Adaptive Front-ends for End-to-end Source Separation

Shrikant Venkataramani, Jonah Casebeer
University of Illinois at Urbana-Champaign
svnktrm2, jonahmc2@illinois.edu

Paris Smaragdis*
University of Illinois at Urbana Champaign
Adobe Research

Abstract

Source separation and other audio applications have traditionally relied on the use of short-time Fourier transforms as a front-end frequency domain representation step. We present an auto-encoder neural network that can act as an equivalent to short-time front-end transforms. We demonstrate the ability of the network to learn optimal, real-valued basis functions directly from the raw waveform of a signal and further show how it can be used as an adaptive front-end for end-to-end supervised source separation.

1 Introduction

Several neural network (NN) architectures and methods have been proposed for supervised single-channel source separation [1, 2, 3, 4]. These approaches can be grouped together into a common two-step workflow as follows. The first step of the separation procedure is to transform the time domain signals into a suitable time-frequency (TF) representation using short-time Fourier transforms (STFTs). The separation procedure forms the second step of the work-flow and is used to separate the source from the interfering signals in the mixture. One approach to solve the separation problem is to train a suitable neural network (NN) to estimate the contribution of the source in the mixture spectrogram [5, 6, 7]. This estimate is then multiplied by the mixture phase and transformed into the time domain by an overlap and add inverse STFT operation. Figure 1 gives the block diagram of such a system using STFT as the front-end transform.

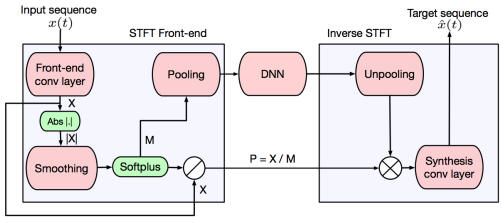

Figure 1: Joint block diagram of generalized NN based source separation system using (i) STFT (outer blue) and (ii) Proposed adaptive front-end transform (inner red and green).

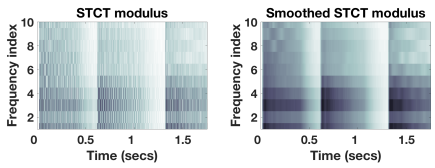

Figure 2: Modulus of Short-time cosine transform (STCT) and smoothed STCT coefficients (first 10 coefficients) for a sequence of piano notes. The STCT coefficients exhibit variations and need to be averaged across time by applying a suitable smoothing filter.

These NN based source separation approaches suffer from the following drawbacks: (i) We restrict the processing to magnitude spectrograms of the training and mixture signals. Consequently, these approaches do not provide a reliable way to deal with complex numbers. This can be alleviated by

*This work was supported by NSF grant 1453104.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

using a real front-end transform like the discrete-cosine transform (DCT). (ii) More importantly, the unavailability of a NN equivalent to these front-end transforms hinders the development of end-to-end separation systems that operate directly on the mixture and source waveforms. In this paper, we investigate the use of alternative adaptive front-end transforms for supervised source separation.

2 Auto-encoder Transforms for Source separation

Given a time domain sequence x, the short-time transform operation of x can be expressed as a generalized equation given by,

$$\mathbf{X}_{nk} = \sum_{t=0}^{N-1} x(nh+t) \cdot w(t) \cdot b(k,t) \tag{1}$$

Here, $\mathbf{X}_{nk}$ represents the energy of the k^{th} component in the n^{th} frame, N represents the size of the low pass window function w and h represents the hop size of the short-time transform. The functions $b(k,t)$ form the basis functions of the transformation. Recently, Sainath et.al [8], and Dieleman and Schrauwen [9] have proposed the use of a convolutional layer as an alternative to front-end STFTs. In this section, we expand upon the premise and develop a real valued, convolutional auto-encoder transform (AET) that can be used as an alternative to front-end short-time transforms. The encoder part of the auto-encoder (AE) acts as the analysis filter bank and produces the transformed (spectrogram like) version of the input. The decoder performs the synthesis step to reconstruct the time domain signal.

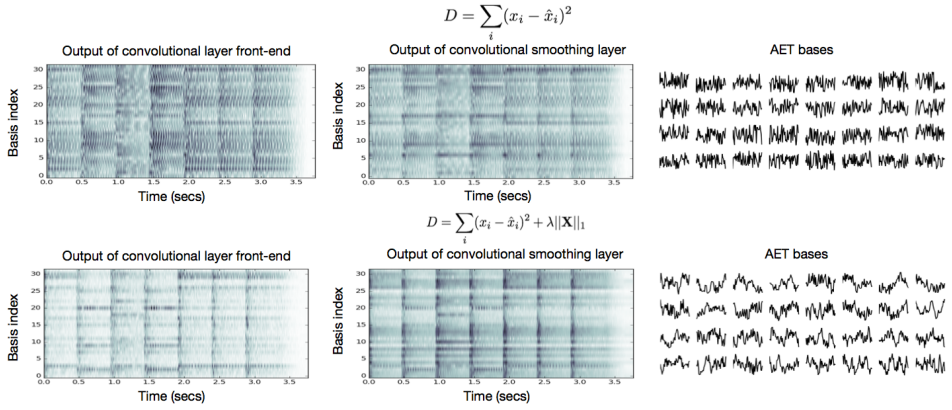

Figure 3: Outputs of front-end and smoothing convolutional layers and corresponding basis functions for AET representation of a piano audio snippet with and without sparsity constraints. The AET basis functions appear like small snippets of the input waveforms when sparsity constraints are imposed.

2.1 Analysis Encoder

Assuming a unit sample hop size, we can interpret (1) as a filtering operation,

$$\mathbf{X}_{nk} = \sum_{t=0}^{N-1} x(n+t) \cdot \mathbf{F}(k,t) \tag{2}$$

Thus, we may replace the front-end transform by a convolutional neural network (CNN) such that the k^{th} filter of the CNN represents the k^{th} row of $\mathbf{F}$. The output of the CNN gives the TF representation of the input signal with a unit hop size. In figure 2, we see that the spectral energies do not maintain locality [10] and exhibit a modulation that could be dependent on the frequency, the window size and the hop size parameters, when using real transforms like the short-time cosine transform. Thus, we need a suitable temporal smoothing operation when using real transforms. This smoothing step can also be interpreted as a CNN applied on $|\mathbf{X}|$. However, since there are no non-negativity constraints applied on the smoothing filters, the elements of the smoothed spectrogram $\mathbf{M}$ can potentially assume negative values. To avoid this solution, we include a non-linearity $g : \mathbb{R} \rightarrow \mathbb{R}^+$, a mapping from

the space of real numbers to the space of positive real numbers. The output of this layer $\mathbf{M}$ can be interpreted as the magnitude spectrogram of the input signal and $\mathbf{P} = \frac{\mathbf{X}}{\mathbf{M}}$ can be interpreted as the corresponding phase. The phase component captures the high-frequency variations in the coefficients which cannot be modeled in the smoothed spectrogram. In order to arrive at a subsampled TF representation, we apply a max-pooling layer that replaces a pool of h frames by a single frame. Note that all the convolution and pooling operations are one-dimensional i.e., the operations are applied across the time-axis only.

2.2 Synthesis Decoder

The next step is to synthesize the signal back into the time domain. This can be achieved by inverting the operations performed by the analysis encoder while computing the forward transform. The first step of the synthesis procedure is to undo the effect of the lossy pooling layer. We use an upsampling operator by inserting as many zeros between the frames as the pooling duration as proposed by Zieler et.al., [11]. The unpooled magnitude spectrogram is then multiplied by the phase using an element-wise multiplication to give an approximation $\hat{\mathbf{X}}$ to the matrix $\mathbf{X}$. We invert the operation of the first transform layer by a convolutional layer to implement the deconvolution operation. This convolutional layer thus, performs the interpolation between the samples. Essentially, the output of the analysis encoder gives the weights of each basis function in representing the time domain signal. The synthesis layer reconstructs each component by adding multiple shifted versions of the basis functions at appropriate locations in time. This inversion procedure is similar to the filterbank-summation technique of inverting the subsampled STFT representation of a sequence [12]. The weights (filters) of the final convolutional layer give the AET basis functions (see figure 3).

2.3 Visualizing Adaptive Front-ends

Having developed the convolutional auto-encoder for AETs, we can now understand the nature of the basis functions and the spectrograms obtained. Figure 3 illustrates these on a simple piano snippet consisting of 3 distinct notes. We apply a 32-dimensional AET transformation i.e., the front-end convolutional layer is assumed to have 32 filters. The filter size and hop are chosen to be 1024 samples and 16 samples respectively. The entire time domain sequence x is given as an input to the network in a single batch and the parameters of the network are updated according to the RMSProp algorithm [13]. We apply a softplus non-linearity to the convolutional smoothing layer. To train the auto-encoder, we minimize the mean squared error between the input sequence and its reconstruction $\hat{x}$. The output of the front-end convolutional layer, the output of the convolutional smoothing layer and the corresponding AET basis functions are shown in figure 3 (top). Imposing additional sparsity constraints on the output of the front-end convolutional layer $\mathbf{X}$ allows the network to learn a sparser spectrogram-like representation, as shown in figure 3 (bottom). Thus, the AET learns adaptive basis functions tailored to the input waveform in trying to represent the input sequence.

2.4 End-to-end Source Separation

Figure 1 shows the application of AET analysis and synthesis layers for end-to-end supervised source separation. The forward and inverse transforms can be directly replaced by the AET analysis and synthesis networks in a straightforward manner. In our experiments, the separation network consisted of a cascade of 3 dense layers with 512 hidden units each, each followed by a softplus non-linearity. We train the network by giving the mixture waveform at the input and minimize the mean squared error between the network output and the clean waveform of the source. Thus, the network learns to estimate the contribution of the source given the raw waveform of the mixture.

3 Experiments

We evaluate the separation performance for three types of front-ends: STFT, AET and orthogonal-AET for source separation. We compare the results based on the BSS_EVAL metrics. [14]. For training the neural networks, we randomly selected 10 male-female speaker pairs from the TIMIT database [15]. In the database, each speaker has a total of 10 recorded sentences. For each pair, we mix the sentences at 0 dB. This gives 10 mixture sentences per pair and a total of 100 mixture sentences overall. For each pair, we train on 8 sentences and test on the remaining two. Thus, the

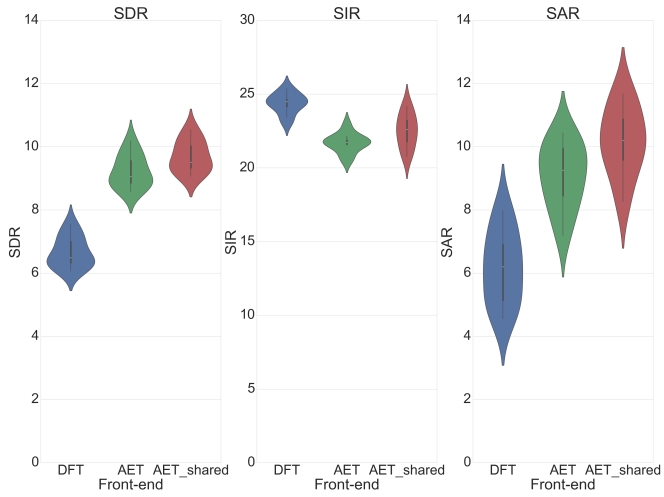

Figure 4: Comparison of source separation performance on 20 speech on speech mixtures in terms of BSS_EVAL parameters. We compare the separation performance for multiple front end transforms viz., STFT, AET and orthogonal AET. The embedded boxplot (thicker black line) indicates the inter-quartile range. We see that opting for an adaptive front-end results in a significant improvement in source separation performance over STFT front-ends.

network is trained on 80 mixture sentences and evaluated on the remaining 20 mixture sentences. We use a batch size of 16 and a dropout-rate of 0.2. All the transforms were computed at a window-size of 1024 samples at a hop of 16 samples. The smoothing layer was selected to have a length of 5 frames. We used a cost function that directly optimizes the signal to distortion ratio (SDR) instead. For a reference signal y and an output x we would maximize:

$$\max \ \text{SDR}(x,y) \ = \ \max \frac{\langle xy \rangle^2}{\langle yy \rangle \ \langle xx \rangle - \langle xy \rangle^2} \ \equiv \ \min \frac{\langle yy \rangle \langle xx \rangle - \langle xy \rangle^2}{\langle xy \rangle^2} \ \propto \ \min \frac{\langle xx \rangle}{\langle xy \rangle^2} \quad (3)$$

Thus, maximizing the SDR is equivalent to maximizing the correlation between x and y, while producing a minimum energy output.

3.1 Results and Discussion

The corresponding violin plots that show the distribution of the BSS_EVAL metrics from our experiments are shown in figure 4. We see that the use of AETs improves the separation performance in terms of of all metrics compared to an STFT front-end. We additionally see that when using the orthogonal AET we obtain additional performance gains, overall in the range of 2dB for SDR, 5dB in SIR and 3dB in SAR. One possible reason for the increased performance of the orthogonal AET could be the reduction in the number of trainable parameters caused by forcing the synthesis transform to be the transpose of the analysis transform, which in turn reduces the possibility of over-fitting to the training data. The above trends appear consistent for both the cost-functions considered.

4 Conclusion and Future Work

In this paper, we develop and investigate a convolutional auto-encoder based front-end transform that can be used as a replacement to STFTs. The adaptive front-end comprises a cascade of three layers viz., a convolutional front-end transform layer, a convolutional smoothing layer and a pooling layer. We have shown that AETs are capable of automatically learning adaptive basis functions and discovering data-specific frequency domain representations directly from the raw waveform of the data. Finally, the use of AETs allows us to interpret source separation and possibly other applications as an end-to-end neural network capable of outperforming current approaches that rely on fixed front-ends. In future work, we plan to investigate alternative neural network architectures and unpooling strategies to propose multi-layer front-end transforms.

References

[1] P. Smaragdis and S. Venkataramani, "A neural network alternative to non-negative audio models," *CoRR*, vol. abs/1609.03296, 2016. [Online]. Available: http://arxiv.org/abs/1609.03296

[2] P. S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Deep learning for monaural speech separation," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 1562–1566.

[3] P. Chandna, M. Miron, J. Janer, and E. Gómez, "Monoaural audio source separation using deep convolutional neural networks," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2017, pp. 258–266.

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

[5] M. Kim and P. Smaragdis, "Adaptive denoising autoencoders: A fine-tuning scheme to learn from test mixtures," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2015, pp. 100–107.

[6] E. M. Grais, M. U. Sen, and H. Erdogan, "Deep neural networks for single channel source separation," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3734–3738.

[7] E. M. Grais and M. D. Plumbley, "Single channel audio source separation using convolutional denoising autoencoders," *arXiv preprint arXiv:1703.08019*, 2017.

[8] T. N. Sainath, R. J. Weiss, A. W. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform cldnns," in *INTERSPEECH*. ISCA, 2015, pp. 1–5.

[9] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 6964–6968.

[10] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[11] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[12] J. O. Smith, *Spectral Audio Signal Processing*. http://ccrma.stanford.edu/ jos/sasp/.

[13] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini–batch gradient descent."

[14] C. Févotte, R. Gribonval, and E. Vincent, "Bss_eval toolbox user guide–revision 2.0," 2005.

[15] J. S. Garofolo, L. F. Lamel, J. G. F. William M Fisher, D. S. Pallett, N. L. Dahlgren, and V. Zue, "Timit acoustic phonetic continuous speech corpus," Philadelphia, 1993.