{CM}, \mathbf{h}_3 \in \mathbf{R}^{CM}$ *and*

$$\mathbf{h} = [\mathbf{h}_1; \mathbf{h}_2; \mathbf{h}_3] \in \mathbf{R}^{(N+2CM)}, \tag{42}$$

The dual problem of the primal problem (41) can be written as

$$-\arg\min_{\mathbf{h}} \quad \tilde{f}(\mathbf{h}) + g^*\{(-\mathbf{W}^\intercal, -\mathbf{I}, -\mathbf{I})\mathbf{h}\}, \tag{43}$$

where $\mathbf{I}$ *denotes the identity matrix and the function* $\tilde{f}$ *is defined as*

$$\tilde{f}(\mathbf{h}) = \frac{1}{2}\|\mathbf{h}_1\|_2^2 + \langle \mathbf{x},\, \mathbf{h}_1\rangle + \rho\mathrm{OT}^*_\sigma(\mathbf{W}_2\mathbf{x}, \mathbf{h}_2/\rho) + \frac{\beta}{2}\|\mathbf{h}_3/\beta\|_2^2, \tag{44}$$

and $g(\mathbf{a})$ *is defined in Eq. (17). The optimal solution* $\mathbf{a}^\star$ *of the primal problem (41) satisfies a primal-dual relationship of*

$$\mathbf{a}^\star = \nabla\mathrm{OT}^*_\sigma(\mathbf{W}_2\mathbf{x}, \mathbf{h}_2^\star/\rho), \tag{45}$$

where the vector differential operator ∇ *operates on the second variable of* OT^*_σ *and* $\mathbf{h}^\star = [\mathbf{h}_1^\star; \mathbf{h}_2^\star; \mathbf{h}_3^\star]$ *is a solution of the dual problem (43).*

2) Solving dual problem: Similar to Theorem 9 [48], we use the primal-dual operator splitting algorithm [45] to solve the dual problem (43).We can then obtain the following iterative algorithm of OT-DURL,

$$\mathbf{h}^{(k+1)} = \mathbf{h}^{(k)} - \tau(\nabla\tilde{f}(\mathbf{h}^{(k)}) - [\mathbf{W}; \mathbf{I}; \mathbf{I}]\mathbf{a}^{(k)}); \tag{46}$$

$$\xi^{(k+1)} = 2\mathbf{h}^{(k+1)} - \mathbf{h}^{(k)}; \tag{47}$$

$$\mathbf{a}^{(k+1)} = \mathrm{prox}_{\gamma g}(\mathbf{a}^{(k)} - \gamma(\mathbf{W}^\intercal, \mathbf{I}, \mathbf{I})\xi^{(k+1)}). \tag{48}$$

Using Eq. (23), $\tilde{f}$ (Eq. (44)), $\mathbf{h}$ (Eq. (42)), and g (Eq. (17)), the iterative algorithm (Eqs. (46)-(48)) can be written as

$$\mathbf{h}_1^{(k+1)} = \mathbf{h}_1^{(k)} - \tau(\nabla_{\mathbf{h}_1^{(k)}}\tilde{f}(\mathbf{h}^{(k)}) - \mathbf{W}\mathbf{a}^{(k)}); \tag{49}$$

$$\mathbf{h}_2^{(k+1)} = \mathbf{h}_2^{(k)} - \tau(\nabla_{\mathbf{h}_2^{(k)}}\tilde{f}(\mathbf{h}^{(k)}) - \mathbf{a}^{(k)}); \tag{50}$$

$$\mathbf{h}_3^{(k+1)} = \mathbf{h}_3^{(k)} - \tau(\nabla_{\mathbf{h}_3^{(k)}}\tilde{f}(\mathbf{h}^{(k)}) - \mathbf{a}^{(k)}); \tag{51}$$

$$\xi^{(k+1)} = 2\mathbf{h}^{(k+1)} - \mathbf{h}^{(k)}; \tag{52}$$

$$\mathbf{a}^{(k+1)} = \mathrm{ReLU}(\mathbf{a}^{(k)} - \gamma(\mathbf{W}^\intercal, \mathbf{I}, \mathbf{I})\xi^{(k+1)}), \tag{53}$$

where

$$\nabla_{\mathbf{h}_1^{(k)}}\tilde{f}(\mathbf{h}^{(k)}) = \nabla_{\mathbf{h}_1^{(k)}}\{\frac{1}{2}\|\mathbf{h}_1^{(k)}\|_2^2 + \langle \mathbf{x},\, \mathbf{h}_1^{(k)}\rangle\} = \mathbf{h}_1^{(k)} + \mathbf{x}; \tag{54}$$

$$\nabla_{\mathbf{h}_2^{(k)}}\tilde{f}(\mathbf{h}^{(k)}) = \nabla_{\mathbf{h}_2^{(k)}}\{\rho\mathrm{OT}^*_\sigma(\mathbf{W}_2\mathbf{x}, \mathbf{h}_2^{(k)}/\rho)\} = \theta\odot\left(K^\intercal\frac{\mathbf{W}_2\mathbf{x}}{K\theta}\right) \quad (\text{with } K := \mathbf{e}^{-D/\sigma} \text{ and } \theta := \mathbf{e}^{\mathbf{h}_2^{(k)}/\rho\sigma}); \tag{55}$$

$$\nabla_{\mathbf{h}_3^{(k)}}\tilde{f}(\mathbf{h}^{(k)}) = \nabla_{\mathbf{h}_3^{(k)}}\{\frac{\beta}{2}\|\mathbf{h}_3^{(k)}/\beta\|_2^2\} = \mathbf{h}_3^{(k)}/\beta. \tag{56}$$

Hyperparameter σ ($\sigma > 0$) controls the entropy regularizer and ρ ($\rho > 0$) controls the penalty weight of the OT-based analysis prior. The computation of Eq. (55) is based on a

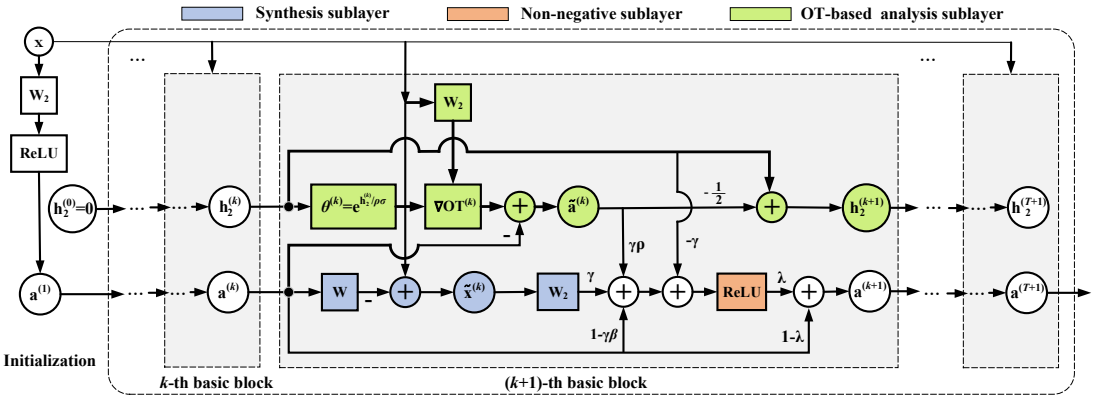


Fig. 4: Encoder unfolded from OT-DURL iterative algorithm.

previous study [49], where the OT_σ^* is differentiable with the σ-Lipschitz gradient, i.e.,

$$\nabla_{\mathbf{y}}\text{OT}_\sigma^*(\mathbf{x},\mathbf{y}) \quad = \quad \theta \odot \left(K^\intercal \frac{\mathbf{x}}{K\theta}\right), \tag{57}$$

where $K := \mathbf{e}^{-D/\sigma}$ and $\theta := \mathbf{e}^{\mathbf{y}/\sigma}$.

3) Comparison between DURL and OT-DURL: The iterative algorithms of both DURL and OT-DURL are derived from three general priors, so we compare their differences. First, we set $\tau = 1/2$, $\beta = 1$, and $\mathbf{W}^\intercal = \mathbf{W}_2$, then the primal-dual iterative algorithm (Eqs. (49)-(56)) becomes

$$\theta^{(k)} \quad = \quad \mathbf{e}^{\mathbf{h}_2^{(k)}/\rho\sigma}; \tag{58}$$

$$\nabla\text{OT}^{(k)} \quad = \quad \theta^{(k)} \odot \left(K^\intercal \frac{\mathbf{W}_2\mathbf{x}}{K\theta^{(k)}}\right); \tag{59}$$

$$\widetilde{\mathbf{a}}^{(k)} \quad = \quad \nabla\text{OT}^{(k)} - \mathbf{a}^{(k)}; \tag{60}$$

$$\widetilde{\mathbf{x}}^{(k)} \quad = \quad \mathbf{x} - \mathbf{W}\mathbf{a}^{(k)}; \tag{61}$$

$$\varphi^{(k+1)} \quad = \quad -\mathbf{W}_2\widetilde{\mathbf{x}}^{(k)} + \mathbf{h}_2^{(k)} - \widetilde{\mathbf{a}}^{(k)} + \mathbf{a}^{(k)}; \tag{62}$$

$$\mathbf{a}^{(k+1)} \quad = \quad \text{ReLU}(\mathbf{a}^{(k)} - \gamma\varphi^{(k+1)}); \tag{63}$$

$$\mathbf{h}_2^{(k+1)} \quad = \quad \mathbf{h}_2^{(k)} - 1/2\widetilde{\mathbf{a}}^{(k)}, \tag{64}$$

where $\widetilde{\mathbf{a}}^{(k)}$ (Eq. (60)) is the OT-based analysis residue (similar to Eq. (32) in DURL), $\widetilde{\mathbf{x}}^{(k)}$ (Eq. (61)) is the synthesis residue (the same as Eq. (31)), and $\mathbf{a}^{(k+1)}$ is the main iteration step.

When substituting $\varphi^{(k+1)}$ (Eq. (62)) for Eq. (63), the main iteration step in Eq. (63) becomes

$$\begin{aligned}\mathbf{a}^{(k+1)} \quad = \quad & \text{ReLU}\Big((1-\gamma)\mathbf{a}^{(k)} + \gamma\mathbf{W}_2\widetilde{\mathbf{x}}^{(k)} \\ & +\gamma\widetilde{\mathbf{a}}^{(k)} - \gamma\mathbf{h}_2^{(k)}\Big).\end{aligned} \tag{65}$$

Considering the definition of $\widetilde{\mathbf{a}}^{(k)}$ (Eq. (60)) and $\widetilde{\mathbf{x}}^{(k)}$ (Eq. (61)), we obtain

$$\begin{aligned}\mathbf{a}^{(k+1)} \quad = \quad & \text{ReLU}\Big((1-\gamma)\mathbf{a}^{(k)} + \gamma\mathbf{W}_2(\mathbf{x} - \mathbf{W}\mathbf{a}^{(k)}) \\ & +\gamma(\nabla\text{OT}^{(k)} - \mathbf{a}^{(k)}) - \gamma\mathbf{h}_2^{(k)}\Big).\end{aligned} \tag{66}$$

We can see that the iterative algorithms (Eq. (66) and Eq. (29)) have similar architectures. If we set $\lambda = \rho = \beta = 1$, the DURL iterative algorithm (Eq. (29)) can be simplified as

$$\begin{aligned}\mathbf{a}^{(k+1)} \quad = \quad & \text{ReLU}\Big((1-\gamma)\mathbf{a}^{(k)} + \gamma\mathbf{W}_2(\mathbf{x} - \mathbf{W}\mathbf{a}^{(k)}) \\ & +\gamma(\mathbf{W}_2\mathbf{x} - \mathbf{a}^{(k)})\Big).\end{aligned} \tag{67}$$

By comparing Eqs. (66) and (67), we can observe that the iterative algorithm of OT-DURL brings two new iteration steps of $\nabla\text{OT}^{(k)}$ and $\mathbf{h}_2^{(k)}$ to the DURL iterative algorithm. That is, OT-DURL will have exactly the same architecture as DURL if we remove $\mathbf{h}_2^{(k)}$ from the OT-DURL algorithm and replace $\nabla\text{OT}^{(k)}$ with $\mathbf{W}_2\mathbf{x}$.

It should be noted that the extra step $\nabla\text{OT}^{(k)}$ (Eq. (59)) introduced in OT-DURL has more potential than $\mathbf{W}_2\mathbf{x}$ in DURL, because it uses not only $\mathbf{W}_2\mathbf{x}$ but also extra information of $\theta^{(k)}$ (Eq. (58)) which includes an adaptive step $\mathbf{h}_2^{(k)}$ that can adjust its value for each iteration step.

B. Interpretable encoder unfolded from OT-DURL

The OT-DURL encoder can be implemented directly using the iterative algorithm (Eqs. (58)-(64)). However, considering that OT-DURL is an extension of DURL, we choose to implement the OT-DURL encoder on the basis of the DURL encoder. This approach enables us to validate the effectiveness of the OT-based analysis prior and ensure a fair comparison between the DURL and OT-DURL encoders.

To inherit the main structure of the DURL encoder (see Eq. (33)), we introduce the following changes (underlined parts) to the main iteration step (i.e. Eq. (65)) of OT-DURL:

$$\begin{aligned}\mathbf{a}^{(k+1)} \quad = \quad & \underline{(1-\lambda)\mathbf{a}^{(k)}} + \underline{\lambda}\text{ReLU}\big((1-\gamma\underline{\beta})\mathbf{a}^{(k)} \\ & +\gamma\mathbf{W}_2\widetilde{\mathbf{x}}^{(k)} + \gamma\underline{\rho}\widetilde{\mathbf{a}}^{(k)} - \gamma\mathbf{h}_2^{(k)}\big).\end{aligned} \tag{68}$$

By substituting Eq. (63) with (68), we proceed to unfold the OT-DURL iterative algorithm (Eqs. (58)-(64)) to construct the OT-DURL encoder. As shown in Fig. 4, the OT-based analysis residue $\widetilde{\mathbf{a}}^{(k)}$ (Eq. (60)) and synthesis residue $\widetilde{\mathbf{x}}^{(k)}$ (Eq. (61)) correspond to the OT analysis sublayer and synthesis sublayer, respectively, where the OT analysis sublayer has two new iteration steps for $\nabla\text{OT}^{(k)}$ and $\mathbf{h}_2^{(k+1)}$, respectively, generated from the OT-based analysis prior.

VI. Evaluation of DURL and OT-DURL Encoders

A. Evaluation conditions

We followed the experimental settings in the study by Mimilakis et al. [16] to evaluate the DURL and OT-DURL encoders. The database was MUSDB18 [50], which consists of 150 two-channel multi-track signals sampled at 44.1 kHz (100 for training and 50 for testing).

1) Training: We trained the DURL and OT-DURL encoders in an unsupervised manner with four multi-tracks as those used by Mimilakis et al. [16]. Each multi-track was down-mixed to single-channel for vocal signal and accompaniment (acc) signal, respectively. Both the vocal and acc sources were partitioned into overlapping clips of $N = 44100$ samples with an overlap of 22050 samples [50]. Following Mimilakis et al. [16], we carried out random shuffling of the clips for each source and introduced corruption to the clean vocal clips in two different manners. For a given clean vocal clip $\mathbf{x}_{\rm v}$, we created two noisy clips $\mathbf{x}_{\rm m}$ and $\mathbf{x}_{\rm n}$, by mixing it with an acc source and Gaussian noise, respectively. The loss function [16] for training was

$$\hat{\mathbf{x}}_{\rm n} = \text{Decoder}\big(\text{Encoder}(\mathbf{x}_{\rm n})\big), \quad (69)$$

$$\mathbf{A}_{\rm m} = \text{Encoder}(\mathbf{x}_{\rm m}), \quad (70)$$

$$\mathcal{L} = \mathcal{L}_{\text{neg-SNR}}(\mathbf{x}_{\rm v}, \hat{\mathbf{x}}_{\rm n}) + \omega\, \mathcal{L}_{\rm TV}(\mathbf{A}_{\rm m}), \quad \omega > 0, \quad (71)$$

where $\mathcal{L}_{\text{neg-SNR}}$ is the negative signal-to-noise ratio (neg-SNR) [51], $\mathcal{L}_{\rm TV}$ is the total-variation (TV) denoising loss, and ω is the weight for the TV regularizer. The batch size for training was 8 with a learning rate of $1\text{e}{-4}$ [16].

2) Testing: We used 50 testing tracks as in the study by Mimilakis et al. [16] for testing. The DURL and OT-DURL encoders were tested in the ISVS framework [16] [25], which evaluates the upper-bound performance of a latent representation with an informed binary mask [16].

Specifically, for an encoder, the ISVS computes three latent representations of a testing signal: the mixture $\mathbf{x}_{\rm m} = \mathbf{x}_{\rm ac} + \mathbf{x}_{\rm v}$, acc signal $\mathbf{x}_{\rm ac}$, and vocal signal $\mathbf{x}_{\rm v}$, i.e.,

$$\mathbf{A}_{\rm m} = \text{Encoder}(\mathbf{x}_{\rm m}), \quad (72)$$

$$\mathbf{A}_{\rm ac} = \text{Encoder}(\mathbf{x}_{\rm ac}), \quad (73)$$

$$\mathbf{A}_{\rm v} = \text{Encoder}(\mathbf{x}_{\rm v}). \quad (74)$$

The informed binary mask $\mathbf{G}_{\rm v} \in \mathbb{R}^{CM}$ can be computed on the basis of the element-wise division of $\mathbf{A}_{\rm v}$ and $\mathbf{A}_{\rm ac}$, i.e.,

$$\mathbf{G}_{\rm v} = Q(\mathbf{A}_{\rm v} \oslash \mathbf{A}_{\rm ac}), \quad (75)$$

where $Q(\cdot)$ is defined as

$$Q(\text{x}) = \begin{cases} 1, & \text{if x} \geq 0.5; \\ 0, & \text{otherwise.} \end{cases} \quad (76)$$

Using $\mathbf{G}_{\rm v}$, we can compute the estimated vocal signal $\hat{\mathbf{x}}_{\rm v}$ by

$$\tilde{\mathbf{A}} = \mathbf{A}_{\rm m} \odot \mathbf{G}_{\rm v}, \quad (77)$$

$$\hat{\mathbf{x}}_{\rm v} = \text{Decoder}(\tilde{\mathbf{A}}), \quad (78)$$

where $\hat{\mathbf{x}}_{\rm v}$ is the estimated vocal signal and "$\odot$" is the element-wise (Hadamard) product. This process is called 'informed' SVS as we use the ground-truth vocal signal and acc signal to compute $\mathbf{G}_{\rm v}$ when estimating the vocal signal.

Following Mimilakis et al. [16], we used the scale-invariant signal-to-distortion ratio (SI-SDR) [52] to evaluate the separation performance of the encoders. The SI-SDR between $\mathbf{x}_{\rm v}$ and $\hat{\mathbf{x}}_{\rm v}$, denoted as SI-SDR-BM[3], is defined as

$$\text{SI-SDR-BM}(\mathbf{x}_{\rm v}, \hat{\mathbf{x}}_{\rm v}) = 10\log_{10}\Big(\frac{||\alpha\mathbf{x}_{\rm v}||_2^2}{||\alpha\mathbf{x}_{\rm v} - \hat{\mathbf{x}}_{\rm v}||_2^2}\Big), \quad (79)$$

$$\alpha = \frac{\hat{\mathbf{x}}_{\rm v}^{\rm T}\mathbf{x}_{\rm v}}{||\mathbf{x}_{\rm v}||_2^2}. \quad (80)$$

3) Hyperparameters: The hyperparameters in the DURL and OT-DURL encoders are β, ρ, and σ for general priors, γ and λ for SCs, ω for the $\mathcal{L}_{\rm TV}$ loss function, and other hyperparameters C, L, and T. Hyperparameters C and ω were inherited from the baseline encoder, therefore were set the same as those in the baseline, where $C = [400, 800, 1600]$, $L = 2048$, and $\omega = 0.5$. The other hyperparameters were set as follows.

Default settings for DURL encoder: For DURL (Eq. (33)), we set $\beta = \rho = 1$ to equally weight the analysis and synthesis priors as we have no prior knowledge for them. Hyperparameter λ was set to 0.1 as we expected that the SC outside the ReLU sublayer would be highly weighted to ensure an easier gradient propagation for deeper layers. Inside the ReLU sublayer, we set γ to 0.9 to emphasize the analysis&synthesis priors, i.e., the inside SC takes 10% weight and the analysis&synthesis priors take 90%. Hyperparameter T was set to 3.

Default settings for the OT-DURL encoder: We inherited all the hyperparameters in DURL for OT-DURL, except for β and T. In OT-DURL, we set $\beta = 0$, since the second item $\frac{\beta}{2}\|\mathbf{a}\|_2^2$ in OT-DURL (Eq. (40)) is a Euclidean-distance-based penalty and should be weighted with a low value or zero so that the OT-based analysis prior can work for OT-DURL[4]. In OT-DURL, we set $T = 2$ for low resolution of $C = 400$ and $T = 3$ for high resolution of $C = 800/1600$. We also set $\sigma = 1$ for OT-DURL.

Experimental setting: To fully examine the performance of the DURL and OT-DUR encoders, we evaluated them with a wider range of values around their default values. Hyperparameter β for the synthesis prior was set to $[0, 0.2, 0.5, 0.8, 1.2]$ for both encoders, ρ for the analysis prior was set to $[0, 0.8, 1.2, 1.6, 2.0]$ for DURL and $[0.8, 1.2, 1.6, 2.0]$ for OT-DURL since 0 cannot be a divisor in Eq. (58). Hyperparameter γ for the inside SC was set to $[0.6, 0.7, 0.8, 0.9, 1.0]$ and λ for the outside SC was set to $[0.05, 0.08, 0.12, 0.15, 0.18]$.

B. Performance of DURL and OT-DURL encoders

1) Effectiveness of l_2 regularizer in synthesis prior: We tested the DURL and OT-DURL encoders as a function of β with the other hyperparameters set as default. The median SI-SDR-BMs of the DURL and OT-DURL encoders under different β are plotted in Fig. 5, where the gray curves are for the DURL encoder and the blue curves are for the OT-DURL encoder. We also labeled the highest and lowest SI-SDR-BM of each curve with the corresponding color.

The SI-SDR-BM of the DURL encoder increased then decreased for all Cs. The best SI-SDR-BM of DURL was obtained at $\beta = 0.5$ for $C = 400$ and $\beta = 0.8$ for $C = 800, 1600$, which were clearly higher than those of $\beta = 0$,

[3]The BM represents Binary Masking (BM) as in [16].

[4]In the derivation of the OT-DURL algorithm, we keep the l_2 norm of $\|\mathbf{a}\|_2^2$ ($\beta \neq 0$) for the comparison between the DURL and OT-DURL.

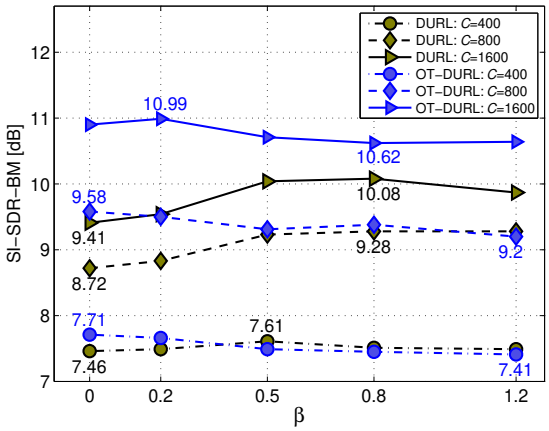


Fig. 5: Performance of DURL and OT-DURL affected by β.

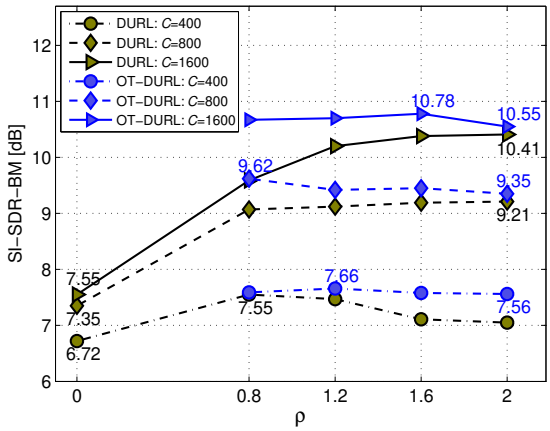


Fig. 6: Performance of DURL and OT-DURL affected by ρ.

i.e., without using the regularizer $\frac{1}{2}\|\mathbf{a}\|_2^2$. These results suggest that $\frac{1}{2}\|\mathbf{a}\|_2^2$ in the synthesis prior is useful to improve the separation performance of the DURL encoder, and we can obtain a satisfactory SI-SDR-BM if $\frac{1}{2}\|\mathbf{a}\|_2^2$ is weighted by a suitable β. In particular, the gap between the best SI-SDR-BM and worst SI-SDR-BM of the DURL encoder (obtained with $\beta = 0$) increased with C: the best SI-SDR-BM for $C = 400, 800, 1600$ was 0.15, 0.56, and 0.67 dB higher than that of $\beta = 0$, respectively, which indicates that the performance of $\frac{1}{2}\|\mathbf{a}\|_2^2$ can be better exploited under high feature resolution.

Similar to DURL, the SI-SDR-BM of OT-DURL also increased with an increase in C. However, different from DURL, OT-DURL achieved its best SI-SDR-BM at around $\beta = 0$ for all different Cs: the highest SI-SDR-BM of OT-DURL was obtained at $\beta = 0$ for $C = 400, 800$ and $\beta = 0.2$ for $C = 1600$. It is clear that the performance of OT-DURL decreased with an increase in β. The reason behind this phenomenon may be that the OT penalty, i.e., OT_σ, is a more effective distance measure than the Euclidean distance, and if the Euclidean-distance-based regularizer is highly weighted (by large β), OT-DURL will be dominated by it and the OT penalty cannot work well for OT-DURL.

By comparing the results of DURL and OT-DURL, we can find that OT-DURL achieved much better results than DURL for $C = 800, 1600$, especially for small β. These results indicate the effectiveness of the OT-based analysis prior for SVS.

2) Effectiveness of analysis prior: Hyperparameter ρ is crucial for the analysis prior. We tested the performance of the DURL and OT-DURL encoders as a function of ρ with the other hyperparameters set as default values.

Fig. 6 shows that the SI-SDR-BM of DURL increased significantly with the increase in ρ for $C = 800, 1600$. The lowest SI-SDR-BM of DURL for all Cs appeared at $\rho = 0$, i.e., without using the analysis prior. These results verified the effectiveness of the analysis prior for DURL. In contrast to DURL, the SI-SDR-BM of OT-DURL did not show a clear change with ρ.

We can also see that the SI-SDR-BMs of both encoders increased with an increase in C. The best SI-SDR-BMs of both encoders were obtained at $C = 1600$. When comparing OT-DURL with DURL, we can see that OT-DURL achieved a better SI-SDR-BM than DURL for each C under the same ρ, which verifies the advantage of the OT-based analysis prior over the Euclidean-distance-based analysis prior.

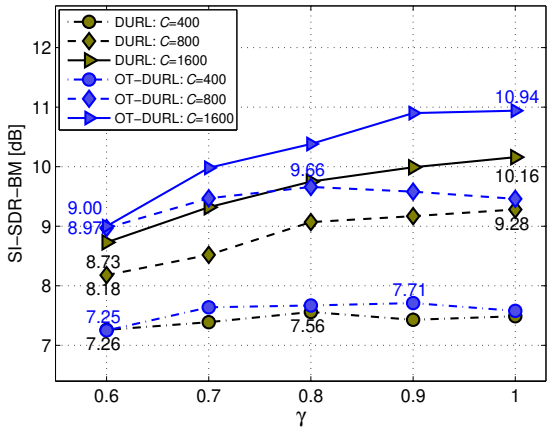


Fig. 7: Performance of DURL and OT-DURL affected by γ.

3) Performance affected by skip connections: We tested the performance of DURL and OT-DURL encoders as a function of γ. As in Eqs. (33) and (68), $1 - \gamma\beta$ controls the weight of the inside SC $\mathbf{a}^{(k)}$ in the ReLU for both encoders. Since we set $\beta = 1$ by default for DURL, $1 - \gamma$ is the weight of the inside SC in the ReLU. For OT-DURL, we set $\beta = 0$ by default; thus, the weight of the inside SC in the ReLU is always 1. Hyperparameter γ also affects the synthesis&analysis priors in both encoders.

The performance of DURL and OT-DURL computed with $\gamma = [0.6, 0.7, 0.8, 0.9, 1.0]$ is plotted in Fig. 7. Both encoders had similar performance for $C = 400$, which did not show much change with different γ. The results for $C = 800, 1600$, in contrast, greatly improved when γ was increased. As stated above, $1 - \gamma$ controls the weight of the inside SC and γ controls the weight of the synthesis and analysis priors in DURL. Therefore, the above results indicate that emphasizing the synthesis and analysis priors and attenuating the inside SC is helpful to improve the performance of DURL.

The performance of OT-DURL improved by increasing γ, especially for $C = 1600$. Since γ in OT-DURL only controlled the synthesis prior and OT-based analysis prior, the performance of OT-DURL can be optimized with a reasonably weighted

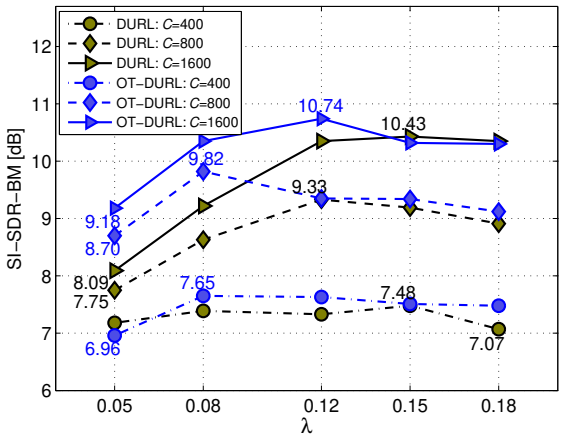


Fig. 8: Performance of DURL and OT-DURL affected by λ.

synthesis prior and OT-based analysis prior.

We then evaluated the performance of DURL and OT-DURL as a function of λ. As shown in Eqs. (33) and (68), $1-\lambda$ controls the outside SC of $\mathbf{a}^{(k)}$ and λ controls the weight of the ReLU. Fig. 8 shows the results. The highest performance of both encoders was obtained when λ was set between 0.08 and 0.15. When λ was reduced to lower than 0.08, the performance of both encoders dropped significantly, as the weight of the ReLU block was too low. It was also found that λ higher than 0.15 was not favored in both encoders. Therefore, λ should be well controlled to ensure the performance of both encoders.

VII. Comparative Evaluations

A. Comparative evaluations in ISVS framework

We first compared the DURL and OT-DURL encoders with other encoders in the unsupervised ISVS framework. The compared encoders were the baseline encoder [16], one-, two-, and three-layer 1D convolution encoders [53] (denoted as Conv1, Conv2, and Conv3), Sinkhorn-distance-based encoder (SinkH) [17], SincNet encoder [16], and other variations of the baseline encoder.

We then implemented the other variations of the baseline encoder by incorporating effective neural substructures to the baseline encoder. These included (1) the self-attention (SA) mechanism [28], (2) optimal transport kernel embedding (OTKE) [29], (3) highway (HW) network [30], and (4) convolutional kernel network (CKN) [31]. OTKE uses an OT sublayer to execute a weighted pooling operation [29] and CKN is a hybrid of CNNs and kernel method [31]. We implemented the variations of the baseline encoder by placing each substructure before and after the ReLU function of the baseline encoder, respectively. These variations are named in the form of "Substructure-B(Before)/A(After)", e.g., "SA-B" is the SA-based baseline encoder with the SA placed before the ReLU function.

1) Computation complexity: We compared the computation complexity of different encoders in Table I. The results are measured by floating point operations (FLOPs [G]) and parameter amount (Param [M]). We can see that the DURL encoders had higher FLOPs than most encoders for each C and the OT-DURL encoders had even higher FLOPs. Similar to FLOPs, the DURL and OT-DURL encoders also had greater Param than most other encoders. We found that the Param of the DURL and OT-DURL encoders did not increase with the number of sublayers for each C, but the FLOPs did. This is because the parameters in DURL and OT-DURL encoders can be shared for multilayers while the computational load (measured by FLOPs) increases when the number of sublayers is increased.

TABLE I: Comparison of computation complexity between DURL and OT-DURL encoders and other encoders in ISVS framework, where "DURL/OT-DURL-1/2/3" stands for DURL/ OT-DURL encoder of 1/2/3 layers.

Model	FLOPs [G]			Param [M]		
	400	800	1600	400	800	1600
Baseline	0.26	1.03	4.13	1.54	4.61	15.33
SincNet	0.26	1.03	4.13	0.77	3.06	12.21
Conv1	0.07	0.15	0.29	1.56	3.13	6.25
Conv2	0.09	0.23	0.64	2.02	4.96	13.57
Conv3	0.12	0.32	0.98	2.48	6.79	20.90
SA-B	0.31	1.00	3.42	1.54	4.61	15.33
SA-A	0.31	1.00	3.42	1.54	4.61	15.33
OTKE-B	0.32	1.00	3.48	1.61	4.75	5.60
OTKE-A	0.32	1.00	3.48	1.61	4.75	5.60
HW1-B	0.32	1.17	4.39	1.63	4.70	15.42
HW1-A	0.32	1.17	4.39	1.63	4.70	15.42
HW3-B	0.46	1.43	4.93	1.80	4.87	15.59
HW3-A	0.46	1.43	4.93	1.80	4.87	15.59
CKN-B	0.88	4.94	32.59	2.61	8.89	32.42
CKN-A	0.88	4.94	32.59	2.61	8.89	32.42
SinkH	0.26	1.03	4.13	1.54	4.61	15.33
DURL-1	0.77	3.09	12.37	2.33	6.18	18.46
DURL-2	1.29	5.16	20.62	2.33	6.18	18.46
DURL-3	1.81	7.22	28.87	2.33	6.18	18.46
OT-DURL-1	0.82	3.18	12.55	2.38	6.24	18.52
OT-DURL-2	1.38	5.33	20.98	2.38	6.24	18.52
OT-DURL-3	1.94	7.49	29.41	2.38	6.24	18.52

2) Separation performance: We compared all encoders with respect to their SI-SDR-BM performance. We used default hyperparameters for the DURL and OT-DURL encoders. According to Mimilakis et al. [16], we also computed the reconstruction performance of all encoders. The reconstructed vocal signal $\hat{\mathbf{x}}_{\mathrm{v}}^0$ was computed by

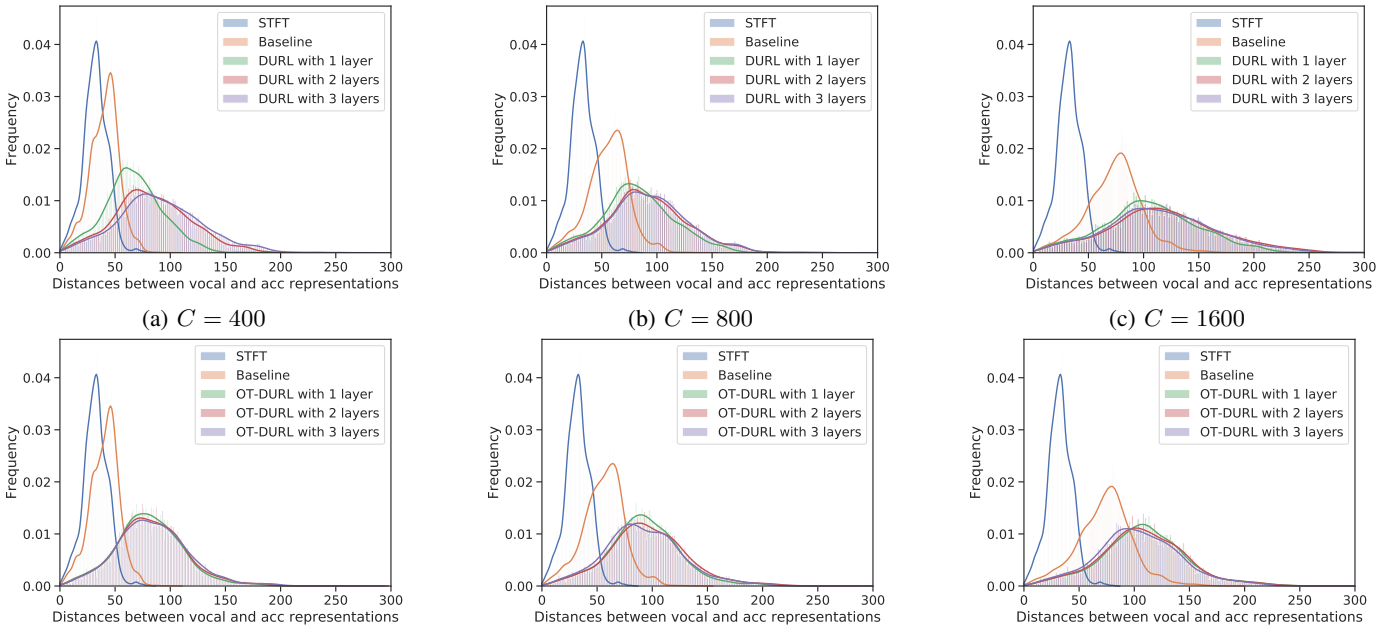
$$\hat{\mathbf{x}}_{\mathrm{v}}^0 = \mathrm{decoder}(\mathbf{A}_{\mathrm{v}}). \tag{81}$$

The reconstruction performance, measured using SI-SDR-RC (RC represents ReConstruction), is calculated by

$$\text{SI-SDR-RC}(\mathbf{x}_{\mathrm{v}}, \hat{\mathbf{x}}_{\mathrm{v}}^0) = 10\log_{10}\Big(\frac{||\alpha\mathbf{x}_{\mathrm{v}}||_2^2}{||\alpha\mathbf{x}_{\mathrm{v}} - \hat{\mathbf{x}}_{\mathrm{v}}^0||_2^2}\Big), \tag{82}$$

$$\alpha = \frac{\mathbf{x}_{\mathrm{v}}^{\mathsf{T}}\hat{\mathbf{x}}_{\mathrm{v}}^0}{||\mathbf{x}_{\mathrm{v}}||_2^2}. \tag{83}$$

The SI-SDR-BM (BM for short) and SI-SDR-RC (RC for short) of different encoders are listed in Table II. The SI-SDR-BM of the traditional STFT-based encoder is also reported for reference, where a hamming window with 2048 samples and hop-size of 256 were used to compute STFT. According to Mimilakis et al. [16], we set $C = 1025$ for the STFT-based encoder. We can see that the DURL encoders achieved significant SI-SDR-BM improvement for each C compared with other encoders, and the OT-DURL encoders achieved even better SI-SDR-BMs than the DURL encoders. Furthermore,



(a) $C = 400$ (b) $C = 800$ (c) $C = 1600$

(d) $C = 400$ (e) $C = 800$ (f) $C = 1600$

Fig. 9: Histogram of l_1-norm distances between vocal representation and acc. representation produced with different encoders.

TABLE II: Comparison of separation performance between DURL and OT-DURL encoders and other encoders in ISVS framework (in dB).

Model	BM			RC		
	400	**800**	**1600**	**400**	**800**	**1600**
Baseline	5.93	6.28	6.68	30.73	32.11	31.54
SincNet	4.68	5.69	5.98	25.40	28.55	27.21
Conv1	2.62	3.54	3.75	10.82	17.02	21.48
Conv2	2.55	3.33	3.57	10.49	16.43	18.10
Conv3	2.23	3.07	1.18	12.84	15.06	15.70
SA-B	4.43	5.12	5.48	29.50	33.11	33.58
SA-A	4.45	5.12	5.49	29.35	33.40	33.48
OTKE-B	4.16	4.84	5.25	28.84	33.02	32.42
OTKE-A	4.17	4.80	5.32	28.85	33.19	33.26
HW1-B	5.25	6.33	5.40	29.41	29.68	29.82
HW1-A	5.26	6.26	5.46	28.27	29.34	31.12
HW3-B	4.86	5.85	3.69	25.45	28.17	31.26
HW3-A	4.85	4.90	4.33	27.36	24.56	29.04
CKN-B	4.99	5.89	7.65	27.83	29.60	25.37
CKN-A	5.02	5.91	7.92	26.20	28.00	24.69
SinkH	–	5.63	–	–	31.61	–
STFT	8.80			N/A		
DURL-1	7.10	7.65	8.02	32.53	33.60	32.33
DURL-2	7.40	8.86	8.90	31.66	32.72	32.29
DURL-3	7.43	9.17	9.99	31.17	32.45	33.02
OT-DURL-1	7.53	9.23	10.05	**33.22**	**34.31**	**33.74**
OT-DURL-2	**7.71**	9.38	10.39	31.99	32.79	33.03
OT-DURL-3	6.93	**9.58**	**10.90**	31.33	33.13	32.22

the DURL and OT-DURL encoders gave slightly better SI-SDR-RC than the baseline encoder and certain variations of the baseline encoder.

We can see that the DURL encoders achieved greater SI-SDR-BM improvement for each C as compared with other encoders. The OT-DURL encoders achieved even better SI-SDR-BMs than the DURL encoders. Furthermore, the DURL and OT-DURL encoders gave better SI-SDR-RCs than most of the other models.

When comparing Table I with Table II, we can easily find that the DURL and OT-DURL encoders could achieve much higher separation performance than other encoders with a smaller number of parameters. For example, DURL-1/2/3 and OT-DURL-1/2/3 for $C = 400$ had much lower FLOPs and Param than most other encoders of $C = 800$ and $C = 1600$, but provided better separation performance. These results suggest that the DURl and OT-DURL encoders are competitive with other encoders.

3) Statistical analysis: We examined whether different encoders tend to distinguish the latent representations of vocal signal and acc signal via statistical analysis.

We first computed the l_1 norm distance between the vocal representation and acc representation for different encoders. To do this, we cut the clean vocal and acc sources of all 50 testing songs in MUSDB18 into pairs of 1-second clips. The clips in each pair were denoted as $\mathbf{s}_i \in \mathbb{R}^N$ $(i = 1, 2)$ for vocal signal and acc signal, respectively. The latent representation of $\mathbf{s}_i$ was obtained by $\boldsymbol{Z}_i = \text{Encoder}(\mathbf{s}_i) \in \mathbb{R}^{CM}$, where the encoder can be the STFT-based encoder, baseline encoder, DURL encoder, and OT-DURL encoder.

We then computed the l_1 norm distance $(\|\boldsymbol{Z}_1 - \boldsymbol{Z}_2\|_1)$ between vocal representation and acc representation for all the clip pairs. The distance histograms for different encoders are shown in Fig. 9. The distance between the vocal representation and acc representation produced with the DURL and OT-DURL encoders were much larger than those produced with the baseline encoder and STFT-based encoder, which means that our encoders are more effective in discriminating the features of vocal and acc signals.

The above l_1 norm computes the distance between vocal representation and acc representation as two 1D vectors $\boldsymbol{Z}_i \in \mathbb{R}^{CM}$ $(i = 1, 2)$. Since $\boldsymbol{Z}_i$ includes 2D intricate time-frequency information, we also examined the difference between $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$ as 2D matrices.

We rewrote $\boldsymbol{Z}_i$ as $\tilde{\boldsymbol{Z}}_i = [\mathbf{y}_i^1, \ldots, \mathbf{y}_i^j, \ldots, \mathbf{y}_i^M] \in \mathbb{R}^{C \times M}$

TABLE III: Discrimination ability of different encoders measured by coding-rate reduction of MCR^2 [54] (in $\Delta R(\tilde{\boldsymbol{Z}})$).

Model	$\Delta R(\tilde{\boldsymbol{Z}})$								
	median	mean	std	median	mean	std	median	mean	std
STFT	64.59	65.08	**22.51**	64.59	65.08	**22.51**	64.59	65.08	**22.51**
	C=400			**C=800**			**C=1600**		
Baseline	129.61	128.47	32.54	201.80	198.47	50.93	276.12	271.12	63.41
DURL-1	129.88	129.73	28.05	206.43	203.53	46.46	282.58	278.23	57.95
DURL-2	135.91	134.62	26.79	218.92	215.11	44.25	293.98	288.54	53.62
DURL-3	142.36	139.95	25.85	241.36	236.08	44.75	312.72	306.23	52.87
OT-DURL-1	148.11	145.85	26.59	269.06	263.12	40.15	358.37	350.59	51.92
OT-DURL-2	**179.55**	**175.17**	23.25	**328.81**	**320.27**	35.46	**414.90**	**405.06**	40.41
OT-DURL-3	168.51	165.09	22.75	297.51	289.26	32.35	360.69	350.81	39.35

($i = 1, 2$), where $\tilde{\boldsymbol{Z}}_i$ consists of M column (frame) with each column having C features, i.e., $\mathbf{y}_i^j \in \mathbb{R}^C$. Since the vocal and acc signals usually have different time-frequency features, we assumed that the frames of vocal and acc representations $\tilde{\boldsymbol{Z}}_i$ ($i = 1, 2$) belong to different low-dimensional linear subspaces. We used the maximal coding-rate reduction (MCR^2) principle [54]–[56] and computed the coding-rate reduction $\Delta R(\tilde{\boldsymbol{Z}})$ to measure the discrimination of vocal and acc representations [54]–[56], where $\tilde{\boldsymbol{Z}} = \tilde{\boldsymbol{Z}}_1 \cup \tilde{\boldsymbol{Z}}_2$. A higher ΔR means better discrimination between vocal and acc representations.

Table III lists the median/mean/standard deviation (std) of ΔRs for all encoders. The vocal and acc representations produced with the baseline encoder had higher ΔR than those of the STFT-based encoder. The DURL encoders had higher ΔR than the baseline encoder and the OT-DURL encoders obtained even higher ΔR. This suggests that the DURL and OT-DURL encoders have better discrimination ability than the baseline encoder and STFT-based encoder. In addition, ΔR increased with increasing C for both DURL and OT-DURL encoders. This suggests that high feature resolution is helpful in discriminating the features of the vocal and acc signals. These results were consistent with those in previous experiments.

B. Comparative evaluations in supervised UMX framework

We evaluated the DURL and OT-DURL encoders in the supervised Open-Unmix (UMX) framework [13]. UMX uses STFT-based encoder as the encoder, Inverse STFT as the decoder, and bidirectional long short-term memory as the separator. To evaluate the DURL and OT-DURL encoders in UMX, we replaced the STFT-based encoder in UMX with the DURL and OT-DURL encoders with the other components remaining unchanged.

The SVS performance is affected by the size of the training dataset; thus, we trained the original UMX and DURL/OT-DURL-based UMX on MUSDB18 with three cases: (1) limited dataset with 4% of the songs, (2) limited dataset with 40% of the songs, and (3) full dataset with all songs. As shown in Table IV, the DURL/OT-DURL-based UMX, which was trained with the limited datasets (4 and 40% of songs) provided considerable improvement compared with the original UMX. For the full dataset, the DURL/OT-DURL-based UMX had only slight improvement on the original UMX. A possible reason behind this phenomenon is that when there is sufficient training data, the separator can be quite powerful and the performance of SVS would be dominated by the separator. As a result, all the models in UMX framework had similar performance, as they all used the same separator.

TABLE IV: Performance of DURL/OT-DURL-based UMX, where "DURL/OT-DURL-1/2/3" stands for UMX with DURL/OT-DURL encoder of 1/2/3 layers.

Training with 4% dataset								
Model	**Vocal signal**				**Acc signal**			
	SDR	SIR	ISR	SAR	SDR	SIR	ISR	SAR
UMX	1.94	5.46	6.62	3.49	8.18	11.57	15.99	**12.15**
DURL-1	2.30	6.37	7.53	3.40	8.59	12.38	16.10	11.31
DURL-2	2.53	6.06	8.00	3.81	8.85	12.80	15.30	11.51
DURL-3	2.40	6.23	**8.39**	3.80	8.71	**13.07**	15.08	11.13
OT-DURL-1	2.61	6.18	8.31	4.01	8.63	12.88	15.41	11.50
OT-DURL-2	2.46	6.14	6.56	3.60	8.75	11.72	15.94	12.13
OT-DURL-3	**2.74**	**6.63**	7.32	**4.09**	**8.96**	12.26	**16.12**	12.09
Training with 40% dataset								
Model	**Vocal signal**				**Acc signal**			
	SDR	SIR	ISR	SAR	SDR	SIR	ISR	SAR
UMX	5.10	11.78	12.18	6.36	10.81	17.28	18.54	13.14
DURL-1	5.76	12.92	12.40	6.62	11.73	17.47	20.09	**13.54**
DURL-2	5.83	13.16	12.64	6.46	11.38	**17.87**	20.21	13.52
DURL-3	5.59	13.23	11.53	6.30	11.55	17.02	20.41	13.50
OT-DURL-1	5.83	12.37	**12.77**	**6.70**	11.50	17.69	19.76	13.26
OT-DURL-2	5.92	13.50	12.40	6.52	11.62	17.74	20.37	13.43
OT-DURL-3	**6.03**	**14.22**	12.08	6.54	**11.81**	17.39	**21.06**	13.50
Training with full dataset								
Model	**Vocal signal**				**Acc signal**			
	SDR	SIR	ISR	SAR	SDR	SIR	ISR	SAR
UMX	6.47	14.96	**14.28**	**6.91**	12.62	**19.82**	20.92	13.98
DURL-1	6.47	15.86	13.74	6.59	12.66	19.29	21.72	13.96
DURL-2	6.41	15.47	13.91	6.68	12.74	19.27	21.66	13.83
DURL-3	6.40	15.29	13.89	6.71	12.71	19.38	21.24	13.76
OT-DURL-1	**6.58**	**16.11**	13.59	6.78	12.81	19.69	21.41	13.94
OT-DURL-2	6.47	15.74	13.17	6.91	**12.83**	19.27	**22.31**	**14.31**
OT-DURL-3	6.57	15.30	13.37	6.85	12.82	19.35	21.54	13.91

By comparing all the results in this section, we can find that the DURL and OT-DURL encoders achieved better performance than the baseline and other encoders in the unsupervised ISVS framework. In the evaluations for the supervised setting, the DURL and OT-DURL encoders also outperformed the original UMX, especially for the datasets of limited size. These observations verified the effectiveness of the DURL and OT-DURL encoders.

VIII. Conclusions

We presented deep-unfolding-based representation learning (RL) models for singing voice separation (SVS), deep unfolded representation learning (DURL) and optimal transport (OT)-deep unfolded representation learning (OT-DURL), to learn effective latent representation of the vocal signal for SVS. In DURL, we formulated the process of RL as a series of signal-reconstruction optimization problems and penalized them with the Euclidean-distance-based synthesis prior, analysis prior, and

non-negative prior. We also extended DURL to an OT version, i.e., OT-DURL, by penalizing the analysis prior with a more suitable OT distance. The iterative algorithms derived from DURL and OT-DURL were unfolded to novel encoders, where different general priors were mapped to specific sublayers of the encoder. Experimental results indicate that the DURL and OT-DURL encoders can provide much better performance for SVS in both unsupervised and supervised frameworks than other encoders.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grants 62176182, 62072335, and 62201314, in part by the Natural Science Foundation of Shandong Province under Grant ZR2020QF007, in part by a Grant-in-Aid for Scientific Research (B) (21H03463), and in part by Fund for the Promotion of Joint International Research (Fostering Joint International Research (B)) (20KK0233).

REFERENCES

[1] Y. Li and D. Wang, "Separation of singing voice from music accompaniment for monaural recordings," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 15, no. 4, pp. 1475–1487, 2007.

[2] Z. Rafii, A. Liutkus, F. Stöter, S. I. Mimilakis, D. FitzGerald, and B. Pardo, "An overview of lead and accompaniment separation in music," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 26, no. 8, pp. 1307–1335, 2018.

[3] S. I. Mimilakis, K. Drossos, E. Cano, and G. Schuller, "Examining the mapping functions of denoising autoencoders in singing voice separation," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 28, pp. 266–278, 2020.

[4] K. Schulze-Forster, C. S. J. Doire, G. Richard, and R. Badeau, "Phoneme level lyrics alignment and text-informed singing voice separation," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 29, pp. 2382–2395, 2021.

[5] X. Ni and J. Ren, "Fc-u^2-net: A novel deep neural network for singing voice separation," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 30, pp. 489–494, 2022.

[6] S. Yuan, Z. Wang, U. Isik, R. Giri, J. Valin, M. M. Goodwin, and A. Krishnaswamy, "Improved singing voice separation with chromagram-based pitch-aware remixing," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 111–115.

[7] Z. Wang, R. Giri, U. Isik, J. Valin, and A. Krishnaswamy, "Semi-supervised singing voice separation with noisy self-training," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 31–35.

[8] W. Choi, M. Kim, J. Chung, D. Lee, and S. Jung, "Investigating u-nets with various intermediate blocks for spectrogram-based singing voice separation," in *Proc. 21th Int. Soc. Music Inf. Ret. Conf.*, 2020, pp. 192–198.

[9] Y. Li, J. Woodruff, and D. Wang, "Monaural musical sound separation based on pitch and common amplitude modulation," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 17, no. 7, pp. 1361–1371, 2009.

[10] P. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson, "Singing-voice separation from monaural recordings using robust principal component analysis," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2012, pp. 57–60.

[11] B. Zhu, W. Li, R. Li, and X. Xue, "Multi-stage non-negative matrix factorization for monaural singing voice separation," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 21, no. 10, pp. 2096–2107, 2013.

[12] Y. Hu and G. Liu, "Separation of singing voice using nonnegative matrix partial co-factorization for singer identification," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 23, no. 4, pp. 643–653, 2015.

[13] F. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, "Open-unmix - A reference implementation for music source separation," *J. Open Source Softw.*, vol. 4, no. 41, p. 1667, 2019.

[14] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: a fast and efficient music source separation tool with pre-trained models," *J. Open Source Softw.*, vol. 5, no. 50, p. 2154, 2020.

[15] S. Rouard, F. Massa, and A. Défossez, "Hybrid transformers for music source separation," *CoRR*, vol. abs/2211.08553, 2022.

[16] S. I. Mimilakis, K. Drossos, and G. Schuller, "Unsupervised interpretable representation learning for singing voice separation," in *Proc. 28th Eur. Signal Process. Conf.*, 2020, pp. 1412–1416.

[17] ——, "Revisiting representation learning for singing voice separation with sinkhorn distances," *CoRR*, vol. abs/2007.02780, 2020.

[18] K. Drossos, S. I. Mimilakis, D. Serdyuk, G. Schuller, T. Virtanen, and Y. Bengio, "Mad twinnet: Masker-denoiser architecture with twin networks for monaural sound source separation," in *Proc. Int. Joint Conf. Neural Netw.*, 2018, pp. 1–8.

[19] 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 *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 721–725.

[20] P. Magron, K. Drossos, S. I. Mimilakis, and T. Virtanen, "Reducing interference with phase recovery in dnn-based monaural singing voice separation," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 332–336.

[21] A. Défossez, N. Usunier, L. Bottou, and F. R. Bach, "Music source separation in the waveform domain," *CoRR*, vol. abs/1911.13254, 2019.

[22] D. Samuel, A. Ganeshan, and J. Naradowsky, "Meta-learning extractors for music source separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 816–820.

[23] S. Mallat, *A Wavelet Tour of Signal Processing - The Sparse Way, 3rd Edition*. Academic Press, 2009.

[24] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, 2021.

[25] E. Tzinis, S. Venkataramani, Z. Wang, Y. C. Sübakan, and P. Smaragdis, "Two-step sound source separation: Training on learned latent targets," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 31–35.

[26] Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[27] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *CoRR*, vol. abs/2012.08405, 2020.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Sys.*, 2017, pp. 5998–6008.

[29] G. Mialon, D. Chen, A. d'Aspremont, and J. Mairal, "A trainable optimal transport embedding for feature aggregation and its relationship to attention," in *Proc. 9th Int. Conf. Learn. Represent.*, 2021.

[30] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. Adv. Neural Inf. Process. Sys.*, 2015, pp. 2377–2385.

[31] D. Chen, L. Jacob, and J. Mairal, "Biological sequence modeling with convolutional kernel networks," *Bioinform.*, vol. 35, no. 18, pp. 3294–3302, 2019.

[32] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010.

[33] P. Smaragdis and S. Venkataramani, "A neural network alternative to non-negative audio models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 86–90.

[34] V. Papyan, Y. Romano, and M. Elad, "Convolutional neural networks analyzed via convolutional sparse coding," *J. Mach. Learn. Res.*, vol. 18, pp. 83:1–83:52, 2017.

[35] B. Tolooshams, S. Dey, and D. E. Ba, "Deep residual autoencoders for expectation maximization-inspired dictionary learning," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 6, pp. 2415–2429, 2021.

[36] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE Trans. Speech Audio Process.*, vol. 15, no. 3, pp. 1066–1074, 2007.

[37] G. Peyré and M. Cuturi, "Computational optimal transport," *Found. Trends Mach. Learn.*, vol. 11, no. 5-6, pp. 355–607, 2019.

[38] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, "From word embeddings to document distances," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, 2015, pp. 957–966.

[39] V. Seguy, B. B. Damodaran, R. Flamary, N. Courty, A. Rolet, and M. Blondel, "Large scale optimal transport and mapping estimation," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018.

[40] R. Flamary, C. Févotte, N. Courty, and V. Emiya, "Optimal spectral transportation with application to music transcription," in *Proc. Adv. Neural Inf. Process. Sys.*, 2016, pp. 703–711.

[41] A. Rolet and V. Seguy, "Fast optimal transport regularized projection and application to coefficient shrinkage and filtering," *Vis. Comput.*, vol. 38, no. 2, pp. 477–491, 2022.

[42] H. Sedghi, V. Gupta, and P. M. Long, "The singular values of convolutional layers," in *Proc. 7th Int. Conf. Learn. Represent.*, 2019.
[43] P. Smaragdis and S. Venkataramani, "A neural network alternative to non-negative audio models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 86–90.
[44] V. Papyan, Y. Romano, and M. Elad, "Convolutional neural networks analyzed via convolutional sparse coding," *J. Mach. Learn. Res.*, vol. 18, pp. 83:1–83:52, 2017.
[45] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.
[46] M. A. Schmitz, M. Heitz, N. Bonneel, F. M. N. Mboula, D. Coeurjolly, M. Cuturi, G. Peyré, and J. Starck, "Wasserstein dictionary learning: Optimal transport-based unsupervised nonlinear dictionary learning," *SIAM J. Imaging Sci.*, vol. 11, no. 1, pp. 643–678, 2018.
[47] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Proc. Adv. Neural Inf. Process. Sys.*, 2013, pp. 2292–2300.
[48] D. A. Lorenz and T. Pock, "An inertial forward-backward algorithm for monotone inclusions," *J. Math. Imaging Vis.*, vol. 51, no. 2, pp. 311–325, 2015.
[49] M. Cuturi and G. Peyré, "A smoothed dual approach for variational wasserstein problems," *SIAM J. Imaging Sci.*, vol. 9, no. 1, pp. 320–343, 2016.
[50] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "The MUSDB18 corpus for music separation," Dec. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.1117372
[51] I. Kavalerov, S. Wisdom, H. Erdogan, B. Patton, K. W. Wilson, J. L. Roux, and J. R. Hershey, "Universal sound separation," in *Proc. IEEE Workshop App. Signal Process. Audio Acoust.*, 2019, pp. 175–179.
[52] J. L. Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR - half-baked or well done?" in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 626–630.
[53] B. Kadioglu, M. Horgan, X. Liu, J. Pons, D. Darcy, and V. Kumar, "An empirical study of conv-tasnet," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7264–7268.
[54] Y. Yu, K. H. R. Chan, C. You, C. Song, and Y. Ma, "Learning diverse and discriminative representations via the principle of maximal coding rate reduction," in *Proc. Adv. Neural Inf. Process. Sys.*, 2020, pp. 9422–9434.
[55] K. H. R. Chan, Y. Yu, C. You, H. Qi, J. Wright, and Y. Ma, "Redunet: A white-box deep network from the principle of maximizing rate reduction," *J. Mach. Learn. Res.*, vol. 23, no. 114, pp. 1–103, 2022.
[56] Y. Ma, H. Derksen, W. Hong, and J. Wright, "Segmentation of multivariate mixed data via lossy data coding and compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 9, pp. 1546–1562, 2007.

Weitao Yuan received the B.S. and M.S. degrees from the School of Information Science and Engineering, Yanshan University, China, in 2000 and 2003, respectively. He received the Ph.D. degree from the School of Mathematical Sciences, Peking University (PKU), China, in 2008. He worked with Jinan University and Yanshan University from 2009 to 2014. He worked with the School of Computer Science and Technology, Tiangong University, from 2015 to 2021. He has been worked with the School of Software, Tiangong University, since 2022. His current research interests are convex analysis, deep learning, and music source separation.

Shengbei Wang received the B.S. and M.S. degrees from Tianjin Polytechnic University, China, in 2009 and 2012, respectively, both in signal processing. She received the Ph.D. degree in information science from the Japan Advanced Institute of Science and Technology (JAIST), Japan, in 2015. She has been worked with the School of Computer Science and Technology and the School of Software, Tiangong University, since 2016, and she is currently an Associate Professor with the School of Software. Her main research interests are signal processing, digital audio/speech watermarking, and deep learning-based source separation.

Jianming Wang received the Ph.D. degree from the School of Electrical Engineering, Tianjin University, China. From 2015 to 2020, he was the dean of the School of Computer Science and Technology in Tiangong University, China. He is currently a Full Professor and the vice-director of Engineering Faculty in Tiangong University. His research interests include computer vision and pattern recognition, robotics, and electrical tomography.

Masashi Unoki (Member, IEEE) received the M.S. and Ph.D. degrees from the Japan Advanced Institute of Science and Technology (JAIST), in 1996 and 1999. His main research interests are in auditory motivated signal processing and the modeling of auditory systems. From 1998 to 2001, he was a Research Fellow with Japan Society for the Promotion of Science (JSPS). He was associated with the ATR Human Information Processing Laboratories as a Visiting Researcher from 1999 to 2000, and he was a Visiting Research Associate with the Centre for the Neural Basis of Hearing (CNBH) in the Department of Physiology at the University of Cambridge from 2000 to 2001. He has been on the faculty of the School of Information Science at JAIST since 2001 and is now a Full Professor. He is a member of the Research Institute of Signal Processing (RISP), the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan, and the Acoustical Society of America (ASA). He is also a member of the Acoustical Society of Japan (ASJ), and the International Speech Communication Association (ISCA). Dr. Unoki received the Sato Prize from the ASJ in 1999, 2010, and 2013 for an Outstanding Paper and the Yamashita Taro "Young Researcher" Prize from the Yamashita Taro Research Foundation in 2005.

Wenwu Wang (M'02–SM'11) was born in Anhui, China. He received the B.Sc. degree in 1997, the M.E. degree in 2000, and the Ph.D. degree in 2002, all from Harbin Engineering University, China. He then worked in King's College London, Cardiff University, Tao Group Ltd. (now Antix Labs Ltd.), and Creative Labs, before joining University of Surrey, UK, in May 2007, where he is currently a Professor in Signal Processing and Machine Learning, and a Co-Director of the Machine Audition Lab within the Centre for Vision Speech and Signal Processing. He is also an AI Fellow within the Surrey Institute for People Centred Artificial Intelligence.

His current research interests include blind signal processing, sparse signal processing, audio-visual signal processing, machine learning and perception, artificial intelligence, machine audition (listening), and statistical anomaly detection. He has (co)-authored over 300 publications in these areas. He is a (co-)author or (co-)recipient of over 15 awards including the 2022 IEEE SPS Young Author Best Paper Award, ICAUS 2021 Best Paper Award, DCASE 2020 Best Paper Award, DCASE 2019 and 2020 Reproducible System Award, LVA/ICA 2018 Best Student Paper Award, FSDM 2016 Best Oral Presentation, ICASSP 2019 and LVA/ICA 2010 Best Student Paper Award Nominees.

He is an Associate Editor (2020-2025) for IEEE/ACM Transactions on Audio Speech and Language Processing. He was a Senior Area Editor (2019-2023) and an Associate Editor (2014-2018) for IEEE Transactions on Signal Processing. He is elected Chair (2023-2024) of IEEE SPS Machine Learning for Signal Processing Technical Committee, elected Vice Chair (2022-2024) of EURASIP Technical Area Committee on Acoustic Speech and Music Signal Processing, an elected Member (2021-2023) of the IEEE Signal Processing Theory and Methods Technical Committee, and an elected Member (2019-) of the International Steering Committee of Latent Variable Analysis and Signal Separation. He was a Publication Co-Chair for ICASSP 2019, a Satellite Workshop Co-Chair for INTERSPEECH 2022 and ICASSP 2024, and a Technical/Program Committee Member of over 100 international conferences.