IMPROVING MUSIC SOURCE SEPARATION BASED ON DEEP NEURAL NETWORKS THROUGH DATA AUGMENTATION AND NETWORK BLENDING

Stefan Uhlich[1], Marcello Porcu[1], Franck Giron[1], Michael Enenkl[1], Thomas Kemp[1], Naoya Takahashi[2] and Yuki Mitsufuji[2]

[1] Sony European Technology Center (EuTEC), Stuttgart, Germany
[2] Sony Corporation, Audio Technology Development Department, Tokyo, Japan

ABSTRACT

This paper deals with the separation of music into individual instrument tracks which is known to be a challenging problem. We describe two different deep neural network architectures for this task, a feed-forward and a recurrent one, and show that each of them yields themselves state-of-the art results on the SiSEC DSD100 dataset. For the recurrent network, we use data augmentation during training and show that even simple separation networks are prone to overfitting if no data augmentation is used. Furthermore, we propose a blending of both neural network systems where we linearly combine their raw outputs and then perform a multi-channel Wiener filter post-processing. This blending scheme yields the best results that have been reported to-date on the SiSEC DSD100 dataset.

Index Terms— Music source separation (MSS), Deep neural network (DNN), Long-short term memory (LSTM), Blending

1. INTRODUCTION

In this paper, we study the problem of separating music into instrument tracks. In particular, we study the separation into a *vocals* and *accompaniment* track, or, more fine-grained, into tracks for *vocals*, *bass*, *drums* and *other*. Various applications require such track estimates ranging from Karaoke systems which use a separation into an instrumental and vocal track, see for example [1, 2], to upmixing where one tries to obtain a multi-channel version of the song, see for example [3–5].

Despite its difficult nature, *music source separation* (MSS) has received increasing interest over the last years. Especially the professionally-mixed music separation task [6] – called MUS –, which is a subtask of the regularly held *Signal Separation Evaluation Campaign* (SiSEC), has helped to stir this interest. For the 2015 version of this contest, the new *MSD100 dataset* was prepared consisting of a *Dev* and *Test* part with 50 songs each [7]. For every song, the mixture and its four sources *bass*, *drums*, *other* and *vocals* are available, and this dataset allowed for the first time to evaluate different source separation methods on a wide variety of music genres. The MSD100 dataset has been further improved for the latest SiSEC 2016 contest by using digital audio workstation software and mixing settings that professional audio engineers use. In order to differentiate this dataset from the previous version, it is called *DSD100*[1].

The best results for the MUS task for SiSEC 2015 and SiSEC 2016 were obtained by approaches that used deep neural networks (DNN)s [8–10] and this observation for MSS is in line with many other applications where DNNs have outperformed the previous best systems[2]. Using training corpora (e.g., the Dev part of DSD100), it is possible to learn neural networks that extract a particular target instrument from the mixture. Furthermore, DNNs have also shown very good performance for the related problems of extracting speech from music in [12] or for vocals separation from music in [13–15].

The contribution of this paper is three-fold: First, we will describe in detail our submissions to the latest SiSEC contest. In particular, we give results for our feed-forward approach from [10] with a *multi-channel Wiener filter* (MWF) post-processing, as we used only *single channel Wiener filters* (SWF) up to now. We also investigate the use of a recurrent neural network for the extraction of the instruments. This recurrent structure has the advantage that it allows to better take into account the context information from neighbouring mixture frames, which is important to capture information about the temporal structure of the song. A second contribution of the paper is the proposal to use data augmentation during training. This is especially important for the training of the recurrent neural networks as we only used the Dev part of DSD100 to learn these networks and do not utilize additional data. We show that data augmentation helps to learn separating networks that generalize better. A final contribution is to show that a blending of the raw network outputs before the MWF post-processing can considerably improve the separation performance for all sources and the results on the DSD100 dataset are the best that have been reported so far. Blending, also known as fusion, of MSS algorithms is a quite new topic in the source separation literature and was discussed in [16] and in [17, 18] where mask estimates of different systems are combined for speech enhancement and vocal separation, respectively. More recently, [19] considers a neural network approach to combine the output of two other neural networks that estimate soft and binary separation masks and [20] discusses a linear combination of the estimated masks from different DNN systems. Furthermore, [21, 22] discusses the training of multi-context networks and [23] a cooperative deep stacking approach where two networks with different inputs are combined at the layer-level. In contrast to these approaches, we blend the raw outputs of two different network structures, a feed-forward and recurrent one, and additionally perform a Wiener filter post-processing of the enhanced separations. This additional step allows to further improve the separations as the MWF is computed from better source estimates, which we obtain after the blending.

This paper is structured as follows: In Sec. 2, we will show two different network structures for the extraction of instruments from music. In particular, we describe a feed-forward and a recurrent architecture. In order to prove the effectiveness of data augmentation, we also trained a recurrent architecture without augmentation and report its performance in Sec. 3. Sec. 4 shows that a blending of the two networks yields a new system that has considerably better separation performance. Finally, Sec. 5 compares our approaches to the DNN system from [9] and to three *non-negative matrix factorization* (NMF) systems known from literature (sparse NMF [24, 25], discriminative NMF [26] and DeepNMF [27]) before we conclude this paper in Sec. 6.

The following notations are used throughout this paper: $\mathbf{x}$ denotes a column vector and $\mathbf{X}$ a matrix where in particular $\mathbf{I}$ is the identity matrix. The matrix transpose and Euclidean norm are denoted by $(.)^T$ and $\|.\|$, respectively.

[1] All results in this paper have been obtained with this improved dataset.

[2] A good general overview of DNNs and their success can be found in the review paper [11] and the references therein.

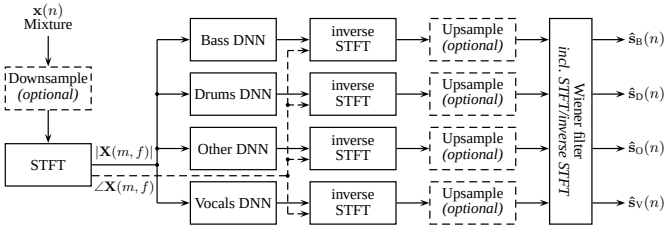
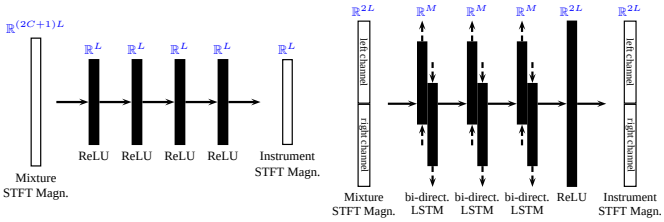


Fig. 1: General DNN approach for MSS of music



(a) Feed-forward (here: FNN-2) (b) BLSTM (here: BLSTM-3)

Fig. 2: DNN architectures for the extraction of an instrument

2. MSS USING DEEP NEURAL NETWORKS

In the following, we will introduce the basic SiSEC MUS problem. Let $\mathbf{x}(n) \in \mathbb{R}^2$ denote the stereo mixture in the time domain which is known to be composed of the four sources *bass* $\mathbf{s}_\mathrm{B}(n)$, *drums* $\mathbf{s}_\mathrm{D}(n)$, *other* $\mathbf{s}_\mathrm{O}(n)$ and *vocals* $\mathbf{s}_\mathrm{V}(n)$ such that

$$\mathbf{x}(n) = \mathbf{s}_\mathrm{B}(n) + \mathbf{s}_\mathrm{D}(n) + \mathbf{s}_\mathrm{O}(n) + \mathbf{s}_\mathrm{V}(n) = \sum\nolimits_{i\in\mathcal{I}} \mathbf{s}_i(n). \quad (1)$$

with $\mathcal{I} := \{\mathrm{B}, \mathrm{D}, \mathrm{O}, \mathrm{V}\}$. The goal of MSS is to retrieve good stereo source estimates $\hat{\mathbf{s}}_i(n)$ such that they are as close as possible to the true sources $\mathbf{s}_i(n)$. A popular performance measure to judge the quality of the separation is *BSS Eval* [28], which is also used in the SiSEC MUS task to compare the different approaches.

Finally, most approaches perform the separation in the *short-time Fourier transform* (STFT) domain and we will denote by $\mathbf{X}(m, f) \in \mathbb{C}^2$, $\mathbf{S}_i(m, f)$ and $\hat{\mathbf{S}}_i(m, f)$ the mixture, the sources and the estimates in the STFT domain, respectively, where m gives the frame index and f the frequency bin index.

We will now explain in more detail the general DNN approach for the MUS problem before we describe in Sec. 2.2 and 2.3 the feed-forward and recurrent approaches.

2.1. General DNN Approach

Fig. 1 shows the general DNN approach for MSS that we use. Instead of using one DNN to estimate the STFT magnitudes, we use an individual DNN for each instrument that is trained to extract this instrument from a mixture, which allows us to better scale to mixtures with different constituting instruments. After an optional downsampling[3], the STFT of the music that we want to separate is computed. We pass the STFT magnitudes – possibly together with some preceding/succeeding context frames – through the DNNs and obtain an estimate of the STFT magnitudes for each source. These magnitude estimates are then combined together with the phase of the mixture to compute the inverse STFT. Finally, a SWF or a MWF post-processing is applied to enhance the separation performance [9,29,30]. Applying the SWF/MWF can be seen as a post-processing which ensures that the sum of the four estimates gives the original mixture. It reduces considerably interferences from other sources and separation artifacts and, hence, SWF/MWF is an important step of the separation. The multi-channel Wiener filter assumes the signal model [9, 29, 30]

$$\mathbf{X}(m, f) = \mathbf{S}_i(m, f) + \mathbf{Z}_i(m, f) \quad (2)$$

with $i \in \mathcal{I}$ and $\mathbf{Z}_i(m, f) = \sum_{j\in\mathcal{I}\setminus i} \mathbf{S}_j(m, f)$ where each STFT time-frequency bin $\mathbf{S}_i(m, f)$ is complex Gaussian with zero-mean and covariance matrix $v_i(m, f)\mathbf{R}_i(f)$. $v_i(m, f)$ is the *power-spectral density* (PSD) and $\mathbf{R}_i(f)$ the time-invariant[4] spatial covariance matrix, respectively. The *minimum mean squared error* estimator for $\mathbf{S}_i(m, f)$ from $\mathbf{X}(m, f)$ is well known and given by [31]

$$\hat{\mathbf{S}}_i(m, f) = v_i(m, f)\mathbf{R}_i(f) \left(\sum\nolimits_{j\in\mathcal{I}} v_j(m, f)\mathbf{R}_j(f)\right)^{-1} \mathbf{X}(m, f). \quad (3)$$

[3]We use a downsampling to 32kHz only for the feed-forward approach in Sec. 2.2. Please refer to [10] for more details.

[4]Hence, we assume that we have a time-invariant, convolutive mixture process which is reasonable for the majority of music mixtures.

In order to apply the MWF, we need to estimate the PSDs $v_i(m, f)$ and spatial covariance matrices $\mathbf{R}_i(f)$ which we do in our case from the raw output of each instrument DNN. In particular, we estimate them from M consecutive frames by

$$\hat{v}_i(m, f) = \frac{1}{2}\|\hat{\mathbf{S}}_i(m, f)\|^2, \quad \hat{\mathbf{R}}_i(f) = \frac{\sum_{m=1}^{M} \hat{\mathbf{S}}_i(m, f)\hat{\mathbf{S}}_i(m, f)^H}{\sum_{m=1}^{M} \hat{v}_i(m, f)}, \quad (4)$$

as was proposed in [9, 32]. The estimator for the spatial covariance matrix can be thought of being a weighted version of the classical *maximum likelihood* estimator. More weight is put on time-frequency bins with high energy as we can assume that we then have a better signal-to-interference ratio. We will see in Sec. 2.2 that the MWF improves the separation performance. Especially when listening to the separation results, one can observe a much more stable stereo location of the extracted sources and no disturbing *flanging effects* appear.

2.2. Feed-Forward Networks (FNN)

The first approach uses the feed-forward architecture as described in [10] and shown schematically in Fig. 2(a). In this paper, we will focus on the changes with respect to [10] and use the same notation as we used there. The interested reader is referred to [10] for more details about the data preparation, weight initialization and layerwise training. We consider two different sets of networks:

FNN-1 For each instrument, we train a *rectified linear unit* (ReLU) [33] network with $K = 3$ layers from $P = 2 \cdot 10^6$ training samples. The training material consists of short instrument loops, which are independent from the Dev or Test part of DSD100. The training samples are created by randomly selecting loops for each instrument and mixing them with random amplitudes. The number of non-overlapping context frames is $C = 3$ and, as we use a FFT size of 1 024, the input vector has length $(2C + 1) \cdot 513 = 3\,591$. Finally, this set of networks uses a single-channel Wiener filter post-processing. Please note that the results from this set of networks was our submission to SiSEC 2015 and is denoted in [7, 9] as *UHL1*.

FNN-2 This set of neural networks is newly trained. Additionally to the dataset of FNN-1, we use non-bleeding stems from MedleyDB [34] and the material that is contained in the Dev part of DSD100. Furthermore, it uses input frames with 50% overlap and $C = 8$ context frames. A *principal component analysis* (PCA) [35] is used to half the size of the input vector. This allows us to train with more training samples, namely $P = 1.2 \cdot 10^7$. Furthermore, FNN-2 uses $K = 4$ ReLU layers while all other settings and the training procedure are the same as for FNN-1.

Table 1 compares the two sets of DNNs on the Test part of DSD100 where the values are obtained by first averaging the *signal-to-distortion ratio* (SDR) values [28] for each song and then computing the median over all 50 songs. This table contains also a *lower baseline BL* and *upper baseline BU*, where BL uses the original mixture scaled by $\frac{1}{4}$ as separations and BU gives the oracle performance of an ideal ratio mask. In order to create an estimate for the accompaniment, i.e., for all sources except the vocals, we compute $\hat{\mathbf{s}}_\mathrm{A}(n) = \mathbf{x}(n) - \hat{\mathbf{s}}_\mathrm{V}(n)$.

From this table, we can observe that FNN-2 exhibits on average a 0.6 dB better SDR than FNN-1. The two main differences between

Network	SDR in dB					Comments
	Bass	Drums	Other	Vocals	Acco.	
BL	0.72	0.95	1.43	0.35	6.82	lower baseline: mix as separations
BU	6.26	7.96	7.76	10.16	16.38	upper baseline: ideal ratio mask
FNN-1	2.22	3.08	2.48	3.63	10.19	*UHL1* from SiSEC 2015 [7,9]
FNN-2	2.54	3.75	2.92	4.47	11.12	
BLSTM-1	2.30	3.71	2.98	3.69	10.33	one BLSTM layer
BLSTM-2	2.77	3.78	3.44	4.91	11.35	two BLSTM layers
BLSTM-3	2.89	4.00	3.24	4.86	11.26	three BLSTM layers

Table 1: FNN and BLSTM networks on Test part of DSD100

the two networks is that FNN-2 uses additionally the Dev part of DSD100 and a MWF instead of a SWF. We also trained a set of networks with only each of these changes and could observe that using the Dev part improves the SDR on average by 0.4 dB and using the MWF improves by another 0.2 dB.

2.3. Bidirectional LSTM Networks

The second approach uses a recurrent neural network architecture with *bidirectional LSTM* (BLSTM) layers [36] and is shown in Fig. 2(b). Compared to traditional recurrent neural networks, (B)LSTM networks have the advantage that they do not suffer from the vanishing/exploding gradient problem and are, therefore, quite popular for problems which require memory. For our separation problem, such a recurrent approach has the advantage that we better take into account the context information than using supervectors with neighbouring magnitude frames as we did in Sec. 2.2. The network can memorize longer dependencies and this helps to improve the MSS performance.

We trained[5] three different sets of networks which differ in their number of BLSTM layers. Each BLSTM layer consists of 250 forward and 250 backward LSTM cells whose output is concatenated to form the overall output of the layer. The instrument loops that we used for the training of the FNNs in Sec. 2.2 are too short for the sequence training of the BLSTM networks. Therefore, we trained them solely on the Dev part of DSD100 and data augmentation is used to avoid overfitting (cf. Sec. 3). The input to the BLSTM networks are stereo magnitude frames from the left and right channel which are obtained by using a frame size of $1\,024$ samples with 50% overlap. The output is the estimated stereo magnitude frame of the target instrument. We again post-process the raw outputs by a MWF to enhance the results.

Table 1 shows the results of the three BLSTM networks. We can observe that the two deeper networks with $K = 2$ and $K = 3$ BLSTM layers show a considerably better performance than BLSTM-1. Comparing BLSTM-2 and BLSTM-3 to FNN-2, we can observe that BLSTM-3 exhibits a more consistent improvement for all instruments over FNN-2 (cf. the SDR value for drums) and we therefore choose this network for the blending in Sec. 4.

3. DATA AUGMENTATION DURING TRAINING

It is well known that data augmentation can help considerably to improve the performance of DNNs, see for example [40, 41] where augmentation was investigated for music information retrieval tasks. We use the following data modifications on-the-fly when we construct a training mini-batch sequence for the BLSTMs:

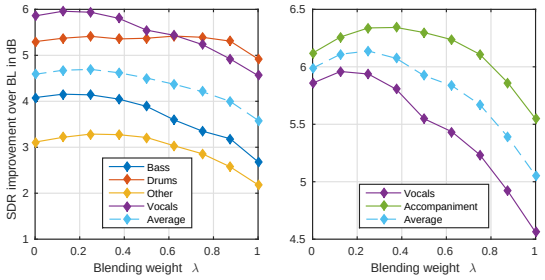
- random swapping left/right channel for each instrument,
- random scaling with uniform amplitudes from $[0.25, 1.25]$,
- random chunking into sequences for each instrument, and,
- random mixing of instruments from different songs.

Each mini-batch consists of ten sequences where each sequence has a length of 500 stereo STFT magnitude frames. In order to prove the effectiveness of data augmentation, we trained the BLSTM-1 network for the extraction of the vocals also without augmentation and the results are shown in Table 2 where the accompaniment is again estimated as $\hat{\mathbf{s}}_{\mathrm{A}}(n) = \mathbf{x}(n) - \hat{\mathbf{s}}_{\mathrm{V}}(n)$. This table gives the raw SDR values without a SWF/MWF post-processing as we want to show more clearly the effect of data augmentation.

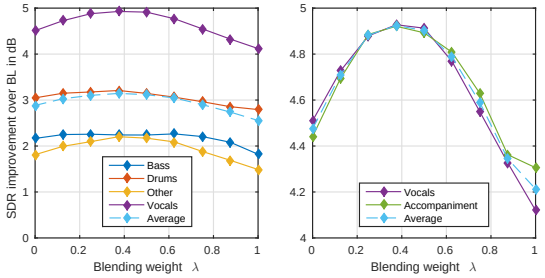
[5] We used Lasagne [37] and Theano [38, 39].

Instr.	Network	SDR in dB (Raw outputs)		
		Dev	Test [All]	Test [New artists]
Vocals	BL	0.91	0.35	0.77
	BLSTM-1 w/o data augm.	7.13	3.37	3.19
	BLSTM-1 with data augm.	6.19	3.59	3.79
Accomp.	BL	6.57	6.82	6.49
	BLSTM-1 w/o data augm.	13.33	9.71	9.53
	BLSTM-1 with data augm.	12.23	9.93	9.64

Table 2: Effect of data augmentation for BLSTM-1



(a) Dev part of DSD100



(b) Test part of DSD100

Fig. 3: Effect of blending ($\lambda = 0$: BLSTM-3, $\lambda = 1$: FNN-2)

From Table 2, we can see that the augmentation is beneficial. Vocals and accompaniment gain on average 0.2 dB if we consider all Test songs and even 0.35 dB if we only consider the subset of Test songs where there is not a song of the same artist in the Dev part. The results on this subset can be used to better estimate the generalization performance of a network and we can see that data augmentation helps to learn networks that generalize better to new artists. Please note that we expect the data augmentation to be even more beneficial for more complex networks like BLSTM-2 and BLSTM-3.

4. BLENDING OF NETWORKS

In order to further improve the MSS performance, we propose to combine the results of the feed-forward and the BLSTM network. It is well known that a blending of different systems can improve the performance if the errors of each individual system are uncorrelated and, e.g., has been successfully used in recommender systems [42, 43]. As we combine two networks which differ in their network structure and their training material, we can assume that both systems are different enough such that a blending of them is beneficial.

In particular, we use a time-invariant blending which takes the form

$$\hat{\mathbf{s}}_{i,\mathrm{BLEND}}(n) = \lambda \hat{\mathbf{s}}_{i,\mathrm{FNN}}(n) + (1-\lambda)\hat{\mathbf{s}}_{i,\mathrm{BLSTM}}(n), \qquad (5)$$

i.e., we linearly blend the raw DNN outputs for each instrument $i \in \mathcal{I}$. Finally, we use a MWF post-processing to enhance the results. Fig. 3 shows the SDR improvement with respect to BL for a blending of the two best systems from Sec. 2.2 and 2.3, i.e., FNN-2 and BLSTM-3. We choose the blending weight λ to have a value of $\lambda = 0.25$ as this is the best average improvement value on the Dev part of DSD100. Looking at the the Test part, we can observe that all instruments improve by this choice and that it yields on average an SDR improvement of 0.2 dB compared to BLSTM-3. In particular, for vocals and accompaniment the gain is largest with 0.4 dB. As we were interested in the generalization performance of the blending

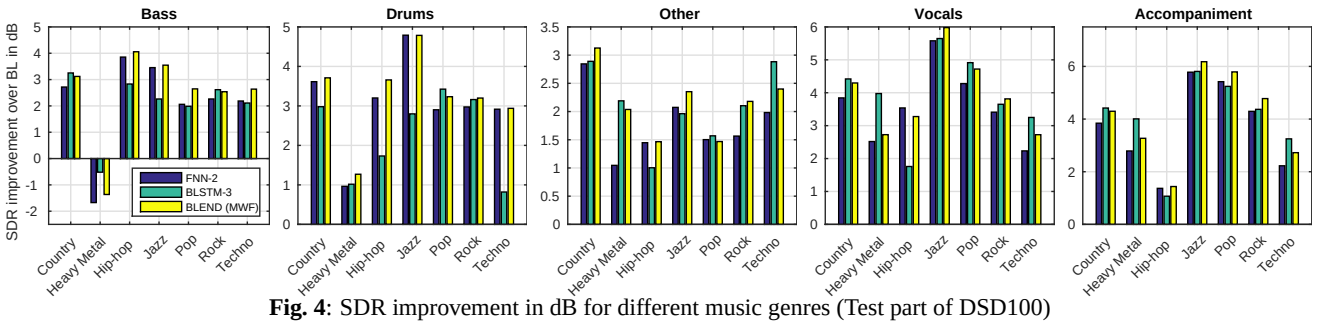


Fig. 4: SDR improvement in dB for different music genres (Test part of DSD100)

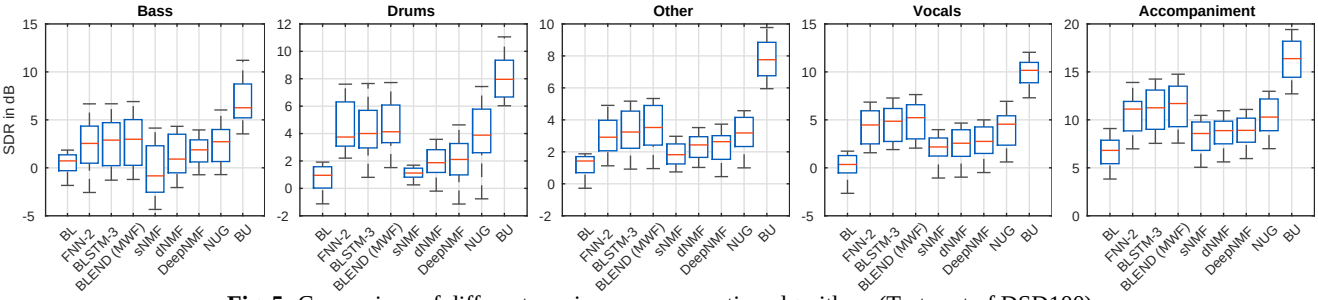


Fig. 5: Comparison of different music source separation algorithms (Test part of DSD100)

	Approach	SDR in dB Bass	Drums	Other	Vocals	Acco.	Comments
Single-channel methods	BLEND (SWF)	2.76	3.93	3.37	5.13	11.53	$\lambda = 0.25$
	sNMF [24, 25]	-0.84	1.12	1.82	2.17	8.58	$Q = 25$
	dNMF [26]	0.91	1.87	2.43	2.56	8.88	$Q = 25$
	DeepNMF [27]	1.88	2.11	2.64	2.75	8.90	$Q = 25$
Multi-channel methods	BLEND (MWF)	2.98	4.13	3.52	5.23	11.70	$\lambda = 0.25$
	NUG [9]	2.72	3.89	3.18	4.55	10.29	

Table 3: Comparison on Test part of DSD100

approach, we also performed an informal listening test with popular Hits and we could observe a significant improvement for our blending system. Especially the temporal stability of the extracted sources is improved, i.e., attack/decay parts of sources are not missing in the separated instruments.

Our blending scheme (5) can be seen as an extension of *learned temporal fusion*, which was proposed in [17]. Instead of linearly combining the systems after the MWF, we blend the raw outputs of each DNN and perform afterwards a MWF post-processing. This final MWF helps to reduce the interference and achieves better results as we obtain better source estimates which in turn allows a better multichannel Wiener filtering. Comparing the two schemes, we can observe that our fusion is on average 0.1 dB better for SDR and 0.3 dB for the *signal-to-interference ratio* (SIR) than the *learned temporal fusion* scheme proposed in [17].

Finally, we also studied the effect of blending for the different music genres in DSD100. The results are shown in Fig. 4 and we can observe that the blending of FNN-2 and BLSTM-3 is often even better than the best individual networks (20 out of 35 times) which shows the effectiveness of the blending.

5. COMPARISON TO OTHER APPROACHES

In this section, we will now compare our blended MSS system from Sec. 4 to four other well known approaches from the literature:

- *sNMF*: Supervised *sparse non-negative matrix factorization* (sNMF) approach [24, 25] with sparsity factor $\mu = 5$.
- *dNMF*: Supervised *discriminative non-negative matrix factorization* (dNMF) approach [26] where we use a discriminative cost function to learn better basis vectors from the Dev songs.
- *DeepNMF*: Non-negative deep network architecture which results from unfolding NMF iterations and untying their parameters [27].
- *NUG*: This approach uses a complex Gaussian time-frequency model where the spectral updates are computed with the help of multi-channel DNNs [9]. Compared to our FNN approach, this approach iteratively updates the spatial and spectral estimates in an *expectation-maximization* (EM) like manner whereas we use the Wiener filter only as post-processing and one can think of our FNN to be the special case of doing only a single EM iteration. The *NUG* system that we consider here has shown very good performance in [9] and was denoted there as *NUG1*[6].

The results on the Test part of DSD100 are given in Table 3 and shown as box plots in Fig. 5. For the three NMF approaches, Q basis vectors are learned for each instrument and Dev song of DSD100 which we concatenate together to obtain finally $50 \cdot Q$ basis vectors per instrument. We tried different numbers Q of basis vectors per song, namely $Q \in \{5, 10, 15, 20, 25, 30\}$ and obtained the best average performance for $Q = 25$ which are reported in Table 3. Among the NMF methods, we can observe that the DeepNMF method from [27] performs best. As all NMF systems are single-channel approaches, we also rerun our blending approach with a SWF to allow a fairer comparison. By using the SWF, we loose on average 0.2 dB SDR compared to the MWF but the blending approach is still better than the DeepNMF approach.

Furthermore, it is also interesting to compare BLEND (MWF) to FNN-1 which was the best system of the SiSEC 2015 contest [7]. We can observe that we could gain 1.1 dB SDR since the last SiSEC competition, which is a considerable improvement.

Overall, we can conclude that the DNN systems NUG and BLEND (MWF) perform best where in particular our BLEND (MWF) system achieves on average a 0.4 dB better SDR than NUG.

6. CONCLUSIONS

In this paper, we described different approaches for the music source separation problem. We could obtain the best results by a linear blending of the outputs of a feed-forward and a recurrent bi-directional LSTM network where the recurrent network was trained with data augmentation. Compared to the individual networks, we gain 0.2 dB SDR by the blending and these results are the best that have been reported so far on the DSD100 dataset.

We participated with our systems[7] in the SiSEC 2016 MUS contest and the results of the SiSEC MUS 2016 competition can be found in [6].

[6] We thank the authors of [9] for providing us with these results.

[7] We submitted the following three systems: FNN-2 as *UHL1*, BLSTM-3 as *UHL2* and BLEND (MWF) as *UHL3*.

7. REFERENCES

[1] Z. Rafii and B. Pardo, "Repeating pattern extraction technique (REPET): A simple method for music/voice separation," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 21, no. 1, pp. 73–84, 2013.

[2] J.-L. Durrieu, B. David, and G. Richard, "A musically motivated mid-level representation for pitch estimation and musical audio source separation," *IEEE Journal on Selected Topics on Signal Processing*, vol. 5, pp. 1180–1191, 2011.

[3] H. Shim, J. S. Abel, and K.-M. Sung, "Stereo music source separation for 3-D upmixing," in *127th AES Convention*, 2009.

[4] D. FitzGerald, "Upmixing from mono - a source separation approach," *Proc. Digital Signal Processing*, 2011.

[5] D. FitzGerald, "The good vibrations problem," *134th AES Convention, e-brief*, 2013.

[6] "SiSEC MUS Homepage," https://sisec.inria.fr/home/2016-professionally-produced-music-recordings/.

[7] N. Ono, Z. Rafii, D. Kitamura, N. Ito, and A. Liutkus, "The 2015 signal separation evaluation campaign," in *Proc. LVA/ICA*, 2015, pp. 387–395.

[8] A. A. Nugraha, A. Liutkus, and E. Vincent, "Multichannel audio source separation with deep neural networks," INRIA Technical Report, 2015.

[9] A. A. Nugraha, A. Liutkus, and E. Vincent, "Multichannel music separation with deep neural networks," in *Proc. EUSIPCO*, 2016.

[10] S. Uhlich, F. Giron, and Y. Mitsufuji, "Deep neural network based instrument extraction from music," in *Proc. ICASSP*, 2015, pp. 2135–2139.

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

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

[13] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Singing-voice separation from monaural recordings using deep recurrent neural networks.," in *Proc. ISMIR*, 2014, pp. 477–482.

[14] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Joint optimization of masks and deep recurrent neural networks for monaural source separation," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 23, no. 12, pp. 2136–2147, 2015.

[15] A. J. Simpson, G. Roma, and M. D. Plumbley, "Deep karaoke: Extracting vocals from musical mixtures using a convolutional deep neural network," in *Proc. LVA/ICA*, 2015, pp. 429–436.

[16] J. Le Roux, S. Watanabe, and J. R. Hershey, "Ensemble learning for speech enhancement," in *Proc. WASPAA*, 2013, pp. 1–4.

[17] X. Jaureguiberry, G. Richard, P. Leveau, R. Hennequin, and E. Vincent, "Introducing a simple fusion framework for audio source separation," in *Proc. MLSP*, 2013, pp. 1–6.

[18] X. Jaureguiberry, E. Vincent, and G. Richard, "Fusion methods for speech enhancement and audio source separation," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 24, no. 7, pp. 1266–1279, 2016.

[19] E. M. Grais, G. Roma, A. J. R. Simpson, and M. D. Plumbley, "Combining mask estimates for single channel audio source separation using deep neural networks," in *Proc. Interspeech*, 2016.

[20] E. M. Grais, G. Roma, A. J. R. Simpson, and M. D. Plumbley, "Single-channel audio source separation using deep neural network ensembles," in *140th AES Convention*, 2016.

[21] X.-L. Zhang and D. Wang, "Multi-resolution stacking for speech separation based on boosted DNN," in *Proc. Interspeech*, 2015, pp. 1745–1749.

[22] X.-L. Zhang and D. Wang, "A deep ensemble learning method for monaural speech separation," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 24, no. 5, pp. 967–977, 2016.

[23] W. Jiang, S. Liang, L. Dong, H. Yang, W. Liu, and Y. Wang, "Cross-domain cooperative deep stacking network for speech separation," in *Proc. ICASSP*. IEEE, 2015, pp. 5083–5087.

[24] J. Eggert and E. Korner, "Sparse coding and NMF," in *Proc. Neural Networks*, 2004, vol. 4, pp. 2529–2533.

[25] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural computation*, vol. 21, no. 3, pp. 793–830, 2009.

[26] F. Weninger, J. Le Roux, J. R. Hershey, and S. Watanabe, "Discriminative NMF and its application to single-channel source separation," in *Proc. Interspeech*, 2014, pp. 865–869.

[27] J. Le Roux, J. R. Hershey, and F. Weninger, "Deep NMF for speech separation," in *Proc. ICASSP*, 2015, pp. 66–70.

[28] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[29] N. Q. Duong, E. Vincent, and R. Gribonval, "Under-determined reverberant audio source separation using a full-rank spatial covariance model," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 18, no. 7, pp. 1830–1840, 2010.

[30] A. Ozerov, E. Vincent, and F. Bimbot, "A general flexible framework for the handling of prior information in audio source separation," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 4, pp. 1118–1133, 2012.

[31] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume 1: Estimation Theory*, Prentice-Hall, 1993.

[32] S. Sivasankaran, A. A. Nugraha, E. Vincent, J. A. Morales-Cordovilla, S. Dalmia, I. Illina, and A. Liutkus, "Robust ASR using neural network based speech enhancement and feature simulation," in *Proc. ASRU*, 2015, pp. 482–489.

[33] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier networks," *Proc. AISTATS*, vol. 15, pp. 315–323, 2011.

[34] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "MedleyDB: A multitrack dataset for annotation-intensive MIR research.," in *Proc. ISMIR*, 2014, pp. 155–160.

[35] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification and Scene Analysis*, John Wiley & Sons, 2001.

[36] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[37] "Lasagne GitHub," https://github.com/Lasagne/Lasagne.

[38] "Theano GitHub," https://github.com/Theano/Theano.

[39] The Theano Development Team, "Theano: A python framework for fast computation of mathematical expressions," *arXiv preprint arXiv:1605.02688*, 2016.

[40] J. Schlüter and T. Grill, "Exploring data augmentation for improved singing voice detection with neural networks," in *Proc. ISMIR*, 2015.

[41] B. McFee, E. J. Humphrey, and J. P. Bello, "A software framework for musical data augmentation," in *Proc. ISMIR*, 2015.

[42] J. Bennett and S. Lanning, "The Netflix prize," in *Proc. KDD Cup and Workshop*, 2007, vol. 2007, p. 35.

[43] R. M. Bell and Y. Koren, "Lessons from the Netflix prize challenge," *ACM SIGKDD Explorations Newsletter*, vol. 9, no. 2, pp. 75–79, 2007.