MONAURAL SCORE-INFORMED SOURCE SEPARATION FOR CLASSICAL MUSIC USING CONVOLUTIONAL NEURAL NETWORKS

Marius Miron, Jordi Janer, Emilia Gómez
Music Technology Group, Universitat Pompeu Fabra, Barcelona
firstname.lastname@upf.edu

ABSTRACT

Score information has been shown to improve music source separation when included into non-negative matrix factorization (NMF) frameworks. Recently, deep learning approaches have outperformed NMF methods in terms of separation quality and processing time, and there is scope to extend them with score information. In this paper, we propose a score-informed separation system for classical music that is based on deep learning. We propose a method to derive training features from audio files and the corresponding coarsely aligned scores for a set of classical music pieces. Additionally, we introduce a convolutional neural network architecture (CNN) with the goal of estimating time-frequency masks for source separation. Our system is trained with synthetic renditions derived from the original scores and can be used to separate real-life performances based on the same scores, provided a coarse audio-to-score alignment. The proposed system achieves better performance (SDR and SIR) and is less computationally intensive than a score-informed NMF system on a dataset comprising Bach chorales.

1. INTRODUCTION

As a special case of audio source separation, music source separation has gained significant attention during the past years. Recovering the sources corresponding to the instruments from an audio mixture allows for interesting applications such as music upmixing [9] or virtual-reality concerts [16], and it is useful in music information retrieval tasks [11, 30].

In contrast to speech separation, music source separation poses different challenges due to the variety of sources which are correlated in time and frequency [7]. Because of the multitude of harmonic instruments, often related timbres, variations in dynamics, Western classical music is a challenging case [21]. On the other hand, results can improve if prior knowledge about the nature of sources [5, 29] and their timbre [2] informs the separation framework. Considerable improvements are obtained in the case of parametric models, such as NMF, which are restricted using coarsely aligned scores [4, 7, 10, 14].

Recently, neural network approaches have outperformed NMF in audio source separation challenges [18]. Deep learning systems estimate soft masks for specific instrument classes [3, 13, 15] or computing the instrument spectra directly [27]. In contrast to NMF methods, a deep learning framework is less computationally expensive [3] at the separation stage, as estimating the sources involves a single feed forward pass through the network rather than an iterative procedure. Thus, it can be used in a low latency scenario. Furthermore, recurrent [15] and convolutional [3, 13] networks have the advantage of modeling a larger time context.

Novel deep learning source separation systems propose specialized models which propose building an NMF logic into an autoenconder [26] or cluster components over large time spans [19]. Including score information into the deep learning separation frameworks can yield further improvements [8].

In this paper we introduce a monaural score-informed source separation framework for Western classical music using convolutional neural networks (CNN). We assume that for a given classical music piece the instruments are known and the score is available. Thus, for a set of given scores we generate renditions which are used to train a CNN. The trained model is used to separate real-life performances based on these scores [22].

A global alignment of the score with the audio of a performance can be obtained by a score following system [4]. Then, the resulting coarsely aligned score, with errors up to 0.2 seconds, is used to derive score-based soft masks for each of the sources. From these masks we generate score-filtered spectrograms as input features for the CNN.

Training neural networks for source separation requires isolated audio tracks which are difficult to obtain. Therefore, we use the data generation method in [22]. Accordingly, we synthesize renditions of original scores with variations in timbre, dynamics and local timing deviations.

The remainder of the paper is structured as follows. In Section 2 we state our contributions in relation with the previous work. In Section 3 we introduce the proposed method including the feature computation, the architecture of the network and the training procedure. In Section 4 we discuss the evaluation of the proposed method. We present our conclusions in Section 5.

2. RELATION TO PREVIOUS WORK

Score-informed constraints [7,10] are imposed to the NMF framework through restricting the activations of the note templates. In a similar manner, we use the score to generate sparse training features which are used as input to the CNN. Furthermore, since score following errors influence the quality of separation [4, 20], we compensate for local misalignments in a similar manner to [7, 10], by allowing a tolerance window around note onsets and offsets while computing our training features. For our experiments and the dataset we used the size of this window is 0.2 seconds.

Deep learning systems can become more robust to real-life cases by increasing the size and variability of the training dataset through data generation [22] or augmentation [24, 25]. In this sense, the difference in performance between two similar deep learning methods can be largely explained by the difference between training datasets rather than new features or methods [1]. We are motivated by recent advance in deep learning which go beyond the black-box model and try to integrate musically meaningful features [19,26]. Thus, we aim at improving source separation for classical music with a context-driven method which includes score information.

The CNN architecture in this paper is adapted from the convolutional autoencoder proposed in [3, 22]. In comparison to [3] our CNN architecture has different filter and layer sizes. Moreover, the original scores from which training data is generated are further used to derive score-informed features which are given as input to the CNN in a representation analogous to multi-channel images. To that extent, our approach contrasts with [8] which uses score restrictions inside the deep learning framework. Furthermore, to our best knowledge, deep learning audio processing methods do not use a multi-channel input as in image processing applications. Thus, we analyze whether the convolutional autoencoder introduced in [3,22] learns a better representation from a multi-channel input than from a single channel input, given that the feature maps are shared between all channels. In addition, we use bootstrapping with replacement to train such an architecture when working with big datasets.

3. METHOD

The diagram of the separation framework with the two stages, training and separation, can be seen in Figure 1. For the training stage, we start from the original scores from which we derive synthetic audio renditions with the method in [22]. The same scores are used to derive features for training the CNN in form of score-based soft masks, explained in Section 3.1.1, and score-filtered spectrograms, explained in Section 3.1.2. For the separation stage, our framework takes as input an audio mixture and the corresponding coarsely aligned score. Similar to the training stage, we compute the score-based soft masks and the score-filtered spectrograms which are feed-forwarded through the CNN model to obtain the magnitude spectrograms of the separated sources.

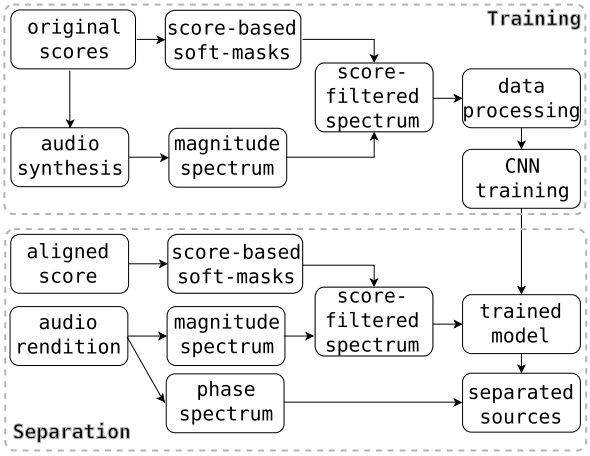

Figure 1. The overview of the separation system comprising the two stages: training and separation

3.1 Feature computation

The goal of computing score-based soft masks is to derive additional sparse score-filtered spectrograms which are used as an input to the CNN.

3.1.1 Score-based soft masks

A score gives the note onsets and offsets time and the MIDI note numbers. Assuming that the source is harmonic and we know the tuning frequency, f_q, the MIDI note associated with A4, m_{A4}, we can compute the fundamental frequency $f_0 = f_q \cdot 2^{\frac{1}{12}\cdot(m-m_{A4})}$, where m is the MIDI note number.

Score information yields the time-frequency zones where the notes are played. Correspondingly, for a given note n that plays between the time frames t_b and t_e we can define the time range as:

$$\boldsymbol{U}_n(t) = u(t - t_b - t_w) - u(t - t_e - t_w) \quad (1)$$

where u is the unit step function, and t_w is tolerance window set around onset t_b and offset t_e which compensates for local misalignments in score-following, similarly to [4,7,10,14]. The tolerance window is applied at training and separation.

Furthermore, if we consider the fundamental frequency f_0 of the note n we define the frequency range as:

$$\boldsymbol{V}_n(f) = \sum_{h=1}^{H} u(f - hf_0/f_i) - u(f - hf_0 f_i) \quad (2)$$

where $h = 1 : H$ are the harmonic partials, and $f_i = 2^{f_c/1200}$ is the allowed frequency interval below and above each harmonic partials, with f_c being the allowed interval in cents, and 1200 is the number of cents per octave.

For each source $j = 1 : J$ and all its notes $n = 1 : N_j$ we can compute score-based binary matrices $\boldsymbol{K_j}(t, f)$ as a sum of outer products:

$$\boldsymbol{K_j}(t, f) = \sum_{n=1}^{N_j} \boldsymbol{U}_n(t) \otimes \boldsymbol{V}_n(f) \quad (3)$$

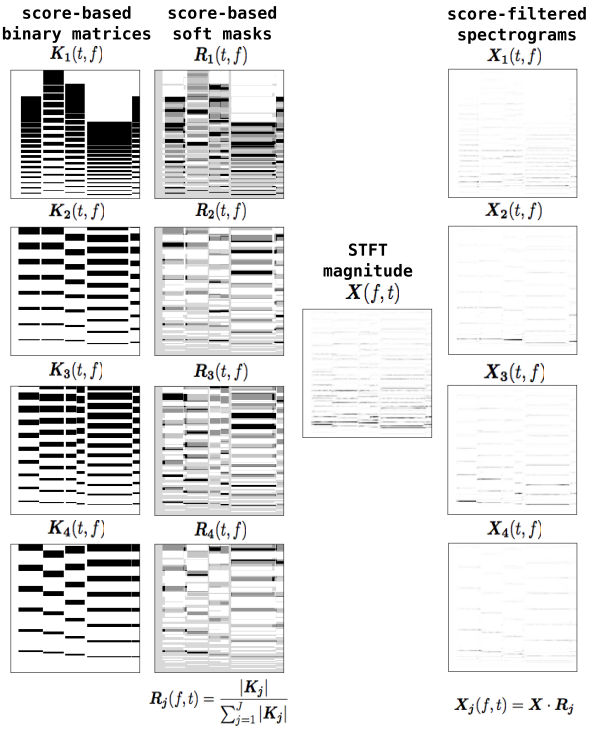

Figure 2. Feature computation for the first 4 seconds and frequencies between 0-6500Hz, for the piece *Ach Gottund Herr* of Bach10 dataset [4] comprising four instruments.

An example of $\boldsymbol{K_j}$ for a classical music piece comprising four harmonic sources is shown in the first column of Figure 2.

The score-based soft masks for each source, $j = 1 : J$, are given by the equation:

$$\boldsymbol{R_j}(f,t) = \frac{|\boldsymbol{K_j}|}{\sum_{j=1}^{J} |\boldsymbol{K_j}| + \epsilon} \tag{4}$$

where $\epsilon = 1^{-10}$ is a constant to handle division by zero. We illustrate a set of $\boldsymbol{R_j}$ matrices in the second column of Figure 2.

In this paper we consider solely combinations between harmonic sources, which are reflected in the initialization of $\boldsymbol{V}_n$ using a series of harmonic partials, as seen in Equation 2. However, the proposed solution can be easily extended to model non-harmonic sources by initializing the vector $\boldsymbol{V}_n(f) = 1$ along all the frequency range, resulting in a less sparse score-filtered spectrogram which is solely informed by onsets and offsets times through $\boldsymbol{U}_n(t)$.

3.1.2 Score-filtered spectrograms

We calculate the STFT magnitude spectrogram of the audio mixture as $\boldsymbol{X}(f,t)$. Then, we derive score-filtered spectrograms for each of the sources $j = 1 : J$, by computing the element-wise product between the spectrogram of the mixture, $\boldsymbol{X}$, and the score-based soft masks, $\boldsymbol{R_j}$:

$$\boldsymbol{X_j}(f,t) = \boldsymbol{X} \cdot \boldsymbol{R_j} \tag{5}$$

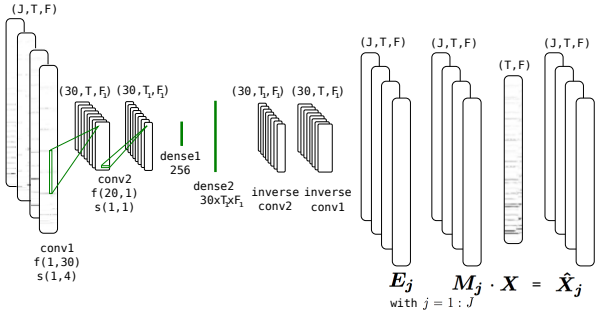

Figure 3. The CNN architecture used in the separation framework for $J = 4$ sources

3.2 Convolutional Neural Network architecture

The convolutional autoencoder architecture can be seen in Figure 3. It comprises a convolution stage with two convolution layers, two dense layers, and a deconvolution stage. The sources are reconstructed using the filters learned at the convolution stage. In addition, we have two deterministic layers to compute the spectrograms of the sources.

In contrast to the CNN architectures in [3, 13, 22], our CNN takes as input J score-filtered spectrograms for a time context T and a number of frequency bins F, rather than a single spectrogram of the mixture. The J score-filtered spectrograms share the same feature maps, in a similar way to image processing deep learning methods that use color RGB channels [1]. Our assumption is that this additional information can better guide the separation between the sources. Furthermore, as shown in Figure 2, the score-filtered spectrograms are sparser versions of the original spectrogram, offering a better representation for source separation [23].

The first layer *conv1* is a convolution layer with filter shape $(1, 30)$, hence the convolution only happens in frequency. For this layer we have a stride [1] of $(1, 4)$, which reduces dimensionality by keeping into account the sparsity of the input.

This layer outputs feature maps of size $(30, T, F_1)$, where $F_1 = (F - 30)/4 + 1$, where 30 is the length of the filter and 4 is the stride. The second layer *conv2* is a convolution layer with filter shape $(20, 1)$, which learns temporal patterns. The output of this layer has the size $(30, T_1, F_1)$, where $T_1 = (T - 20) + 1$ with 20 being the length of the filter. This layer has a stride of $(1, 1)$, since we are interested in maintaining a good temporal resolution at the reconstruction. Note that the convolution layers have a linear activation function.

We use a dense bottleneck layer as in [3] with 256 units and a rectifier linear unit activation function [1], denoted as *dense1*. The limited number of units and the activation function have been proven to better guide the parameter learning and prevent overfitting in the case of timbre-informed source separation [3].

To match dimensions necessary for the deconvolution $(30, T_1, F_1)$ for each of the J sources, we introduce a layer

[1] The stride controls how much a filter is shifted on the input.

dense2 comprising J dense layers of shape $30 \cdot T_1 \cdot F_1$. For each of these J layers we perform the inverse operations of *conv2* and *conv1* and we obtain a set of estimations $\boldsymbol{E_j}$ for each of the separated sources $j = 1 : J$.

Following [3, 15], we integrate the computation of the soft masks into the architecture of the network as an additional deterministic layer. Thus, the soft masks $\boldsymbol{M_j}$, for each source $j = 1 : J$, are computed from the output of the previous layer, $\boldsymbol{E_j}$, as:

$$\boldsymbol{M_j} = \frac{|\boldsymbol{E_j}|}{\sum_{j=1}^{J} |\boldsymbol{E_j}| + \epsilon} \tag{6}$$

where $\epsilon = 1^{-10}$ is a constant to handle division by zero. The magnitude spectrogram corresponding to the sources, $\hat{\boldsymbol{X}}_{\boldsymbol{j}}$, are given by the element-wise multiplication between input spectrogram and the soft-masks $\hat{\boldsymbol{X}}_{\boldsymbol{j}} = \boldsymbol{M_j} \cdot \boldsymbol{X}$. The soft masks $\boldsymbol{M_j}$ are not to be confounded with the score-based soft-masks $\boldsymbol{R_j}$ introduced in Section 3.1.1 and used to derive input features for the CNN.

3.3 Training procedure

The network is trained according to the mean-squared error between the magnitude spectrograms of the target sources, $\hat{\boldsymbol{X}}_{\boldsymbol{j}}$, and the magnitude spectrograms of the sources yielded by the network, $\boldsymbol{X_j}$, as: $Loss = \sum_{j=1}^{J} \|\hat{\boldsymbol{X}}_{\boldsymbol{j}} - \boldsymbol{X_j}\|^2$.

The parameters of the CNN are updated using mini-batch Stochastic Gradient Descent with the *AdaDelta* algorithm [31].

With the method in [22] we can generate a high number of renditions, covering a high number of possibilities, which makes the framework more robust to real-life data. However, training on big datasets is an expensive procedure and we experimented with a faster training method summarized in the Algorithm 1. In this case, we sample a limited number data points before each epoch rather than having a fixed dataset at the beginning of training. In statistics, this procedure is known as bootstrapping with replacement [17]. Note that, for this training procedure, the concept of *epoch* (a single pass through the entire training set) does not hold anymore.

Algorithm 1 *Bootstrapping with replacement*

```
repeat
    randomly sample a number of data points from the dataset
    for each training batch do
        compute weights and bias gradients for the current batch
        accumulate the gradients
    end for
    adjust weights and bias using accumulated gradients
until total number of stages is reached
```

3.4 Separated source estimation

We assume that the individual sources $y_j(t), j = 1 : J$, that compose the audio mixture $x(t)$ are linearly mixed, so that $x(t) = \sum_{j=1}^{J} y_j(t)$. Therefore, from the estimated magnitude spectrograms $\boldsymbol{X_j}$ and using the original phase of the audio mixture we can obtain the signals associated to the sources, $y_j(t)$, with an inverse overlap-add STFT [10].

The neural network yields estimations of shape (T, F) for each of the J sources. Considering an audio mixture of variable time shape, the estimation is done for overlapping segments of shape (T, F), with the algorithm described in [22].

4. EVALUATION

4.1 Datasets

For evaluation purposes we use ten Bach chorales from the Bach10 dataset [4], played by bassoon, clarinet, tenor saxophone, and violin. The mean duration of a piece is ≈ 30 seconds. In addition, each piece is accompanied by the score aligned with the audio, the original score, and an automatic alignment obtained with the algorithm in [4]. This dataset has been widely used in tasks as source separation, alignment, and transcription.

4.2 Generating training data

We generate training data with the method in [22] which uses sample-based synthesis with samples from the RWC instrument sound database [12]. The method synthesizes original scores at different tempos, dynamics, considering local timing deviations, and using different timbres to generate a wide variety of renditions of given pieces.

In this case, we have three different timbres and three level of dynamics. In addition, to account for local timing variations, we circular-shift the audio with $s = \{0, 0.1, 0.2\}$ seconds. An analogous transformation needs to be applied to the associated score by adding s seconds to the note onsets and offsets.

Considering the variations of the factors above ($3 \cdot 3 \cdot 3 = 27$) for the four instruments, we can generate a total number of $27^4 = 531441$ renditions for a single piece. Because it is not feasible to generate such a high number of audio files, we randomly choose 400 renditions to build our training dataset. Samples are uniformly distributed across the dataset. Since we are training a CNN model for all the 10 pieces in Bach10 dataset, we have a total number of 4000 renditions.

4.3 Evaluation setup

We used the evaluation framework and the metrics described in [28] and [6] : *Source to Distortion Ratio* (SDR), *Source to Interference Ratio* (SIR), and *Source to Artifacts Ratio* (SAR).

The STFT is computed using a Blackman-Harris window of length 4096 samples, which at a sampling rate of 44.1 KHz corresponds to 93 milliseconds (ms), and a hop size of 512 samples (11ms).

When computing the soft-masks from the score, as described in Section 3.1.1, we consider the tuning frequency,

$f_q = 440$Hz, the MIDI note associated with A4, $m_{A4} = 69$, and we allow $f_c = 40$ cents above and below each harmonic partials to account for vibrato. Additionally, because we want to train our score-informed system to account for errors in score following, we set the tolerance window to be $t_w = 0.2$ seconds around onsets and offsets.

The time context modeled by the CNN is $T = 30$ frames. Furthermore, a more robust system is achieved by taking consecutive T-sized frames with an overlap of 25 frames with the algorithm described in [22].

The number of epochs is variable for each training experiment. The size of a mini-batch is set to 32.

This paper follows the principles of research reproducibility [2] . The code used in this paper is made available online [3] . It is built on top of Lasagne, a framework for neural networks using Theano [4] . We ran the experiments on a Ubuntu 16.04 PC with GeForce GTX TITAN X GPU, Intel Core i7-5820K 3.3GHz 6-Core Processor, X99 gaming 5 x99 ATX DDR44 motherboard.

4.4 Experiments

In a first experiment, we compare the proposed framework with an NMF counterpart on the Bach10 dataset. We train our CNN framework on the synthetic dataset we described in Section 4.2 (10×400 renditions) and the corresponding scores. Because we want the model to learn to deal with errors in alignment we set a tolerance window around notes' onsets and offsets. Then, we test the resulting model on real-life performances in Bach10 dataset and the scores yielded by the score-following system in [4].

Because we want to isolate the influence of the score-following system, we test our system on the score perfectly aligned (PA) with the audio. For this case, denoted as CNN PA, the tolerance window is not needed, neither for training nor testing. Furthermore, to assess the influence of the proposed features, we train the CNN architecture without any score information, having as input the magnitude spectrogram of the mixture, similarly to the system in [22]. We denote this experiment as CNN T.

We compare our score-informed system to a state of the art NMF counterpart [20]. The note templates are trained on the RWC dataset and are kept fixed during the factorization. Score-information is introduced through the activation matrix by setting to zero the activations corresponding to notes which are not played. The activations which are set to zero will remain this way during factorization, allowing the energy to be distributed between the active templates.

For the NMF system we use as input the score aligned with [4] with a tolerance window of 0.2 seconds, and the perfectly aligned score, as two separate cases, denoted as NMF and NMF PA. Furthermore, for the NMF we kept the default parameters presented in the paper [20]: 50 iterations for the factorization, beta-divergence distortion $\beta = 1.3$, STFT window size $93ms$, and hop size $11ms$.

For this first experiment we do not test the bootstrapping with replacement procedure. To that extent, we train the CNN with all the 4000 renditions for a maximum number of 20 epochs and we stop training if the loss between two epochs drops below 0.2.

In a second experiment, we test the effectiveness of the training procedure based on bootstrapping with replacement, described in Algorithm 1 and compare it with the standard training procedure which maintains the same data points during training. Furthermore, since we want to determine the optimal value for the number of renditions used at each epoch or stage, we train the CNN successively with the two procedures using different numbers of renditions. For this experiment we train for a number of 50 epochs or stages.

4.5 Results

The SDR, SIR, and SAR for our system (CNN and CNN PA), the timbre informed version CNN T, and the NMF framework are presented in the Figure 4. Error bars are drawn for a confidence interval of 95%.

We observe that the proposed score-informed framework performs better than NMF when working with coarsely aligned scores: 6dB vs 5dB in SDR. Hence, with our framework we are able to compensate for local misalignment errors around 0.2 seconds. This results in less interference, since the CNN method has $2dB$ more in SIR than the NMF, and can be due to the fact that the CNN models temporal patterns in the *conv2* layer and to the nonlinearities in the bottleneck *dense1* layer.

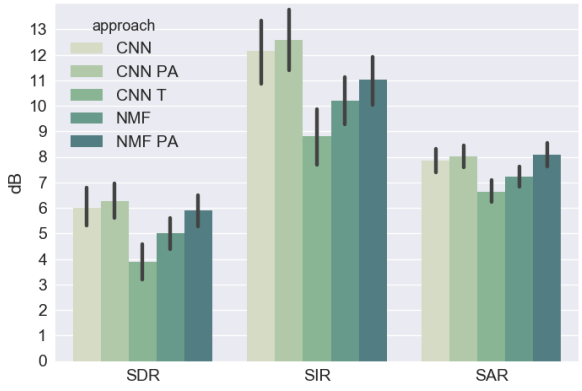

Figure 4. Results in terms of SDR, SIR, SAR for the proposed CNN framework and the NMF framework [20]

Having score-filtered spectrograms as input (CNN) improved $2dB$ in SDR in comparison to giving the magnitude spectrograms as input (CNN T), which proves the effectiveness of the features derived from score.

When the score is perfectly aligned with the audio, there is no significant difference in SDR between the CNN PA and NMF PA. However, the proposed method has $1dB$ higher SIR and similar SAR values to the NMF PA. Note that CNN PA is trained on the original scores and it is not

[2] http://soundsoftware.ac.uk/resources/
[3] https://github.com/MTG/DeepConvSep
[4] http://lasagne.readthedocs.io/en/latest/Lasagne and http://deeplearning.net/software/theano/Theano

targeted for special case. To that extent, as the CNN and CNN PA achieve similar results, we believe that having a perfect alignment does not improve results for this particular type of CNN architecture. This is in line with the results obtained in [22].

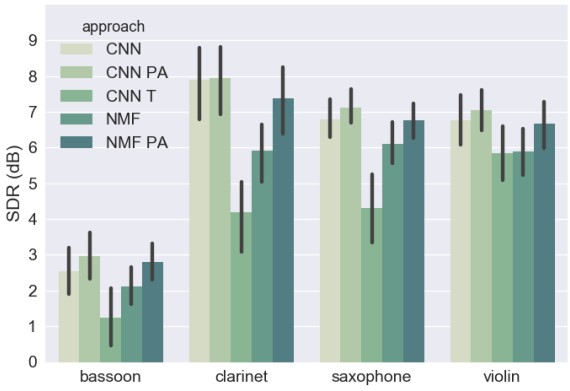

Figure 5. Results for each instrument in terms of SDR for the considered approaches: CNN and NMF [20]

We present the results in terms of SDR for each instrument in Figure 5. The CNN framework performs significantly better than the NMF for all the instruments, with the exception of bassoon. While experimenting with different STFT window sizes, we observed that the quality of the separation for bassoon improved considerably with the increase in the window size, while remaining the same for the other instruments. However, a larger window size means a higher feature dimensionality, hence more weights to be trained and a larger model.

We observe that the proposed framework effectively compensates for errors in alignment across all instruments, especially for clarinet.

The audio examples for the CNN framework and the computed metrics for CNN,CNN PA, CNN T, NMF, and NMF PA as .mat files can be accessed online [5] .

In the second experiment we are interested in testing the bootstrapping with replacement training procedure and the standard procedure. The results for various number of renditions can be seen in Figure 6.

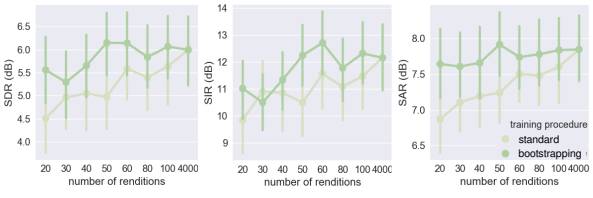

Figure 6. Results in terms of SDR,SIR,SAR when training the proposed CNN with stardard training method vs *bootstrapping with replacement* with various number of training samples

We observe that *bootstrapping with replacement* always improves over the standard training procedure, particularly for a small number of training renditions. However, a lower than 50 number of renditions, decreases the performance for both of the training methods. In some cases (50,60,100), using the proposed training procedure with fewer samples is slightly better than training with the whole dataset, as it prevents overfitting, in a similar way to early stopping [1]. The optimum number of renditions for our experimental scenario is 50 samples.

[5] http://doi.org/10.5281/zenodo.821128

5. CONCLUSION

We proposed a score-informed source separation framework targeted at Western classical music. Our framework is based on the assumption that classical music pieces are accompanied by scores and this information can be leveraged. Thus, we proposed a framework which is trained with generated renditions synthesized from the original scores. Provided an accurate automatic audio-to-score alignment can be obtained by a score-following system, our framework separates with low latency any real-life performances based on those scores, accompanied by a coarse alignment.

We presented a novel method to derive training features in the form of score-filtered spectrograms, which can easily be integrated with CNN architectures. In particular, these sorts of homogeneous features are well suited to learning convolutional filters which are shared between the input channels of the CNN.

The proposed system has better SDR and SIR than a state of the art score-informed NMF framework, particularly when working with coarsely aligned score, as it is the case of the output of score-following systems. Furthermore, we tested a faster training procedure, bootstrapping with replacement, which preserves the performance and in some cases prevents overfitting. As future work, we plan on extending this framework to multi-microphone orchestral music which is a more complex scenario due to increased number of instruments. Moreover, reiterating the method, by inputting the output of the network to another similar network, could improve results [27].

6. ACKNOWLEDGMENTS

The TITANX used for this research was donated by the NVIDIA Corporation. This work is partially supported by the Spanish Ministry of Economy and Competitiveness under CASAS project (TIN2015-70816-R) and by the Spanish Ministry of Economy and Competitiveness under the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502). We thank Matthew E.P. Davies for his feedback.

7. REFERENCES

[1] Y Bengio. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[2] JJ Carabias-Orti, T Virtanen, P Vera-Candeas, N Ruiz-Reyes, and FJ Canadas-Quesada. Musical Instrument Sound Multi-Excitation Model for Non-Negative Spectrogram Factorization. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1144–1158, October 2011.

[3] 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*, 2017.

[4] Z. Duan and B. Pardo. Soundprism: An online system for score-informed source separation of music audio. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1205–1215, 2011.

[5] J.-L. Durrieu, A. Ozerov, C. Févotte, G. Richard, and B. David. Main instrument separation from stereophonic audio signals using a source/filter model. In *Signal Processing Conference, 2009 17th European*, pages 15–19. IEEE, 2009.

[6] V. Emiya, E. Vincent, N. Harlander, and V. Hohmann. Subjective and Objective Quality Assessment of Audio Source Separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2046–2057, September 2011.

[7] S. Ewert and M. Müller. Using score-informed constraints for nmf-based source separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 129–132. IEEE, 2012.

[8] S Ewert and MB Sandler. Structured dropout for weak label and multi-instance learning and its application to score-informed source separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 2277–2281. IEEE, 2017.

[9] D Fitzgerald. Upmixing from mono-a source separation approach. In *Digital Signal Processing (DSP), 2011 17th International Conference on*, pages 1–7. IEEE, 2011.

[10] J. Fritsch and M.D. Plumbley. Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 888–891. IEEE, 2013.

[11] E. Gómez, F. Cañadas, J. Salamon, J. Bonada, P. Vera, and P. Cabañas. Predominant Fundamental Frequency Estimation vs Singing Voice Separation for the Automatic Transcription of Accompanied Flamenco Singing. *13th International Society for Music Information Retrieval Conference*, 2012.

[12] M. Goto. Development of the RWC music database. In *Proceedings of the 18th International Congress on Acoustics (ICA 2004)*, pages 553–556, 2004.

[13] 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*, pages 3734–3738. IEEE, may 2014.

[14] R. Hennequin, B. David, and R. Badeau. Score informed audio source separation using a parametric model of non-negative spectrogram. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, number 1, pages 45–48. IEEE, 2011.

[15] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis. Deep Learning for Monaural Speech Separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 1562–1566, 2014.

[16] J. Janer, E. Gómez, A. Martorell, M. Miron, and B. de Wit. Immersive orchestras: audio processing for orchestral music VR content. In *Games and Virtual Worlds for Serious Applications (VS-Games), 2016 8th International Conference on*, pages 1–2. IEEE, 2016.

[17] R Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Stanford, CA, 1995.

[18] A Liutkus, F-R Stöter, Z Rafii, D Kitamura, B Rivet, N Ito, N Ono, and J Fontecave. The 2016 signal separation evaluation campaign. In *International Conference on Latent Variable Analysis and Signal Separation*, pages 323–332. Springer, 2017.

[19] Y. Luo, Z. Chen, J. R Hershey, J. Le Roux, and N. Mesgarani. Deep clustering and conventional networks for music separation: Stronger together. *arXiv preprint arXiv:1611.06265*, 2016.

[20] M Miron, J.J. Carabias, and J Janer. Improving score-informed source separation for classical music through note refinement. *16th International Society for Music Information Retrieval Conference*, 2015.

[21] M Miron, JJ Carabias-Orti, JJ Bosch, E Gómez, and J Janer. Score-informed source separation for multichannel orchestral recordings. *Journal of Electrical and Computer Engineering*, 2016, 2016.

[22] M. Miron, J. Janer, and E. Gómez. Generating data to train convolutional neural networks for classical music source separation. In *Proceedings of the 14th Sound and Music Computing Conference*, pages 227–233, 2017.

[23] M.D. Plumbley, T. Blumensath, L. Daudet, R. Gribonval, and M.E. Davies. Sparse representations in audio and music: from coding to source separation. *Proceedings of the IEEE*, 98(6):995–1005, 2010.

[24] J. Salamon and J.P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017.

[25] J. Schlüter and T. Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *16th International Society for Music Information Retrieval Conference*, pages 121–126, 2015.

[26] P. Smaragdis and S. Venkataramani. A neural network alternative to non-negative audio models. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 86–90. IEEE, 2017.

[27] S. Uhlich, F. Giron, and Y. Mitsufuji. Deep neural network based instrument extraction from music. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 2135–2139. IEEE, 2015.

[28] E. Vincent, R. Gribonval, and C. Fevotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 14(4):1462–1469, jul 2006.

[29] T. Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3):1066–1074, 2007.

[30] JR Zapata and E Gomez. Using voice suppression algorithms to improve beat tracking in the presence of highly predominant vocals. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 51–55. IEEE, may 2013.

[31] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.