Improving single-network single-channel separation of musical audio with convolutional layers

Gerard Roma, Owen Green, and Pierre Alexandre Tremblay

Centre for Research in New Music
University of Huddersfield
Huddersfield, UK
g.roma@hud.ac.uk
o.green@hud.ac.uk
p.a.tremblay@hud.ac.uk

Abstract. Most convolutional neural network architectures explored so far for musical audio separation follow an autoencoder structure, where the mixture is assumed to be a corrupted version of the original source. On the other hand, many approaches based on deep neural networks make use of several networks with different objectives for estimating the sources. In this paper we propose a discriminative approach based on traditional convolutional neural network architectures for image classification and speech recognition. Our results show that this architecture performs similarly to current state of the art approaches for separating singing voice, and that the addition of convolutional layers allows improving separation results with respect to using only fully-connected layers.

Keywords: audio source separation, convolutional neural networks

1 Introduction

The concept of *musical audio* is commonly used to refer to polyphonic mixtures of musical instrument recordings and/or electronic sounds that have been produced and mastered for distribution. Separating such mixtures into source streams has many interesting applications, such as remixing and upmixing [1]. In a production context, being able to adjust an already assembled mix is useful in several situations. A mastering engineer may be able to make adjustments to vocal levels in the order of 1–2 dB without requesting a new mix or vocal stem from their client; or alter the perspectival position of a stream by selectively applying reverb or delay; or use an isolated stream as a key signal for a compressor. In remixing, more radical treatments might use extracted streams as sources for processing, although this is contingent on the degree and type of artefacts introduced by separation. In these contexts, it is important that the raw, extracted streams sum exactly to the original mixture and would pass what sound engineers commonly call a null sum test: inverting the phase of one

mixture and adding this to the other should result in silence. In this way, the producer is always working from an uncompromised, original mix and—accepting that extracted streams will have artefacts—can judge the outcome of adjustments relative to this neutral starting point. As mastering engineer Bob Olhsson notes, "audio processing is the art of balancing subjective enhancement against objective degradation " [2].

The success of deep learning architectures in other domains has sparked interest in applying them to separating musical audio. Here, we are interested in current machine learning systems for musical audio separation to the extent that they can provide *useful approximations* for audio processing applications. Most deep learning approaches rely on having training examples, which requires consistent instrumentation labels. As such, many models are limited to separating just singing voice or other specific streams, which is not very useful in a general production context. Moreover, we should note how much work these labels are made to do in terms of the territory they cover: for instance "vocals" could span Stevie Wonder, Björk, and T-Pain (without even considering more extreme examples). As such, aiming for a 'perfect' decomposition of sources is unrealistic, given the need for labelled data and the complexities of production processes, typically including non-linear effects. A compromise solution is provided by the Demixing Secrets Dataset (DSD100) used in the MUS task of the SiSEC evaluation campaign [3], where each track is consistently seen as a mix of vocals, bass and drums, while other instruments are grouped into a "other" category. On this basis, we take our separation to be yielding *streams* rather than sources that we can think of as being vocal-*like*, bass-*like* and so forth, and take as a priority that the sum of the streams matches exactly the original mixture. For this reason, we adopt the well established framework of time-frequency masking [4].

A good measure of the state of the art is provided by the results of the last SiSEC campaign. The best performance was obtained by systems using Deep Neural Networks (DNN) [5, 6], either feed-forward or recurrent. Most DNN approaches have been based on two-step algorithms. For example, the system in [6] is formulated as a variant of the Expectation Maximization (EM) algorithm where one DNN tries to separate the sources, while a second one tries to enhance the result. A similar approach was presented in [7]. A recent system proposed in [8] also uses two networks (a "Masker" and a "Denoiser"). The rapid adoption of Convolutional Neural Networks (CNN) in domains such as image recognition (including image segmentation) has fostered expectations with respect to audio source separation. Audio is typically analyzed to produce spectrograms, which can be processed as images.

However, applications of CNNs to musical audio separation have only surfaced recently. Most approaches follow an autoencoder structure, where the network tries to produce a de-noised version of the input. In this case, the mixture is seen as a signal where noise has been added to the target source. These networks follow a U-shaped structure where the input is a slice of the spectrogram, and the output is a slice with the same size. This means that after several down-

sampling layers (via convolution or max pooling) there is a series of upsampling (often called "deconvolution" [9]) layers that recover the original dimensions. For example, the system in [10] is composed of an encoding step and a decoding step (using deconvolution layers) connected by a fully-connected layer. The approach in [11, 12] uses a similar system with upsampling layers. These systems have both been evaluated with the DSD100 dataset. Another similar system was proposed in [13] specifically for singing-voice separation. Although it was not evaluated with the same dataset, it seems to provide improvements with the iKala [14] dataset used for singing voice extraction in the MIREX evaluation challenge[1]. However, it relies on a large private training dataset, based on artist distribution of instrumental tracks, so it is not clear whether it would extend to other instruments.

In this paper we investigate a different approach to CNNs for musical audio separation, based on the classic models used in image classification [15]. As opposed to autoencoder-like approaches, our model can be seen as "discriminative", in the sense that the problem is modelled as a classification of time-frequency bins. This implies adding some fully-connected layers after the convolutional layers. With this method, we hope to combine the discriminative power of fully-connected networks with the possibility of learning features from a wider temporal context provided by convolutional layers. As we are interested in the potential for real-time implementation, we limit our temporal scope to texture windows of around 200ms. While this is still a relatively long latency, it improves on the approach proposed by [13] which is based on processing spectrograms of several seconds. Each texture window is used to predict a filter for a given spectral frame.

The rest of the paper is organized as follows. In the next section we describe the proposed approach based on CNNs with two variants. In Section 3, we assess the potential of this model in experiments with the DSD100 dataset. Finally, in Section 4 we reflect on future possibilities for this work.

2 Proposed approach

2.1 Problem formulation

Most recent work on audio source separation is based time-frequency representations typically the Short-Time Fourier Transform, STFT) and relies on the assumption that the transform X of a mixture signal x at time index n and frequency index k results from the sum of i component streams [16]:

$$X_{n,k} = \sum_i S^i_{n,k} \tag{1}$$

As seen in the previous section, in musical audio this may not really need to correspond to the original acoustic sources, but it is assumed that such decomposition would result in useful component signals (also, this assumes that a

[1] `http://www.music-ir.org/mirex/wiki/MIREX_HOME`

constant overlap-add window is used for an STFT). Audio source separation attempts to recover an estimate of each stream S_i, typically with the hope that the original sources do not overlap in X. Hence, the most common way of extracting the stream is by applying a time-frequency mask to X, so that

$$\hat{S}^i_{n,k} = M^i_{n,k} X_{n,k}. \tag{2}$$

The mask M^i can be either binary or soft. Ideal binary masks, specified to be 1 when the target source dominates a given time-frequency bin and 0 otherwise, are routinely used in the STFT domain as an upper bound of automatic music separation, which shows that, even for such broad band signals, the components do not overlap too much in this representation. However, in general for better-sounding estimates, a soft mask where $M^i_{n,k}$ ranges between 0 and 1 is preferred.

2.2 Mask estimation

CNNs have become the standard method for image classification and object detection in images. Conventional CNNs include both convolutional and fully-connected layers. In this setting they are typically used to obtain progressively smaller feature maps, which works for the mentioned tasks, where the output is just a class label, or a label and a set of coordinates. Here, this combination of convolutional and fully-connected layers is useful, given the importance of the temporal context for estimating M_i at a given point. Using DNNs with only fully-connected layers is problematic for this because in order to use multiple frames as input, large numbers of parameters are required. We define the nth input of the network as the sequence of $2c + 1$ magnitude frames:

$$\hat{X_n} = [|x_{n-c}|...|x_n|...|x_{n+c}|], \tag{3}$$

With respect to the objective, one option is to see the separation problem as a classification of time-frequency bins [4]. In this case, the spectrogram can be encoded as:

$$Y_{n,k} = \arg\max_i (S^i_{n,k}). \tag{4}$$

Here, Y is an integer matrix that contains the index of the source with the largest magnitude at each time-frequency bin. We would seek to estimate $\hat{Y}$ so that the mask ${M^i}_{n,k}$ for each time-frequency bin can be obtained as

$${M^i}_{n,k} = \begin{cases} 1 & \text{if } \hat{Y}_{n,k} = i \\ \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

This setting allows us to use the popular softmax function, which we can compute for each time-frequency bin. Our model predicts a single frame, which means that the sequence of outputs produced by the last fully-connected layer of the network , O, is a $N \times K \times I$ tensor which can be seen as the non-normalized

probability of source i at frequency k for frame n. The softmax function then produces normalized probabilities:

$$P^i{}_{n,k} = \frac{e^{O_{n,k,i}}}{\sum_j^I e^{O_{n,k,j}}} \tag{6}$$

The goal of the network is then to minimize the negative log likelihood for the correct class, averaged across frequency bins:

$$l_{nll} = \frac{1}{K} \sum_{\substack{k, \\ i=Y_{n,k}}} -log(P^i{}_{n,k}). \tag{7}$$

Such setting can be used to obtain binary masks that are guaranteed to split the spectrogram evenly, so the estimates would pass the null sum test. However, binary masks typically introduce audible artifacts. Alternatively, an ideal soft mask is computed as

$$M^i_{n,k} = \frac{|S^i_{n,k}|}{\sum_i |S^i_{n,k}|}. \tag{8}$$

In this case, the estimate can be obtained by using a sigmoid function at the output of the network, which is also a $N \times K \times I$ tensor:

$$P^i{}_{n,k} = \frac{e^{O_{n,k,i}}}{1 + e^{O_{n,k,i}}}. \tag{9}$$

Then the mean square error loss can be used:

$$l_{mse} = \frac{1}{I} \sum_i (\frac{1}{K} \sum_k (M^i_{n,k} - P^i{}_{n,k})^2). \tag{10}$$

This is equivalent to estimating a soft mask separately for each source, but with all masks being computed simultaneously by the same network. Hence, while the target soft masks $M^i_{n,k}$ are normalized to sum to one, the output of the network is not guaranteed to do so. In order to preserve this quality, the estimate masks need to be normalized again.

2.3 Network architecture

As mentioned in the previous sections, the proposed architecture consists in combining convolutional layers with a fully-connected output, as featured in the original CNNs for image classification [15]. This architecture has been applied to classification and recognition tasks in speech [17] and musical audio [18], but, to the best of our knowledge, not to musical audio separation (note that [10] included fully-connected layers, but not at the ouptput, which has a different interpretation). These architectures can be applied to the task of musical audio separation based on both optimization targets described on Section 2.2.

Figure 1 describes the basic architecture used in our experiments. Dimensionality reduction is achieved via max-pooling layers, as convolutional layers are padded to result on outputs of the same size. Convolutional layers are connected via ReLU functions, and fully-connected layers are connected with sigmoid functions. After the last fully-connected layer, either a softmax function or a sigmoid activation functions can be used.

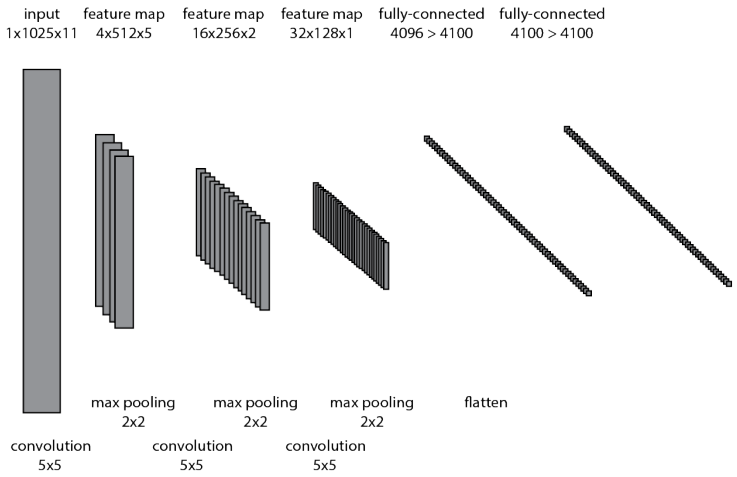

Fig. 1. Base CNN architecture

3 Evaluation

In order to assess the proposed model, we compared three different models on the task of separating musical audio using the DSD100 dataset, roughly following the experimental setting of the SiSEC campaign, which allows us to compare the results to state of the art approaches. Our goal was specifically to assess the addition of convolutional layers to a DNN network. The baseline DNN model (*dnn*) was devised by removing the convolutional layers and extending the input layer to accommodate 5 input frames. The second model (*cnn1*) included the convolutional layers. Both were trained to optimize l_{mse} with a sigmoid function and soft masks. The third model (*cnn2*) was trained to optimize l_{nll} with a softmax output function. For each model, we extracted estimates for vocals, bass, drums and other, the instrument categories of the dataset.

3.1 Experiment setup

The development set, composed of 50 songs, was used for training, while the test set (also 50 songs) was used for testing. The dataset was managed using the

dsdtools package[2]. All songs were mixed to mono by averaging both channels, and downsampled to 22050Hz for processing. The estimates were upsampled again for evaluation, while the reference tracks were also downmixed. Each track was analyzed using a STFT with a window of 2048 samples (~100ms) and hops of 256 samples (~10ms). For each frame, we grouped a sequence of 11 context frames (~200ms) and obtained the magnitude spectrogram of the mixture, and both the classification target and soft masks described in Section 2.2. The networks were trained using the Adaptive Moment Estimation (ADAM) variant of Stochastic Gradient Descent [19].

After shuffling the training set, a validation set of 20% of the data was used to determine the number training epochs. A threshold of 5 epochs was used to stop the training process if the loss had not decreased for the validation set during that time. We used batch normalization [20] for each convolutional layer. In our experience, using large enough batches this made normalizing the data unnecessary. For the *dnn* model, training data was normalized to zero mean and unit variance. Implementation was based on the Pytorch[3] python library. We extracted the SDR, SIR and SAR measures typically used for source separation [21]. Results for each measure and target stream were compared using a Wilcoxon signed-rank test with Bonferroni correction.

3.2 Results and discussion

The results of the experiment are shown in Figure 2. All pair-wise comparisons were found to be significant ($p < 0.01$), with a few exceptions.

The system worked particularly well for separating vocals, with a median SDR just above 4dB. This is similar to state-of-the art results employing multiple network setups, such as [6, 8], but using a single network. Also, the result for vocals is higher than previously published methods based on CNNs [10, 12]. In addition, our system extracts estimates for multiple sources in one pass. Results for other instruments are not as good. This may be due to the fact that early versions of the system were evaluated for extracting of vocals. In early experiments, separating the four streams improved the result for vocals, as opposed to separation of vocals vs accompaniment. The difference may also be due to the breadth of material that non-vocal categories encompass. Balancing the performance between the different instruments would probably require a compromise in terms of window size and overlap factor.

With respect to the different models, we were mainly interested in comparing *cnn1* with the other two, since *dnn* and *cnn2* have a different architecture as well as a different loss function. It should be noted that *dnn* did not have access to the same temporal context, but since this was extended to 5 frames, the number of parameters was higher than for the CNN models (38M vs 34M parameters). Hence, the results show that for a very similar architecture, convolutional layers

[2] https://github.com/faroit/dsdtools
[3] `http://pytorch.org/`

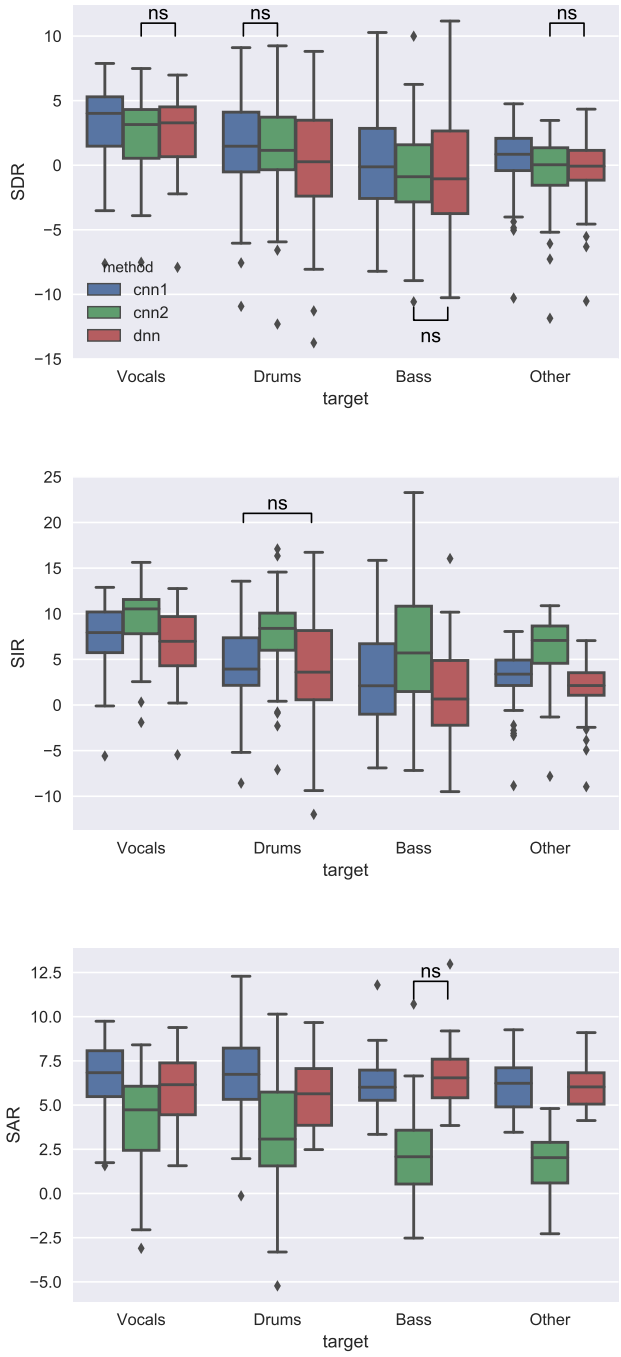

Fig. 2. Results of separation task with the Test set of the DSD100 dataset. All pairwise differences within each measure and target are statistically significant ($p < 0.01$) except where noted ("ns").

allow increasing the temporal context seen by the network, resulting in better performance with a small addition of trainable parameters.

Finally, it could be expected that, since it produces a binary mask, *cnn2* would result in better SIR and lower SAR. This model still gives a similar overall result (SDR) and can be possibly adapted to work in remixing applications where artifacts would be diminished by the presence of all sources.

4 Conclusions

In this paper we have studied the application of Convolutional Neural Networks, in the "traditional" sense used in image classification, to separation of musical audio. Since the use of DNNs is already well established for this task, this work can be seen as incremental, showing that the addition of convolutional layers can improve the results of DNN architectures by allowing access to a longer temporal context. Another advantage of these layers is that it is easy to add additional features. We hope to study this further to keep advancing this model. Our results show that this architecture allows achieving state-of-the-art separation for vocals using a single-network algorithm. We plan to investigate how to improve the results for other instruments. Some examples of the output of our system can be found in the companion page `http://www.flucoma.org/LVA-ICA-2018/`.

5 Acknowledgement

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement n 725899).

References

1. Roma, G., Grais, E.M., Simpson, A.J., Plumbley, M.D.: Music remixing and upmixing using source separation. In: Proceedings of the 2nd AES Workshop on Intelligent Music Production. (2016)
2. Katz, B.: Mastering Audio: The Art and the Science. 3 edn. Focal Press
3. Liutkus, A., Stöter, F.R., Rafii, Z., Kitamura, D., Rivet, B., Ito, N., Ono, N., Fontecave, J.: The 2016 signal separation evaluation campaign. In: International Conference on Latent Variable Analysis and Signal Separation, Springer (2017) 323–332
4. Wang, Y., Narayanan, A., Wang, D.: On training targets for supervised speech separation. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) **22**(12) (2014) 1849–1858
5. Uhlich, S., Giron, F., Mitsufuji, Y.: Deep neural network based instrument extraction from music. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). (April 2015) 2135–2139
6. Nugraha, A., Liutkus, A., Vincent, E.: Multichannel music separation with deep neural networks. (August 2016) 1748–1752

7. Grais, E.M., Roma, G., Simpson, A.J., Plumbley, M.D.: Discriminative enhancement for single channel audio source separation using deep neural networks. In: International Conference on Latent Variable Analysis and Signal Separation, Springer (2017) 236–246
8. Mimilakis, S.I., Drossos, K., Santos, J.F., Schuller, G., Virtanen, T., Bengio, Y.: Monaural Singing Voice Separation with Skip-Filtering Connections and Recurrent Inference of Time-Frequency Mask. arXiv:1711.01437 [cs, eess] (November 2017) arXiv: 1711.01437.
9. Zeiler, M.D., Krishnan, D., Taylor, G.W., Fergus, R.: Deconvolutional networks. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE (2010) 2528–2535
10. Chandna, P., Miron, M., Janer, J., Gómez, E.: Monoaural audio source separation using deep convolutional neural networks. In: International Conference on Latent Variable Analysis and Signal Separation, Springer (2017) 258–266
11. Grais, E.M., Plumbley, M.D.: Single Channel Audio Source Separation using Convolutional Denoising Autoencoders. arXiv:1703.08019 [cs] (March 2017) arXiv: 1703.08019.
12. Grais, E.M., Wierstorf, H., Ward, D., Plumbley, M.D.: Multi-Resolution Fully Convolutional Neural Networks for Monaural Audio Source Separation. arXiv:1710.11473 [cs, eess] (October 2017) arXiv: 1710.11473.
13. Jansson, A., Humphrey, E.J., Montecchio, N., Bittner, R., Kumar, A., Weyde, T.: Singing voice separation with deep U-Net convolutional networks. In: Proceedings of the International Society for Music Information Retrieval Conference (ISMIR). (2017) 323–332
14. Chan, T.S., Yeh, T.C., Fan, Z.C., Chen, H.W., Su, L., Yang, Y.H., Jang, R.: Vocal activity informed singing voice separation with the ikala dataset. In: Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on, IEEE (2015) 718–722
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
16. Vincent, E., Bertin, N., Gribonval, R., Bimbot, F.: From blind to guided audio source separation: How models and side information can improve the separation of sound. IEEE Signal Processing Magazine **31**(3) (2014) 107–115
17. Abdel-Hamid, O., Mohamed, A.r., Jiang, H., Deng, L., Penn, G., Yu, D.: Convolutional neural networks for speech recognition. IEEE/ACM Transactions on audio, speech, and language processing **22**(10) (2014) 1533–1545
18. Schlüter, J., Grill, T.: Exploring data augmentation for improved singing voice detection with neural networks. In: ISMIR. (2015) 121–126
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
20. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
21. Vincent, E., Gribonval, R., Févotte, C.: Performance measurement in blind audio source separation. IEEE transactions on audio, speech, and language processing **14**(4) (2006) 1462–1469