ONLINE SINGING VOICE SEPARATION USING A RECURRENT ONE-DIMENSIONAL U-NET TRAINED WITH DEEP FEATURE LOSSES

Clement S. J. Doire

Audionamix, Paris, France

ABSTRACT

This paper proposes an online approach to the singing voice separation problem. Based on a combination of one-dimensional convolutional layers along the frequency axis and recurrent layers to enforce temporal coherency, state-of-the-art performance is achieved. The concept of using deep features in the loss function to guide training and improve the model's performance is also investigated.

Index Terms— source separation, online, convolutional neural networks, deep feature losses

1. INTRODUCTION

Audio source separation is a key research topic for both the music information retrieval and speech processing communities. With applications ranging from automatic speech recognition to automatic music transcription or lyrics alignment, it has attracted a lot of attention in recent years.

Algorithms based on Non-negative Matrix Factorization (NMF) have been widely used in source separation [1]. Although interesting results can be achieved by such methods, for instance when the separation is informed [2], algorithms based on deep neural networks have been found to deliver better performance [3].

Fully-connected feed-forward networks have been applied successfully to source separation by predicting the separated sources spectra one frame at a time, with the neighbouring frames stacked together to account for temporal context [3]. Recurrent Neural Networks (RNNs) and more specifically Long-Short-Term-Memory (LSTM) [4] networks have been proposed to better model long term temporal dependencies, leading to significantly better results in source separation [3, 5]. In [6], a skip-filtering connection is used in order to predict a time-frequency soft mask while still using the target spectrogram in the computation of the objective function, thus removing the need to explicitly define the ideal target mask.

Convolutional Neural Networks (CNNs) have been used extensively in computer vision [7] and have been employed recently to tackle the source separation task [8, 9, 10, 11]. They all present an encoder-decoder structure where, from the input spectrogram, the encoder takes care of successively computing feature maps and downsampling them, while the decoder progressively upsamples the feature maps back to the dimensionality of the original space. The advantage of using such a bottleneck structure is that the receptive field of the computed feature maps increases with depth, so the layers can learn meaningful features at different time and frequency scales. In the U-Net [9], feature maps at the output of convolutional layers on the downsampling side are concatenated with the feature maps at the same resolution on the upsampling side, in order to allow more layer interactions and reuse features previously computed without loss of information from the successive downsampling operations. In the MMDenseNet [10], a multi-band structure is proposed where each frequency band is processed by a separate CNN along with a full-band one before combining the results. This work is extended in [12] where the MMDenseNet architecture is combined with LSTM blocks. A one-dimensional adaptation of the U-Net architecture working directly on the time-domain samples was also proposed in [13]. The concept of improving network performance by using deep features in the loss function was investigated previously in the context of end-to-end speech denoising in [14].

In this paper, we present an online method for singing voice separation based on a convolutional and recurrent neural network architecture, the Online Recurrent U-Net (OR-U-Net). By using one-dimensional convolutional layers along the frequency axis together with Gated Recurrent Unit (GRU) layers [15] to enforce temporal coherency, we are able to separate the input mixture spectrogram in an online manner. To further improve the performance of the system, we train a separate CNN to be used as a loss network [16], and incorporate the computed deep features in the objective function.

The paper is organized as follows. The singing voice separation problem and the basic notation are detailed in Sec. 2. The proposed neural network architecture is described in Sec. 3 and training using deep feature losses is explained in Sec. 4. Experimental results are shown in Sec. 5 before concluding in Sec. 6.

2. PROBLEM STATEMENT

In the present work, a Short Time Fourier Transform (STFT) is applied to each channel of the stereo mixture signal and

the processing is done in the time-frequency domain. Let the singing voice, background music, and input mixture stereo magnitude spectrograms at time-frame t be given respectively by $\mathbf{V}_t$, $\mathbf{B}_t$, $\mathbf{X}_t$. We define the task of online singing voice separation as finding the time-varying filters $\mathbf{M}_{V_t}$ and $\mathbf{M}_{B_t}$, also called masks, so that:

$$\hat{\mathbf{V}}_t = \mathbf{M}_{V_t} \odot \mathbf{X}_t \tag{1}$$

$$\hat{\mathbf{B}}_t = \mathbf{M}_{B_t} \odot \mathbf{X}_t \tag{2}$$

where $\odot$ is the Hadamard product and estimates are denoted by $\hat{}$. The time-domain signals of the separated sources are then reconstructed by applying the inverse STFT with the estimated magnitude spectra and the phase of the input mixture.

We aim to train a single neural network to compute the two time-varying masks, so that $(\mathbf{M}_{V_t}, \mathbf{M}_{B_t}) = f(\mathbf{X}_t, \mathbf{h}_{t-1})$ where $\mathbf{h}_{t-1}$ contains information about the past time-frames, akin to a hidden state.

3. ONLINE RECURRENT U-NET

In the following, the U-Net architecture described in [9] is modified and adapted to the task of online singing voice separation. In the aforementioned architecture, two-dimensional convolutional layers are applied to the input spectrogram directly. In our case, to be able to perform the source separation in an online manner, we use a similar encoder-decoder network architecture to the U-Net but with one-dimensional convolutional layers computing feature maps along the frequency axis only. To enforce temporal coherency between successive time-frames, we use recurrent layers in the bottleneck. The resulting network therefore outputs a mask at time-frame t only as a function of the input mixture spectrogram at time-frame t and the hidden states of the recurrent layers, in accordance with the problem statement of Sec. 2. In order to avoid defining explicitly what the target masks should be during training, we define the spectrogram reconstruction loss function as [6]

$$\begin{aligned}\mathcal{L}_R &= \frac{1}{T}\sum_{t=1}^{T}\left(\left\|\mathbf{V}_t - \hat{\mathbf{V}}_t\right\|_1 + \left\|\mathbf{B}_t - \hat{\mathbf{B}}_t\right\|_1\right) \\ &= \frac{1}{T}\sum_{t=1}^{T}\left(\|\mathbf{V}_t - \mathbf{M}_{V_t}\odot\mathbf{X}_t\|_1 + \|\mathbf{B}_t - \mathbf{M}_{B_t}\odot\mathbf{X}_t\|_1\right)\end{aligned} \tag{3}$$

where T is the number of time-frames in the sequences used during training and the L_1 norm is computed as the mean absolute value of the array elements.

To train our network, we use stereo audio data sampled at 44.1 kHz. The STFT is computed using a window size of 2048 samples and a hop size of 512 samples. Patches of 128 time-frames are extracted, resulting in an input array of size $(128, 1025, 2)$ to the network. The input magnitude spectrograms are standardized according to the training set statistics, i.e. each frequency bin is zero-centered and scaled according to the mean and standard deviation computed over the whole training dataset.

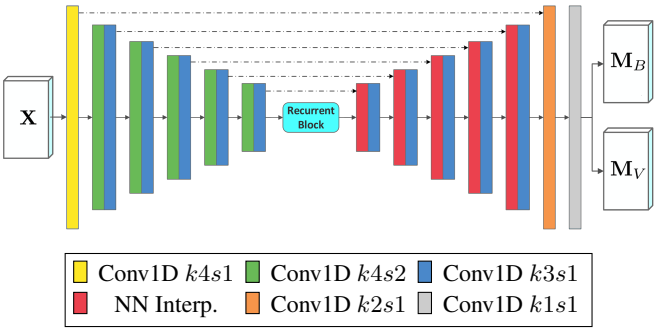

Fig. 1. Structure of the OR-U-Net neural network architecture proposed in this paper. k and s denote respectively the kernel size and stride of the convolution. Dashed lines indicate a concatenation operation.

3.1. Architecture details

The network architecture is detailed in Fig. 1. The implementation starts with a zero-padded unit-stride 1D convolution layer with a kernel size of 4, which adapts the number of frequency bins from 1025 to 1024. After that, the encoder consists in alternating a downsampling layer and a convolutional layer as in [11], although we use strided 1D convolutions with a kernel size of 4 and a stride of 2 instead of max-pooling. The feature maps computed at the output of the unit-stride convolutional layer at each frequency resolution are concatenated with the feature maps at the same resolution on the decoder side. The number of computed feature maps is doubled at each downsampling layer, to reach a maximum of 128.

At the bottom of the encoder, the last convolutional layer outputs a reduced number of feature maps (e.g. 16) so that we can flatten the feature maps computed at each time frame into a single vector. The flattened feature maps are fed to 2 GRU layers whose hidden layers are zero-initialized at the start of the sequence. The resulting vectors at each time-frame are then reshaped into feature maps at the appropriate frequency resolution before being passed on to the decoder, as shown in Fig. 2.

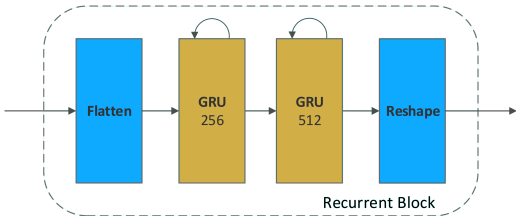

Fig. 2. Structure of the recurrent block at the bottleneck of the proposed model.

Each layer of the decoder is an upsampling stage consisting in an interpolation followed by a convolution, as described

in Sec. 3.2. After having upsampled the frequency resolution to 1024 bins, a zero-padded unit-stride convolution layer with a kernel size of 2 is used to recover the original frequency resolution. The rectified linear unit (ReLU) activation function is used after every layer in both the encoder and decoder. Finally, the network ends with a feature map averaging layer followed by a sigmoid activation to restrict the values of the masks to the range $[0, 1]$.

3.2. Upsampling and checkerboard artifacts

To upsample the feature maps on the decoder side and recover the original dimensionality, many networks use transposed convolutional layers [9, 10]. These layers are based on the idea that the convolution operation's backward pass is the transposed operation of the forward pass, allowing to go from the reduced dimensional space to the original space [17].

According to [18], using a transposed convolution layer can lead to strong checkerboard artifacts in image generation networks, as the kernel passes over some indexes more times than others in a periodic way. To counter these artifacts, they suggest using a simple interpolation technique followed by a standard convolutional layer that keeps the dimensionality intact. Similar artifacts were observed in [13] in the case of source separation on raw audio samples. They propose a similar workaround technique, interpolation followed by a convolution operation, with the interpolation being a learned upsampling function.

The learned upsampling layer of [13] having nearest neighbour and linear interpolations as edge cases, we conducted initial tests with these two methods only. With a doubling of the number of features along the frequency axis at each upsampling layer, nearest neighbour consists in repeating each frequency bin once. This seems to indicate that nearest neighbour interpolation might lead to more abrupt changes in the frequency content, a cause of musical noise, while the smoothing provided by linear interpolation might be more pleasant to the ear. In practice however, both lead to similar perceptual quality and objective evaluation metrics. Informal listening tests by professional audio engineers tended to indicate a slight preference for the nearest-neighbour interpolated versions as they exhibited less interference from the other source, hence its use in our final implementation.

4. TRAINING WITH DEEP FEATURE LOSSES

Taking inspiration from the computer vision field [19, 16], we train our source separation network using deep feature losses. The goal is to compare the activations of different layers in a pre-trained loss network that is applied to both the predicted and target sources. In the context of magnitude spectrograms and if the loss network is a CNN with an encoder structure, it means that minimizing the differences between feature maps extracted at different depths will minimize differences in local patterns corresponding to different time and frequency scales.

4.1. Designing the loss network

In computer vision, there are standard CNN architectures pre-trained on large-scale datasets for classification tasks, such as VGG-19 [7], with the layers of such networks representing increasing levels of abstraction [19]. In the audio processing community, such standard pre-trained networks do not exist and we need to design and train the loss network.

Considering the OR-U-Net is an online architecture, the only temporal coherency enforced from one frame to the next is through the use of GRU layers at the bottleneck of the network. By training a loss network that makes use of 2D convolutions, we are able to extract both frequency patterns and temporal patterns. Combining different losses corresponding to different depths of the loss network should be able to steer the GRU layers towards learning to respect meaningful temporal patterns at different scales.

Therefore, rather than training the loss network on a classification task as in [14], we use a more straightforward approach by training the loss network on a similar task, offline singing voice separation, and the same database as the OR-U-Net. To do so, we use the U-Net architecture described in [9], with a few adaptations: the first and last layers are adapted to work with spectrograms of size $(128, 1025)$, and both the singing voice and background music masks are predicted by the same network, using loss function (3).

4.2. Designing the loss

Let $\phi_i(\mathbf{Z})$ be the feature maps at the output of the i^{th} layer of the loss network for an input $\mathbf{Z}$. We define the feature loss at the i^{th} layer as

$$\mathcal{L}_{\phi_i} = \left\| \phi_i(\mathbf{V}) - \phi_i(\hat{\mathbf{V}}) \right\|_1 + \left\| \phi_i(\mathbf{B}) - \phi_i(\hat{\mathbf{B}}) \right\|_1 \quad (4)$$

where the L_1 norm is computed as the mean absolute value of the array elements. The total loss used during training is therefore

$$\mathcal{L} = w_0 \mathcal{L}_R + \sum_{i=1}^{N} w_i \mathcal{L}_{\phi_i} \quad (5)$$

with N the total number of deep features to use in the loss function. w_0 represents the proportion of direct frequency-bin reconstruction error in the loss function. The weights w_i, for $i \geq 1$, can be chosen depending on whether the reconstruction of long term or short term temporal patterns should be emphasized when training the GRU layers.

5. EVALUATION

To evaluate our singing voice separation model, we used the standard Blind Source Separation (BSS) metrics Signal-to-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR)

and Signal-to-Artifacts Ratio (SAR) [20]. The values were computed with the mir-eval implementation [21]. To avoid strong outliers, the audio tracks were partitioned into segments of 5 seconds, with the very low energy segments being discarded.

To train our model, we used the 50 songs of the DSD100 training set [22], 50 songs from the MedleyDB database [23] and 30 songs from the CCMixter database [24]. Out of the total 130 songs, 20 were randomly assigned to the validation set. All models are evaluated on the 50 songs of the test set of DSD100.

5.1. Implementation Details

We compare two versions of the proposed method: OR-U-Net, the network architecture described in Sec. 3 trained using the L_1 cost function (3), and OR-U-Net$_{df}$, the same architecture trained using the deep feature losses cost function (5) with $N = 4$ deep features and the decreasing weight strategy $w_i = N - i + 1$, normalized so that $\sum_{i=0}^{N} w_i = 1$. We evaluate them against two competing methods: the adapted U-Net architecture used as a feature loss network to train OR-U-Net$_{df}$ (see Sec. 4.1), and an LSTM network inspired by [5] with 3 LSTM layers of 256 units and a final fully-connected layer. Contrary to [5], the LSTM layers are not bi-directional in order to use this network as a baseline for online singing voice separation. Both of these competing methods are trained to output the two stereo masks directly using the L_1 cost function of (3). All four methods are evaluated using the masks directly at the output of each network, without any post-processing applied.

During training, for all four methods, we use the RMSprop optimizer with a learning rate of 0.0001 and a batch size of 10. Training is stopped when no improvement is observed on the validation set for 30 epochs and the model with the best validation loss is selected. Data augmentation is used, with random scaling, panning, low-pass/high-pass filtering and eventual reverberation of the individual sources before summing them to build the input mixture.

5.2. Results

Median BSS evaluation metrics measured on the test set of the DSD100 database are presented in Table 1. The proposed OR-U-Net model trained without deep feature losses achieves performance on a par with the U-Net. The higher vocals and background SIR scores indicate a stronger separation with less interference is achieved through the U-Net, but the OR-U-Net achieves best vocals SAR performance overall and exhibits similar SDR scores to the U-Net. It is interesting to note that the OR-U-Net achieves stronger performance than the baseline LSTM online model on all criteria.

The OR-U-Net$_{df}$ model achieves best SDR and SIR overall performance on the vocals separation task, with scores significantly higher than both the base OR-U-Net model and the U-Net loss network used for its training. However, with a strong SAR but low SIR score, the OR-U-Net$_{df}$ model shows more mitigated performance on the background separation task.

These results suggest that even when using a loss network with limited capacity trained on a similar task with the same database, it is indeed possible to significantly change the learning behaviour of a model, making the deep feature losses technique very promising. In this case, with the decreasing weight strategy used in the loss function, we believe that by enforcing the correct reconstruction of local temporal patterns the GRU layers learned weights that are more efficient at handling short-term temporal context.

	Vocals			Background		
Method	SDR	SIR	SAR	SDR	SIR	SAR
LSTM [5]	2.83	6.89	6.02	9.48	12.39	12.99
U-Net [9]	3.21	8.34	5.86	9.81	**13.42**	12.98
OR-U-Net	3.14	7.41	**6.20**	**9.87**	13.25	12.99
OR-U-Net$_{df}$	**3.70**	**9.52**	5.80	9.65	11.84	**14.16**

Table 1. Median BSS evaluation metrics in dB for singing voice separation on the test set of the DSD100 database. Best performance is shown in bold for each metric.

6. CONCLUSION

In this paper, we have presented a novel online approach to the singing voice separation problem. Using one-dimensional convolutional layers to form an encoder-decoder structure along the frequency axis together with GRU layers along the time axis, we achieve state-of-the-art performance compared to a CNN architecture using 2D convolutions working jointly on the time and frequency axes. Furthermore, we showed that even when using a loss network with limited capacity, deep feature losses can be used to improve the model's performance significantly.

7. ACKNOWLEDGEMENTS

The author would like to thank Clement Godard for the many fruitful discussions and helpful suggestions.

8. REFERENCES

[1] C. Févotte, E. Vincent, and A. Ozerov, *Audio Source Separation*, chapter Single-channel audio source separation with NMF: divergences, constraints and algorithms, Springer, 2018.

[2] R. Hennequin, J. J. Burred, S. Maller, and P. Leveau, "Speech-guided source separation using a pitch-adaptive guide signal model," *IEEE International Con-*

ference on Acoustics, Speech and Signal Processing (ICASSP), 2014.

[3] F. Weninger, J. R. Hershey, J. Le Roux, and B. Schuller, "Discriminatively trained recurrent neural networks for single-channel speech separation," *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014.

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

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

[6] S. I. Mimilakis, K. Drossos, J. Santos, G. Schuller, T. Virtanen, and Y. Bengio, "Monaural singing voice separation with skip-filtering connections and recurrent inference of time-frequency mask," *arXiv preprint arXiv:1711.01437v2*, 2018.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556v6*, 2014.

[8] P. Chandna, M. Miron, J. Janer, and E. Gómez, "Monoaural audio source separation using deep convolutional neural networks," *International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, 2017.

[9] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, "Singing voice separation with deep u-net convolutional networks," *18^{th} International Society for Music Information Retrieval (ISMIR) Conference*, 2017.

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

[11] S. Park, T. Kim, K. Lee, and N. Kwak, "Music source separation using stacked hourglass networks," *19^{th} International Society for Music Information Retrieval (ISMIR) Conference*, 2018.

[12] N. Takahashi, N. Goswami, and Y. Mitsufuji, "MMDenseLSTM: an efficient combination of convolutional and recurrent neural networks for audio source separation," *arXiv preprint arXiv:1805.02410v2*, 2018.

[13] D. Stoller, S. Ewert, and S. Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," *19^{th} International Society for Music Information Retrieval (ISMIR) Conference*, 2018.

[14] F. G. Germain, Q. Chen, and V. Koltun, "Speech denoising with deep feature losses," *arXiv preprint arXiv:1806.10522v2*, 2018.

[15] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoderdecoder for statistical machine translation," *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

[16] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[17] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[18] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016.

[19] Q. Chen and V. Koltun, "Photographic image synthesis with cascaded refinement networks," *IEEE International Conference on Computer Vision (ICCV)*, 2016.

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

[21] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, "mir-eval: A transparent implementation of common MIR metrics," *15^{th} International Society for Music Information Retrieval (ISMIR) Conference*, 2014.

[22] A. Liutkus, F.-R. Stöter, Z. Rafii, D. Kitamura, B. Rivet, N. Ito, N. Ono, and J. Fontecave, "The 2016 signal separation evaluation campaign," *International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, pp. 323–332, 2015.

[23] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "MedleyDB: A multitrack dataset for annotation-intensive MIR research," *15^{th} International Society for Music Information Retrieval (ISMIR) Conference*, 2014.

[24] A. Liutkus, D. Fitzgerald, and Z. Rafii, "Scalable audio separation with light kernel additive modelling," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.