

MULTI-VIEW NETWORKS FOR DENOISING OF ARBITRARY NUMBERS OF CHANNELS

Jonah Casebeer*, Brian Luc*

University of Illinois at Urbana-Champaign
jonahmc2, luc2@illinois.edu

Paris Smaragdis

University of Illinois at Urbana-Champaign
Adobe Research

ABSTRACT

We propose a set of denoising neural networks capable of operating on an arbitrary number of channels at runtime, irrespective of how many channels they were trained on. We coin the proposed models *multi-view networks* since they operate using multiple views of the same data. We explore two such architectures and show how they outperform traditional denoising models in multi-channel scenarios. Additionally, we demonstrate how multi-view networks can leverage information provided by additional recordings to make better predictions, and how they are able to generalize to a number of recordings not seen in training.

Index Terms— multichannel, denoising, deep learning

1. INTRODUCTION

Suppose you are provided with multiple noisy recordings of the same event and wish to produce a single clean recording. Historically, this problem has been addressed using microphone array techniques, which can seamlessly scale to an arbitrary number of channels if the necessary hardware is in place. Although such techniques work well, recently we have seen learning-based methods being used for such tasks because of their ability to resolve much more complex denoising problems. More specifically, we have seen a strong wave of deep learning-based methods that introduce powerful non-linear models that can include more parameters and that can learn to resolve more challenging denoising problems.

Deep learning-based denoising and source separation have been explored in a variety of settings. Liu *et al.* [1] has studied deep learning for single channel denoising through spectral masking and regression as applied independently over each spectral frame of the noisy input. Representations capable of leveraging the temporal nature of audio were introduced by Huang *et al.* in 2014 and 2015 [2, 3]. In this work the authors constructed Recurrent Neural Networks (RNNs) to leverage the strong dependency between consecutive spectral frames in single-channel audio recordings. Others have since employed similar techniques [4, 5, 6, 7, 8, 9].

Multi-channel and deep learning techniques have been combined by Swietojanski *et al.*[10] in 2013 and Araki *et al.*[11] in 2015 who both constructed multi-channel features for speech enhancement in the context of ASR. The authors of these works found that some multi-channel features could outperform conventional methods. Similarly Nugraha constructed a multi-channel framework using DNNs for estimation of the spectra of sources and an EM algorithm to combine these in to a multi-channel filter [12]. When several recordings are available these methods are able to capture new information. Li *et al.* and Xiao *et al.* both in 2016 where able to further leverage multiple channels with a time domain and frequency domain neural beamformers respectively [13, 14]. Now, with deep clustering Wang *et al.* [15] performed multi-channel speaker independent source separation.

A common problem with learning-based methods (whether based on deep learning or not), is that the training setup needs to be replicated during inference. That means that when one trains a multi-channel system to perform, e.g., 4-channel denoising, that system can only be straightforwardly deployed on a 4-channel system. That is in contrast to analytical approaches, like classical array methods, that make use of geometric information to perform their task. The price to pay for more using more powerful deep learning models is that this analytical flexibility is lost, and one can only train and deploy using very similar setups. Here, we address this problem by introducing two neural network architectures that can be trained on a different number of channels than the number of channels that they are deployed for. This allows us to train systems on, e.g., 4-channels and deploy them on an 8-channel (or 2-channel) array without having to retrain, or in any way modify the learned models.

With evidence of RNNs surpassing feedforward networks, we propose two RNN architectures to extend current multi-channel techniques. Our first solution is to construct an RNN that instead of unrolling over time, it unrolls across the number of input channels. This allows us to train and deploy this model on an arbitrary number of input channels. We subsequently propose an additional model which unrolls both across time and channels. We call these multi-view networks (MVN), because they combine multiple views of the input. We find that they are able to consistently leverage information

*These two authors contributed equally.

This work was supported by NSF grant #1319708

provided from additional recordings, as well as to generalize to a number of recordings not seen in training.

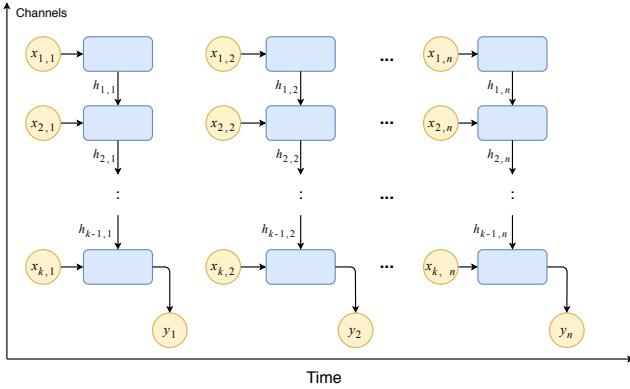


Fig. 1: 1D MVN unrolling across channels. $x_{i,j}$ represents the j th spectral frame of the i th recording. $h_{i,j}$ represents the hidden state produced by the RNN at the j th spectral frame of the i th recording. y_i is the predicted clean spectral frame.

2. MULTI-VIEW MODELS FOR DENOISING

Suppose you are provided with a noisy signal $s(t)$ which you wish to denoise. In the classic denoising RNN setup we apply a Short-Time Fourier Transform (STFT) on $s(t)$ to obtain a series of spectral magnitude frames x_i . An RNN would then unroll through these frames over time as follows:

$$\begin{aligned} h_i &= \sigma(W_h x_i + U_h h_{i-1}) \\ y_i &= \sigma(W_x h_i) \end{aligned} \quad (1)$$

and would find optimal matrices W_h and U_h to provide a set of denoised magnitude STFT frames y_i . The function σ can be any appropriate neural network activation. The unrolling scheme in the equations above takes advantage of temporal information that lies across spectral frames, and can also let us process inputs with an arbitrary length irrespective of the training data. Suppose now that you are provided with multiple noisy recordings $s_{1:k}(t)$ of the same event and wish to produce a single clean recording. We define $x_{i,j}$ to represent the j^{th} spectral frame in the i^{th} recording's STFT. If we wanted to use the aforementioned model and take advantage of the multiple channels, we could apply it on the averaged channel spectra, or on each input channel separately and then average all the outputs. Although this makes use of the multiple channels, these approaches are not particularly effective.

We propose using the RNN unrolling scheme across input recordings $x_{1:k,t}$ at every spectral frame t . This approach can allow us to take advantage of multi-channel information at each time frame, and has a number of advantages over averaging. For example, it is possible that at different points in time a different channel might provide the best input for denoising; unrolling in this fashion allows the model to leverage that instead of averaging the result with worse channels.

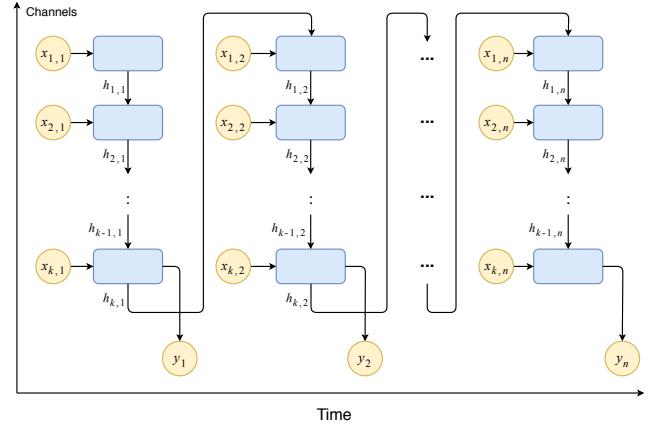


Fig. 2: 2D MVN unrolling across both channels and time, using the same notation as Fig. 1, note how the last channel's hidden state feeds into the first channel of the next time step

Additionally, this approach allows us to test on an arbitrary number of channels regardless of how many we used in training. To reconstruct the denoised spectra, we experimented with averaging each output of the RNN as it unrolls over channels, as well as taking the output after the last channel is processed. We found using the last hidden state as the base for a prediction to work best. Figure 1 demonstrates how unrolling across channels works. The obvious disadvantage of this approach is that we do not make use of temporal structure anymore. To address this problem we introduce a 2-dimensional RNN that unrolls across both time and sources. This allows a model to leverage the temporal dependency between time steps as well as the mutual information between different channels. Now, if the source of noise or clean signal moves with respect to the microphones the model can find the best recording to denoise and leverage previous spectral information about the sound. We accomplish this with the recurrence shown below:

$$\begin{aligned} h_{i,j} &= \begin{cases} \sigma(W_h x_{i,j} + U_h h_{k,j-1}) & \text{if } i = 1 \\ \sigma(W_h x_{i,j} + U_h h_{i-1,j}) & \text{otherwise} \end{cases} \\ y_j &= \sigma(W_x h_{k,j}) \end{aligned} \quad (2)$$

Note that in the case of a single input recording the 2D MVN simply unrolls across time. Figure 2 illustrates the 2D unrolling over channels and time.

Given these two different unrolling schemes we construct two different networks. First, for the 1D case. A denoising MVN is composed of a fully-connected front layer, a recurrent layer, and a fully-connected back layer. The front layer is given the magnitude STFT of the input channels, the recurrent layers Eq. (1) performs the unrolling operations across recordings, and the back layer regresses the RNNs output into the original STFT dimensions. To transform back to the time domain we use the phase STFT of the last channel. For the 2D

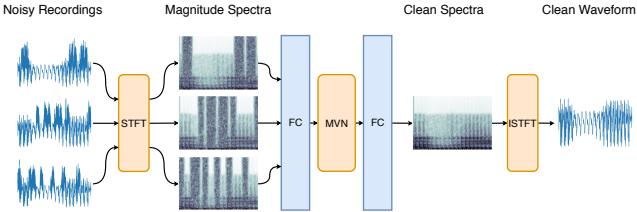


Fig. 3: Denoising pipeline using an MVN.

case, a denoising MVN is identical to a 1D MVN except for the recurrence which is defined in Eq. (2). Figure 3 broadly illustrates an MVN supplied with 3 noisy recordings containing mutual information. In practice, we used GRUs [16] instead of plain RNNs.

We find that training MVNs with an SDR-proxy loss as defined in [17] works far better than traditional norm-based loss functions. The loss used is:

$$SDRLoss(x, y) = -\frac{(x^\top y)^2}{x^\top x} \quad (3)$$

where y and x are vectors containing the target and output time-domain signals respectively.

3. EXPERIMENTS

Our system was evaluated on two different kinds of noisy mixtures. Both are created from speakers in the TIMIT data set and from segments of "Babble", "Airport", "Train", and "Subway" noises. From TIMIT we select 12 female speakers each of which has 10 unique utterances [18]. From the 120 utterances, we randomly select 100 for training and 20 for validation. From each utterance we create a two-second noisy mixture by adding one of the "Babble", "Airport", "Train" and "Subway" noises. We propose two mixing techniques and evaluate our performance with the BSS-Eval Metric: Source to Distortion Ratio (SDR) [19]. In both setups we only show results for the 2D MVN as it outperforms the 1D MVN.

3.1. Naive Averaging RNN

As a benchmark we used an averaging model. This model averages all channels then passes the averaged STFT frames through a dense layer which reduces 1024-point DFTs to a size of 512. We then unroll a GRU with a hidden size of 512. The hidden size is expanded from 512 dimensions to 1024 with a dense layer. Finally, we perform an ISTFT to produce a denoised signal. We refer to this model as the averaging RNN. This model is trained with the softplus nonlinearity.

3.2. Model Parameters

We construct a 2D MVN which operates on 1024-point DFTs, has a fully-connected layer that goes from 1024 to 512 dimensions, and then a GRU over channels with a hidden size

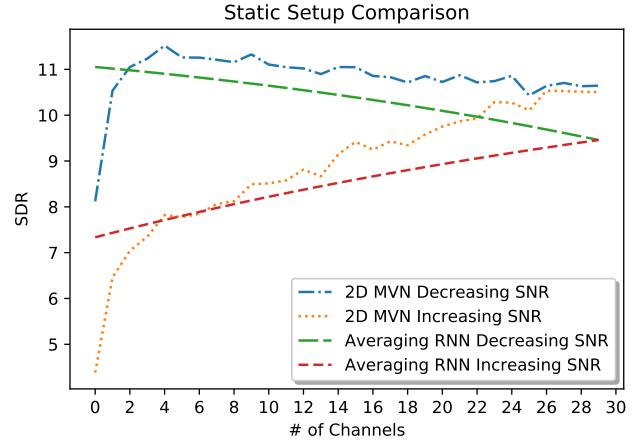


Fig. 4: Testing how 2D MVNs leverage new channel information. The x-axis on the plot denotes the number of provided noisy channels, and the y-axis shows the average SDR on the validation set. In the case of decreasing SNR the MVN obtains peak performance after a few channels and then largely retains it, not being influenced by the latter poor-quality input channels. With increasing SNR, the MVN keeps performing better as additional cleaner channels are being received. Ultimately, the MVN performs the same on both cases (which both exhibit recordings from -5 to 5 dB but in reverse order), showing its performance is permutation invariant.

of 512. This hidden size is projected back to 1024 dimensions with a final fully-connected layer. All models used the softplus nonlinearity.

3.3. Static Noise Setup

Because RNNs process inputs serially we need to verify that the output we obtain is not mostly dependent on the last few observed channels. More specifically, we want to see that this model can leverage cleaner inputs (and ignore excessively noisy inputs) regardless of whether they are presented first or last. And more generally, we want to ensure that performance is invariant to input presentation order. In order to do that, we simulate a static mixing scenario with a static target and noise, and randomly placed static microphones.

In this setup we examine two cases. In the first case we use k randomly-ordered input channels which exhibit an SNR ranging from -5 to $-5 + k/3$ dB, with $k \in [1, \dots, 30]$. That way for, e.g., six input channels we would observe SNRs ranging from -5 to -3 dB. Conversely, in the second case the input SNRs for k channels are ordered from 5 to $5 - k/3$ dB. In both cases, once $k = 30$ we will see recordings varying from -5 to 5 dB, but for $k < 30$ depending on the case we will either see a set of cleaner recordings, or one of noisier recordings. The goal here is to see how this approach gets swayed by the presentation order of the input channels.

For both increasing and decreasing SNR scenarios, we test a 2D MVN trained on five channels with random order SNRs. The results are shown in Figure 4. For reference we compare the 2D MVN with the baseline RNN.

We observe that MVNs outperform the averaging RNN at leveraging new information and that they are good at ignoring noisy channels and not being swayed by the input channel ordering. This is a very desirable behavior as it means we can safely provide this denoiser with multiple recordings, without having to worry about their ordering and whether the best inputs come first or last.

3.4. Dynamic Noise Setup

The second setup replicates a physically stationary target source, a noise source moving in a circle, and stationary microphones randomly placed within the circle formed by the noisy source path. We set the average SNR across each mixture to 0 while the instantaneous SNR is dependent on the microphone-source geometry. Figure 5 illustrates this setup. Here we reward models capable of leveraging the dynamic nature of the instantaneous SNR for each recording. We call this the dynamic setup.

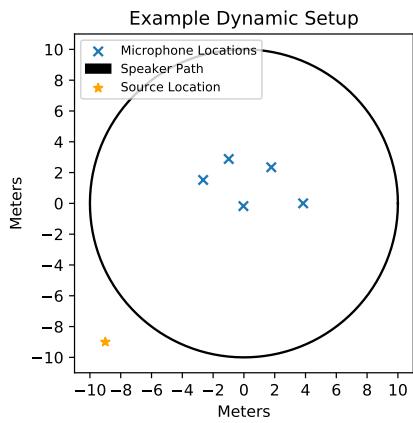


Fig. 5: Example dynamic noise setup with five channels. The noise source moves along a circle, changing which microphone gets the best input each time.

In training we generated samples with five channels. Every training sample simulates a new setup where mic placement and ordering has been randomized. This means our model can't memorize any particular mic setup. However, at test time we maintain the same setup for each different number of microphones to more accurately show how providing additional noisy channels to our model changes performance.

Using this setup we generate a single result plot. The x-axis denotes the number of noisy channels provided to the model. The y-axis shows the average SDR in the validation set for that number of channels. The horizontal line

across the graph denotes the performance of the averaging RNN model, which mostly stays constant regardless of the input channels. We observe that the 2D MVN outperforms the

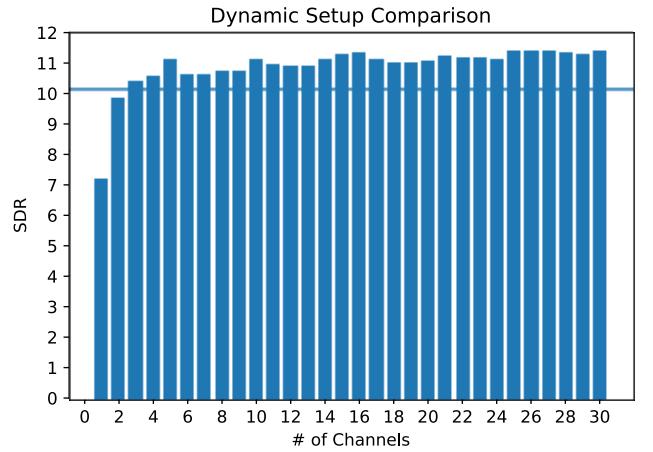


Fig. 6: 2D MVN performance vs number of recordings. Even though the network was trained on 5 channels (note the peak), the quality of outputs keeps improving past 5 channels. In contrast, the regular averaging RNN cannot take advantage of additional channel information. This plot corresponds to a bidirectional GRU across channels.

averaging RNN and is able to leverage numbers of channels far beyond the amount it was trained on. It does not however do as well when only observing one or two channels. We hypothesize that this is because it doesn't fully utilize the recurrent connection in these cases. Furthermore, the MVN can do so even when the "best" recording is different at every time step. This ability is exciting since new information with each channel can be always leveraged, without necessitating changes in the model.

4. CONCLUSION

We have proposed a denoising RNN capable of operating on an arbitrary number of input recordings and leveraging new channel information. We show how the order of the channels does not influence the quality of the results, and that its denoising ability keeps improving as we provide more input channels, even past the amount we trained on. Finally we show how this network outperforms the alternative approach of averaging the input channels. Although not shown here due to space constraints, this model can also operate on an arbitrary number of recordings at every time step, allowing for deployment on settings with a dynamically changing number of sensors.

5. REFERENCES

- [1] Ding Liu, Paris Smaragdis, and Minje Kim, “Experiments on deep learning for speech denoising,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.
- [2] Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis, “Deep learning for monaural speech separation,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1562–1566.
- [3] Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis, “Joint optimization of masks and deep recurrent neural networks for monaural source separation,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 12, pp. 2136–2147, 2015.
- [4] Felix Weninger, John R Hershey, Jonathan Le Roux, and Björn Schuller, “Discriminatively trained recurrent neural networks for single-channel speech separation,” in *Proceedings 2nd IEEE Global Conference on Signal and Information Processing, GlobalSIP, Machine Learning Applications in Speech Processing Symposium, Atlanta, GA, USA*, 2014.
- [5] Jen-Tzung Chien and Kuan-Ting Kuo, “Variational recurrent neural networks for speech separation,” *Proc. Interspeech 2017*, pp. 1193–1197, 2017.
- [6] Prithish Chandna, Marius Miron, Jordi Janer, and Emilia 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.
- [7] Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, Dinei Florêncio, and Mark Hasegawa-Johnson, “Speech enhancement using bayesian wavenet,” *Proc. Interspeech 2017*, pp. 2013–2017, 2017.
- [8] Keiichi Osako, Yuki Mitsufuji, Rita Singh, and Bhiksha Raj, “Supervised monaural source separation based on autoencoders,” in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 11–15.
- [9] Yong Xu, Jun Du, Li-Rong Dai, and Chin-Hui Lee, “A regression approach to speech enhancement based on deep neural networks,” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 1, pp. 7–19, 2015.
- [10] Paweł Swietojanski, Arnab Ghoshal, and Steve Renals, “Hybrid acoustic models for distant and multichannel large vocabulary speech recognition,” in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 285–290.
- [11] Shoko Araki, Tomoki Hayashi, Marc Delcroix, Masakiyo Fujimoto, Kazuya Takeda, and Tomohiro Nakatani, “Exploring multi-channel features for denoising-autoencoder-based speech enhancement,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 116–120.
- [12] Aditya Arie Nugraha, Antoine Liutkus, and Emmanuel Vincent, “Multichannel audio source separation with deep neural networks,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 9, pp. 1652–1664, 2016.
- [13] Bo Li, Tara N Sainath, Ron J Weiss, Kevin W Wilson, and Michiel Bacchiani, “Neural network adaptive beamforming for robust multichannel speech recognition,” in *INTERSPEECH*, 2016, pp. 1976–1980.
- [14] Xiong Xiao, Shinji Watanabe, Hakan Erdogan, Liang Lu, John Hershey, Michael L Seltzer, Guoguo Chen, Yu Zhang, Michael Mandel, and Dong Yu, “Deep beamforming networks for multi-channel speech recognition,” in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5745–5749.
- [15] Zhong-Qiu Wang, Jonathan Le Roux, and John R Hershey, “Multi-channel deep clustering: Discriminative spectral and spatial embeddings for speaker-independent speech separation,” 2018.
- [16] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.
- [17] Shrikant Venkataramani, Jonah Casebeer, and Paris Smaragdis, “Adaptive front-ends for end-to-end source separation,” in *Proc. NIPS*, 2017.
- [18] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett, “Darma timit acoustic-phonetic continuous speech corpus cd-rom. nist speech disc 1-1.1,” *NASA STI/Recon technical report n*, vol. 93, 1993.
- [19] Cédric Févotte, Rémi Gribonval, and Emmanuel Vincent, “Bss_eval toolbox user guide—revision 2.0,” 2005.