Score-informed Music Source Separation: Improving Synthetic-to-real Generalization in Classical Music

Eetu Tunturi
Audio Research Group
Tampere University
Tampere, Finland
eetu.tunturi@tuni.fi

David Diaz-Guerra
Audio Research Group
Tampere University
Tampere, Finland
david.diaz-guerra@tuni.fi

Archontis Politis
Audio Research Group
Tampere University
Tampere, Finland
archontis.politis@tuni.fi

Tuomas Virtanen
Audio Research Group
Tampere University
Tampere, Finland
tuomas.virtanen@tuni.fi

***Abstract*—Music source separation is the task of separating a mixture of instruments into constituent tracks. Music source separation models are typically trained using only audio data, although additional information can be used to improve the model's separation capability. In this paper, we propose two ways of using musical scores to aid music source separation: a score-informed model where the score is concatenated with the magnitude spectrogram of the audio mixture as the input of the model, and a model where we use only the score to calculate the separation mask. We train our models on synthetic data in the SynthSOD dataset and evaluate our methods on the URMP and Aalto anechoic orchestra datasets, comprised of real recordings. The score-informed model improves separation results compared to a baseline approach, but struggles to generalize from synthetic to real data, whereas the score-only model shows a clear improvement in synthetic-to-real generalization.**

***Index Terms*—Music source separation, deep learning, machine learning, classical music**

I. Introduction

The target of music source separation is to isolate the signals of individual sources from a mixture. In monaural source separation, tracks are separated from a single-channel input mixture. Music source separation has applications in music upmixing, remixing, virtual reality, and music analysis.

Most of the time, music source separation models use only the mixture of all the instruments as their input. The mixture can contain multiple sources that share timbral characteristics, leading to high correlations among the sources. Additional information can be used to aid the separation model, such as visual information [1], instrument activity labels [2], or musical score [3] [4]. This study focuses on using scores as additional information. It is reasonable to assume that one could have the score available in a real application, especially in the case of classical music. When aligned with the audio, scores indicate the onsets and the offsets of every instrument and the pitch of every note, which can help us know in which frequencies we can expect to find said notes. Score information has been found to improve separation results in frameworks like non-negative matrix factorization (NMF) [5] [6] [7] and hidden Markov models (HMMs) [8].

In the past decade, deep neural networks have provided a significant improvement over NMF and HMMs in music source separation [9] [10] [11] [12] [13]. However, there is not much research about integrating score information into deep learning methods. In [3], the scores are used to filter the magnitude spectrograms that are used to train a CNN. A score-filtered spectrogram is created for each instrument, and the CNN takes the spectrograms of all of the instruments as its input. In [4], the scores are used as weak labels to train separation models in an unsupervised way. The scores are used to enforce the information of every note to be in different dimensions of the latent space of an autoencoder, so that this structured latent space can be modified to separate every note from the mixture. The approach is evaluated by separating the right and left-hand notes from piano recordings, but it has not been studied how this approach would work for separating multiple different instruments.

Source separation of classical music is an especially difficult problem, largely due to the limited amount of training data. There are many instruments in classical music, and songs are recorded such that all of the instruments are played at the same time in the same room. Due to the recording setup, ground-truth signals corresponding to isolated instruments cannot be obtained from normal recordings. Some datasets have been specifically recorded with music information retrieval in mind, such as [14] and [15], which contain isolated ground truth signals. However, the number of real recordings remains so low that it is difficult to train deep neural networks on them. There are large synthetic datasets like [16] and [17], but based on their baseline results, models trained on synthetic data do not generalize well to real data.

This paper proposes two different ways of using the score information for music source separation: concatenating it to the audio information and using only the score to calculate the separation masks. In the latter approach, the separation masks

This work was supported by "REPERTORIUM" Project under Grant Agreement 101095065. Horizon Europe. Cluster II. Culture, Creativity and Inclusive Society. Call HORIZON-CL2-2022-HERITAGE-01-02.
The authors wish to acknowledge CSC—IT Center for Science, Finland, for computational resources.

are less dependent on training data, which helps it generalize better to unseen conditions. We evaluate our methods on the SynthSOD [16], Aalto anechoic orchestra [14], and URMP [15] datasets. Results indicate that the use of score information improves separation performance and generalization from synthetic to real data.

II. Methodology

A. Baseline approach

Our methods use the magnitude of the STFT of the input mixture to estimate a spectral mask for each target instrument. We apply the spectral masks to the STFT of the mixture to obtain the separated STFTs, and then apply the inverse STFT to obtain the separated signals; this high-level structure of our methods is illustrated in Figure 1.

We chose X-UMX [18] as our baseline model as it was also used as the baseline for our training dataset [16]. We used the implementation of X-UMX from the open-source Asteroid library [19]. As we trained our models to separate 15 different instruments, the amount of GPU memory required for training became large. Therefore, we split the model into four separate models corresponding to the four instrument families: strings (violin, viola, cello, bass), woodwinds (flute, clarinet, oboe, bassoon), brass (horn, trombone, tuba, trumpet), and percussion (timpani, harp, untuned percussion). The four models are trained completely independently.

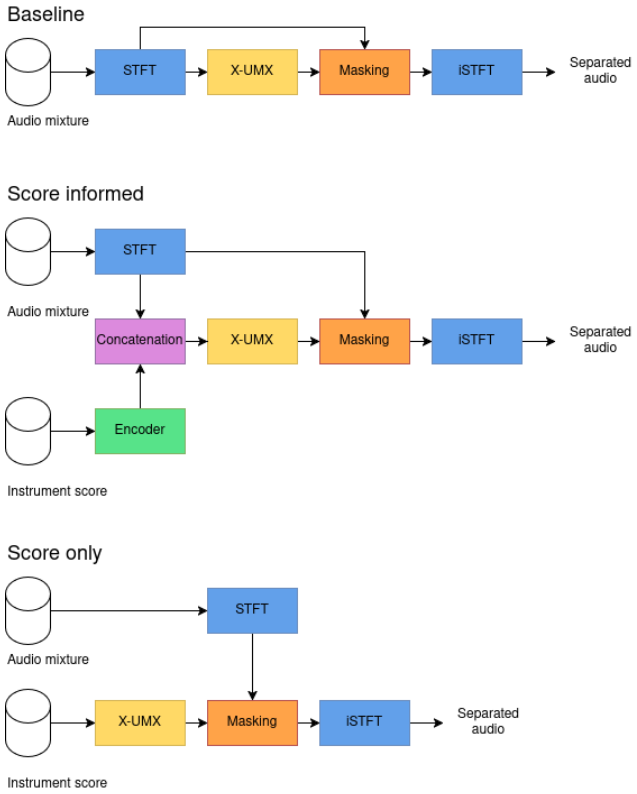

Fig. 1. Block diagrams of the three evaluated methods.

X-UMX consists of an encoder, three bidirectional LSTM layers, a decoder, and two global averaging layers. The encoder is a linear layer followed by batch normalization and hyperbolic tangent activation. The decoder contains two linear layers, each followed by batch normalization, and the first followed by ReLU activation. The encoder, LSTM block, and decoder can have multiple branches which all have their own set of weights. We set the number of branches to the number of instruments for each model and for each of the three parts of the model. The decoder has to have as many branches as instruments to obtain the correct number of separation masks, but the encoder and LSTM block could have a different number of branches. However, we decided to keep them all the same for simplicity. There is an averaging layer between the encoder and the LSTM block, and between the LSTM block and the decoder, which computes the mean across the branches. In the baseline model, the input of each encoder branch is the same: the magnitude spectrogram of the audio mixture.

For our loss function, we adopt both multi-domain loss and combination loss as described in [18]. Multi-domain loss is the sum of the mean squared error between the estimated and ground truth magnitude spectrograms for the frequency domain component, and a weighted signal-to-distortion ratio for the time domain component. In combination loss, instead of only considering the loss of each instrument independently, we also calculate the loss for combinations of instruments by combining their separation masks. The final loss is the mean of all instrument combinations, except for the combination of all instruments together.

B. Proposed approaches

We preprocess the score into a piano roll representation. Using the same temporal resolution as the spectrogram, for every time frame we represent the note activity as a vector

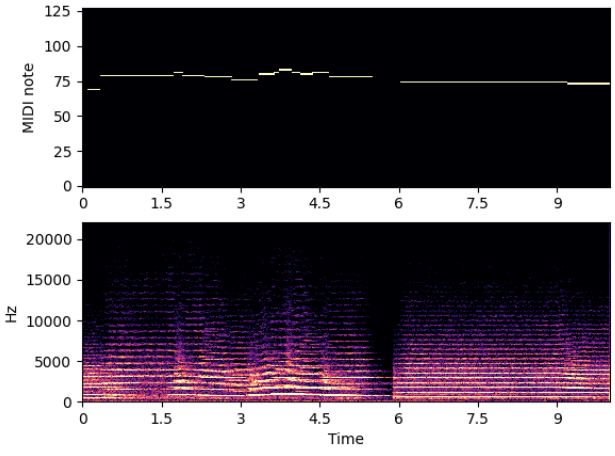

Fig. 2. The piano roll score representation (top) and magnitude spectrogram (bottom) of a 10-second segment of violin from the SynthSOD dataset.

TABLE I

SIGNAL TO DISTORTION RATIOS [DB] FOR MODELS TRAINED WITH SYNTHSOD AND EVALUATED IN THE ENSEMBLES (UP TO 5 INSTRUMENTS) OF THE TEST PARTITION OF SYNTHSOD, AND URMP. THE FIRST COLUMN OF EVERY EVALUATION DATASET INDICATES THE SDR OF THE ORIGINAL MIXTURES.

Evaluation on:	Ensembles in SynthSOD				URMP			
Instrument	Original	Baseline	Score informed	Score only	Original	Baseline	Score informed	Score only
Violin	-3.73	8.15	**8.61**	5.26	-2.32	0.70	0.72	**5.49**
Viola	-5.87	6.06	**6.58**	5.08	-5.59	0.21	0.60	**6.16**
Cello	1.25	**10.28**	10.06	8.63	-5.06	3.39	4.97	**5.83**
Bass	-5.17	6.85	**7.37**	6.27	-6.56	3.43	3.36	**6.20**
Flute	-11.79	2.14	**4.13**	1.27	-2.67	1.19	0.95	**3.35**
Clarinet	-7.41	2.36	**4.98**	4.32	-4.44	0.20	0.56	**3.78**
Oboe	-5.19	9.81	**10.14**	3.77	-6.54	0.17	0.31	**1.87**
Bassoon	-3.79	5.77	**7.16**	2.07	-3.39	0.65	0.49	**3.04**
Horn	-4.38	1.51	**4.64**	2.10	-6.38	1.56	1.51	**2.84**
Trumpet	-0.02	8.27	**9.01**	6.25	-2.38	1.84	3.07	**5.73**
Trombone	1.02	7.36	**9.05**	6.06	-3.84	0.69	1.83	**3.79**
Tuba	5.08	4.00	**7.85**	6.78	-6.43	0.03	1.09	**5.78**
Harp	-11.56	2.72	**3.40**	2.53				
Timpani	-13.80	**3.64**	0.82	0.42				
Unt. perc.								
MEAN	-4.67	5.64	**6.70**	4.34	-4.63	1.17	1.62	**4.49**

with length 128 (i.e., the number of MIDI notes), where the note activity is denoted by 1 (on) or 0 (off). Figure 2 shows the piano roll of one of the instruments of a track from SynthSOD aligned with the corresponding magnitude spectrogram. Before feeding the score to the model, it is aligned with the audio during preprocessing, as explained in Section II-C.

In our score-informed model, we first encode the score information with a small encoder consisting of a 3-layer bidirectional LSTM followed by batch normalization and ReLU activation. The score encoder has its own set of trainable weights for each instrument. The encoded score information is concatenated framewise with the magnitude spectrogram of the mixture and then given to an X-UMX model with the same hyperparameters as the baseline, with the exception of the input size. Each branch of the encoder of the X-UMX gets the score of one instrument. We also tried different approaches to integrate the score information with the audio, but we obtained very similar results with all of these approaches. The different approaches we experimented with included combining the score and audio at different intermediate layers in the X-UMX architecture, and multiplying the score and audio instead of concatenating them.

We also propose a score-only model. The score-only model uses only the score to create the spectral masks, as seen in Figure 1. In this case, we do not use a separate encoder for the scores, instead, X-UMX directly takes the piano rolls of every instrument as inputs, each in its own encoder branch. The score representation used is the same as for the score-informed model. For the score-only model, we removed the input and output normalizations of X-UMX since we found that it worked better without them in this case.

C. Score alignment

It is important for separation performance that the score and audio are aligned. Of the three datasets used in our experiments, URMP [15] and the Aalto anechoic orchestra dataset [14] provide aligned scores. However, our training dataset, SynthSOD [16], does not provide aligned score information. SynthSOD is a large-scale dataset of synthetic classical music containing both ensembles and orchestral music. We obtained our score information from the MIDI files that were used to synthesize the dataset. However, since random tempo segments were used to increase the diversity of the dataset, there are significant misalignments between the original scores and the audio. Misalignments could cause problems in training the model, so we chose to align the MIDI files with the audio as a preprocessing step.

To align the scores to the audio, we used the system described in [20]. This system splits the score into different segments according to the concurrent notes and their transitions, which are synthesized and modeled using non-negative matrix factorization, and aligns them to the audio using a dynamic time warping algorithm.

From the alignment, we obtain a CSV file for each song, which contains, for every note in the song, the corresponding onset, offset, pitch, and instrument. The synthesizer used for the alignment cannot parse some of the MIDI files perfectly, so we chose to exclude those songs from all of our experiments. We release the aligned scores and updated metadata files[1]. In this paper we focus on the effectiveness of aligned scores. Temporal misalignments will affect results, but we leave the study of those effects for future work.

III. EXPERIMENTS

A. Training setup

We train all of our models on the train partition of SynthSOD, excluding the songs that we were unable to align from the training set, as mentioned in Section II-C. Of the 408 original songs, we were able to successfully align 351. SynthSOD contains two different violin tracks for many songs,

[1][Online]. Available: https://doi.org/10.5281/zenodo.15575778

TABLE II
SIGNAL TO DISTORTION RATIOS [DB] FOR MODELS TRAINED WITH SYNTHSOD AND EVALUATED IN THE ORCHESTRA PIECES (MORE THAN 5 INSTRUMENTS) OF THE TEST PARTITION OF SYNTHSOD, AND AALTO. THE FIRST COLUMN OF EVERY EVALUATION DATASET INDICATES THE SDR OF THE ORIGINAL MIXTURES.

Evaluation on:	Orchestras in SynthSOD				Aalto			
Instrument	Original	Baseline	Score informed	Score only	Original	Baseline	Score informed	Score only
Violin	-9.05	**4.07**	3.91	2.12	-7.27	2.60	**2.72**	2.09
Viola	-10.61	1.87	**2.60**	2.10	-13.36	0.01	**0.25**	-0.07
Cello	-5.60	3.86	**4.24**	3.53	-15.07	-2.68	**-0.42**	-1.33
Bass	-5.65	7.58	**8.04**	7.44	-15.63	4.74	5.04	**6.08**
Flute	-12.60	1.30	**3.50**	2.06	-15.32	0.06	**0.45**	-1.33
Clarinet	-8.95	0.45	**2.64**	1.69	-8.31	0.12	0.54	**1.62**
Oboe	-10.51	4.96	**5.52**	3.19	-5.96	0.96	0.92	**2.59**
Bassoon	-13.29	0.69	**1.79**	1.36	-7.99	1.06	0.43	**2.19**
Horn	-5.85	1.23	**2.98**	1.64	-5.98	0.77	0.46	**1.83**
Trumpet	-6.23	2.21	**3.19**	2.97	-14.49	**-0.02**	**-0.02**	-3.04
Trombone	-11.89	**0.00**	-0.02	-0.91				
Tuba								
Harp	-12.24	0.00	**0.72**	0.18				
Timpani	-11.50	**3.18**	1.12	1.15				
Unt. perc.	-17.93	7.54	-2.52	**8.64**				
MEAN	-10.14	**2.78**	2.69	2.65	-10.94	0.76	1.04	**1.06**

which we chose to combine into one single violin track. As done in the original baseline of SynthSOD, we also join the piccolo to the flute and the coranglais to the oboe, because the piccolo and coranglais have little data in the dataset and are very similar to the flute and oboe, respectively. All audio in SynthSOD has a sampling rate of 44.1 kHz, which we also use for training. The STFT was computed using a window size of 4096 with a hop size of 1024. We train using 6-second segments of audio and score, which are randomly sampled during the training.

B. Evaluation setup

For evaluation, we use three datasets: the test partition of SynthSOD [16], the Aalto anechoic orchestra dataset [14], and the University of Rochester Multi-Modal Music Performance (URMP) dataset [15]. SynthSOD contains a mixture of ensembles, which are simpler songs with fewer instruments, and orchestral pieces, which contain more instruments. To make the results more comparable to the other datasets, we split the SynthSOD test partition into ensembles (5 or fewer instruments) and orchestral pieces (more than 5 instruments) and show the results separately. Even though we show the results separately, all our models have been trained with both ensembles and orchestral pieces.

The Aalto dataset contains four real recordings of symphonic music, for which note onset and offset annotations are provided in CSV format. The URMP dataset consists of 44 real recordings of songs with between two and five instruments each. We have omitted songs that only contain one kind of instrument from the evaluation. We also omit songs that contain saxophone, since it was not in our training data. URMP audio has a sampling rate of 48 kHz, so we resample it to 44.1 kHz to match the other datasets. URMP provides note onset, frequency, and duration annotations in a CSV format for each individual track. We noticed that every track of the URMP recordings contains a distinct low-frequency noise, which could impact the evaluation results. Therefore, we decided to denoise the recordings using Wiener filtering based on noise statistics calculated from silent regions of the recordings. All of the evaluations in this paper were done using the denoised tracks. We provide the Python script used for the denoising in our Github repository[2], along with the code used to train and evaluate the models.

We use signal-to-distortion ratio (SDR) as a metric to evaluate the separation performance. We use the museval library for computing the SDRs [21]. The SDRs are computed in non-overlapping one-second frames for each instrument. We take the median of the frame-wise SDRs to obtain song-level metrics, and then we take the median of the songs to get the final results. The museval library excludes silent frames from the metrics. Museval defines silent frames as frames where all of the samples are exactly zero. However, for the URMP and Aalto datasets, the silent parts are not exactly zero. Before the evaluation, we take each of the ground-truth tracks and set all of the silent one-second windows to exact zeros, considering a window as silent if the absolute values of all of the samples in the window are below 0.01. We also applied this threshold to the SynthSOD test partition, which explains why the results of the baseline presented in this paper are better than the ones presented in the original SynthSOD paper. However, these results better represent the separation performance of the models since SDR diverges when the target energy approaches zero. Silent frames were having a disproportionate impact on the final SDR results despite being perceptually negligible.

IV. RESULTS

We have retrained the baseline model for a fairer comparison since we had to omit some of the data due to the alignment problems described in Section II-C. Table I shows

[2][Online]. Available: https://github.com/ee7u/score-mss

the SDRs for the ensembles in the test partition of SynthSOD and the URMP dataset. In SynthSOD, the score-informed model achieves 6.70 SDR, which is 1.06 dB better than the baseline's 5.64 SDR. The score-only model's 4.34 SDR is significantly worse than the baseline, which could be expected since the model has less information at its input. In the URMP results, the score-informed model beats the baseline as well, with the score-informed model achieving 1.62 SDR, which is 0.45 dB better than the baseline's 1.17 SDR, but it presents the same generalization issues as the original audio-only baseline. However, the score-only model does clearly better in URMP, achieving 4.49 SDR, which is a 3.32 dB improvement over the baseline, while having the best metrics in all instruments. The score-only model's performance is almost the same in both the synthetic and the real-recording domains, with only a small difference of 0.15 dB between the two datasets. This shows the score-only model's ability to generalize from synthetic data to real data without overfitting to the specific audio characteristics of the synthetic data since it does not use the audio to compute the frequency masks of every instrument.

Table II contains the results for orchestras in the test partition of the SynthSOD dataset and the Aalto dataset. These datasets are much more challenging, which can be seen in the results. In orchestras in SynthSOD, all of the models have similar performance, as all of them fall within only 0.13 dB of one another. Looking at the instrument specific metrics, the score-informed model seems to be the most consistent, but the one outlier score for untuned percussion brings down the mean score a lot. Similarly, in the Aalto dataset, all of the models are separated only by 0.3 dB. The score-only model is the most consistent, but the results for all instruments except bass are bad. Both of our methods improve slightly compared to the baseline in Aalto, but in all of our experiments, the performance of the models in both matched and mismatched conditions indicates that the separation of orchestral pieces remains a challenging task.

V. Conclusion

We presented two ways of incorporating score information into deep neural networks for music source separation. The score-informed model, which uses the concatenation of the audio mixture and the score to create the separation masks, shows a slight improvement over the baseline in both synthetic and real data. We also show that the score-only model, which uses only score information to create the separation masks, can generalize to real data with minimal overfitting. The score-only model clearly beats the baseline model across the board in all instruments by a mean of 3.32 dB SDR in the URMP dataset. The availability of datasets with real recordings is one of the biggest problems in music source separation, so the generalization capability of the score-only model presents a promising direction for future work.

References

[1] O. Slizovskaia, G. Haro, and E. Gomez, "Conditioned Source Separation for Musical Instrument Performances," IEEE/ACM Transactions on Audio, Speech, and Language Processing, PP. 1-1. 10.1109/TASLP.2021.3082331, 2021.

[2] M. Schwabe, and M. Heizmann, "Improved Separation of Polyphonic Chamber Music Signals by Integrating Instrument Activity Labels," in IEEE Access, vol. 11, pp. 42999-43007, 2023.

[3] M. Miron, J. Janer and E. Gómez, "Monaural Score-Informed Source Separation for Classical Music Using Convolutional Neural Networks," International Society for Music Information Retrieval Conference, 2017.

[4] S. Ewert, and M. B. Sandler, "Structured dropout for weak label and multi-instance learning and its application to score-informed source separation," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing, New Orleans, LA, USA, 2017, pp. 2277-2281.

[5] J. Fritsch, and M. D. Plumbley, "Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 888-891, 2013.

[6] S. Ewert, and M. Müller, "Using score-informed constraints for NMF-based source separation," 2012 IEEE International Conference on Acoustics, Speech and Signal Processing, Kyoto, Japan, 2012, pp. 129-132.

[7] S. Ewert, B. Pardo, M. Müller, and M. D. Plumbley, "Score-Informed Source Separation for Musical Audio Recordings: An overview," in IEEE Signal Processing Magazine, vol. 31, no. 3, pp. 116-124, May 2014.

[8] Z. Duan, and B. Pardo, "Soundprism: An Online System for Score-Informed Source Separation of Music Audio," in IEEE Journal of Selected Topics in Signal Processing, vol. 5, no. 6, pp. 1205-1215, Oct. 2011.

[9] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Music source separation in the waveform domain," 2019, arXiv preprint arXiv:1911.13254.

[10] A. Défossez, "Hybrid spectrogram and waveform source separation," in Proc. International Society for Music Information Retrieval Conference 2021 Workshop Music Source Separation, 2021.

[11] S. Rouard, F. Massa, and A. Défossez, "Hybrid Transformers for Music Source Separation," IEEE International Conference on Acoustics, Speech and Signal Processing, 1-5, 2022.

[12] Y. Luo and J. Yu, "Music Source Separation With Band-Split RNN," IEEE/ACM Transactions on Audio, Speech, and Language Processing. PP. 1-10. 10.1109/TASLP.2023.3271145, 2023.

[13] G. Fabbro, et al., "The Sound Demixing Challenge 2023 – Music Demixing Track," Transactions of the International Society for Music Information Retrieval, vol. 7, no. 1, 2024, p. 63–84, 2023.

[14] J. Pätynen, V. Pulkki, and T. Lokki, "Anechoic recording system for symphony orchestra," Acta Acustica united with Acustica, vol. 94, pp. 856–865, 11 2008.

[15] B. Li, X. Liu, K. Dinesh, Z. Duan, and G. Sharma, "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications," IEEE Transactions on Multimedia, vol. 21, no. 2, pp. 522–535, 2019.

[16] J. Garcia-Martinez, D. Diaz-Guerra, A. Politis, T. Virtanen, J. Carabias-Orti, and P. Vera-Candeas, "SynthSOD: Developing an Heterogeneous Dataset for Orchestra Music Source Separation," IEEE Open Journal of Signal Processing, 6, 129–137, 2025.

[17] S. Sarkar, E. Benetos, and M. Sandler, "EnsembleSet: a new high quality synthesised dataset for chamber ensemble separation," In International Society for Music Information Retrieval Conference, (pp. 625-632), 2022.

[18] R. Sawata, S. Uhlich, S. Takahashi, and Y. Mitsufuji, "All for one and one for all: Improving music separation by bridging networks," in IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 51–55, 2021.

[19] M. Pariente, S. Cornell, J. Cosentino, S. Sivasankaran, E. Tzinis, J. Heitkaemper, M. Olvera, F.-R. Stöter, M. Hu, J. M. Martín-Dõnas, D. Ditter, A. Frank, A. Deleforge, and E. Vincent, "Asteroid: the PyTorch-based audio source separation toolkit for researchers," in Proc. Interspeech, 2020.

[20] J. Carabias-Orti, J. Rodriguez-Serrano, M. Vera-Candeas, N. Ruiz-Reyes, and M. Cañadas-Quesada, "An audio-to-score alignment framework using spectral factorization and dynamic time warping," in International Society for Music Information Retrieval Conference, pp. 742-748, 2015.

[21] F.-R. Stöter and A. Liutkus, "Museval 0.3.0," Zenodo, Aug. 2019, doi: 10.5281/zenodo.3376621.