Sampling Frequency Independent Dialogue Separation

Jouni Paulus*
Fraunhofer Institute for Integrated Circuits IIS, and
International Audio Laboratories Erlangen†
Erlangen, Germany

Matteo Torcoli
Fraunhofer Institute for Integrated Circuits IIS, and
International Audio Laboratories Erlangen†
Erlangen, Germany

***Abstract*—In some DNNs for audio source separation, the relevant model parameters are independent of the sampling frequency of the audio used for training. Considering the application of dialogue separation, this is shown for two DNN architectures: a U-Net and a fully-convolutional model. The models are trained with audio sampled at 8 kHz. The learned parameters are transferred to models for processing audio at 48 kHz. The separated audio sources are compared with the ones produced by the same model architectures trained with 48 kHz versions of the same training data. A listening test and computational measures show that there is no significant perceptual difference between the models trained with 8 kHz or with 48 kHz. This transferability of the learned parameters allows for a faster and computationally less costly training. It also enables using training datasets available at a lower sampling frequency than the one needed by the application at hand, or using data collections with multiple sampling frequencies.**

***Index Terms*—deep learning, dialogue separation, sampling frequency**

I. Introduction

When developing and training a deep neural network (DNN) for audio signal processing, the input signal sampling frequency is normally fixed and defined by the target application, e.g., 8 or 16 kHz for speech, 44.1 kHz for music, and 48 kHz for broadcast applications. Alternatively, the sampling frequency of the model is dictated by the available training data. If we want to process signals at a sampling frequency different from the one in training, signal re-sampling may be needed, a new model for the new sampling frequency needs to be trained, or a different paradigm needs to be applied, e.g., [1].

Focusing on the dialogue separation application [2], [3], this paper presents the observation of sampling frequency independence of relevant model parameters in certain audio source separation DNN architectures. In these architectures the trainable parameters are effectively independent of the actual signal sampling frequency. This opens the possibilities for training a single model with multiple datasets of different sampling frequencies, or training and deploying a model in different sampling frequencies. Additionally, training at a lower sampling frequency is computationally cheaper and faster, possibly reducing the carbon footprint.

*Now with WS Audiology, Erlangen, Germany.
†International Audio Laboratories Erlangen is a joint institution of Universität Erlangen-Nürnberg and Fraunhofer IIS.

To the best of our knowledge, the only directly related work is by Saito et al. [4]. They construct a sampling-frequency-independent (SFI) convolutional analysis and synthesis filter bank (i.e., encoder and decoder) for ConvTasNet [5] by sampling analog gammatone filter impulse responses at the provided sampling frequency. The filter center frequencies are independent of the sampling frequency, and their number is fixed, meaning that the encoded representation covers always the same frequency range. In this contribution, we show that it is possible to achieve sampling frequency independence also with the more commonly-used short-time Fourier transform (STFT), with the advantage that the analytic definition of the transform allows covering the full frequency range. We propose that the key is keeping the spectral and temporal granularity constant and independent of the sampling frequency.

In the remainder of this paper, we look into two DNN architectures built into a common framework (Sec. II), we describe an experimental setup for the sampling frequency change (Sec. III), and show that the parameters of these models can be transferred from a model trained with 8 kHz data to a model for processing 48 kHz data with no significant quality degradation compared to a native 48 kHz model (Sec. IV).

II. Method

A. Sampling frequency independence

The main assumption in this paper is that the time-frequency resolution of the representation in the DNN is constant even when the signal sampling frequency is different. Let us consider the STFT representation as an example. A constant time-frequency resolution across sampling frequencies can be fulfilled by setting a constant frame length and spacing in units of seconds. From this and the sampling frequency, the length of the transform can be determined. As a result, the STFT representations obtained from two signals with different sampling frequencies (but same duration in seconds) will have the same number of frames, e.g., $t_0, \ldots, t_4$, as illustrated in Fig. 1. The number of frequency axis elements (or *bins*) will differ, e.g., f_0, f_1 for the lower sampling frequency, and $f_0, \ldots, f_5$ for a sampling frequency 3 times higher. Still, the spacing of the bins is the same, i.e., f_0 and f_1 will refer to the same frequency sub-bands at both sampling frequencies.

Operations acting only on a local subset of the frequency axis elements (e.g., f_0 and f_1) will now have similar infor-

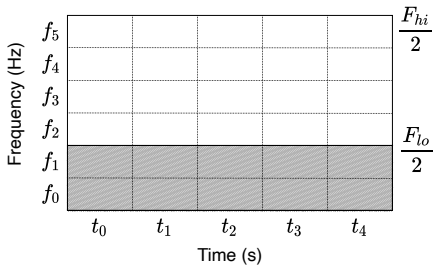

Fig. 1. Time/frequency tiles after the *encoding* layer with an STFT adapted to two different sampling frequencies F_{lo} and $F_{hi} = 3F_{lo}$. The shaded tiles represent the resolution obtained for the lower sampling frequency.

mation available on the overlapping (low) frequency region of all sampling frequencies. Examples of such operations are 1D and 2D convolutions, or 1D recurrent layers operating along the frequency axis or per sub-band, e.g., [6]. Since operations of these layers are independent of the absolute bin frequency, it appears that training with band-limited data will result into parameters that will work also for the frequency region not present in the training.

This is not a generic solution, but dependent on the exact network architecture and layers used, e.g., a model containing a fully-connected layer cannot be trivially converted.

One should note that even though the sampling frequencies used for the main demonstration in this paper have an integer multiple relationship, this is not a strict requirement in the real world. In fact, the resulting frame lengths and transform lengths used in the experiments are not exact integer multiples (see Sec. II-B). Even though the time/frequency-resolutions are not identical and the assumption outlined above is relaxed, no significant perceptual quality degradation is observed (Sec. IV). As a related effect, the resulting time-to-frequency transform lengths may be non-power-of-2.

B. Common DNN framework

The experiments make use of a U-Net core [7]–[10] and a fully-convolutional core, both inside the encoder-decoder architecture shown in Fig. 2 with the common processing steps consisting of:

1) Encoder: The time-domain input signal $x(t)$ is transformed into an encoded representation. The input is assumed to be a mixtures of a foreground target and background non-target components $x(t) = x_{FG}(t) + x_{BG}(t)$. We use STFT implemented as a strided 1D convolution. The number of convolutional filters depends on the sampling frequency for which the model is instantiated. Each filter corresponds to the combination of the windowing function (here, sine window) and the time-reversed impulse response of the STFT kernel. The frame length is ca. 42.7 ms, corresponding to 2048 samples at 48 kHz sampling frequency (342 samples at 8 kHz and 1882 samples at 44.1 kHz), and the stride is 50%.

2) Compression: The magnitudes of the spectral elements $c = \Re(c) + i\Im(c)$ are compressed with $c_z = q\Re(c) + qi\Im(c)$,

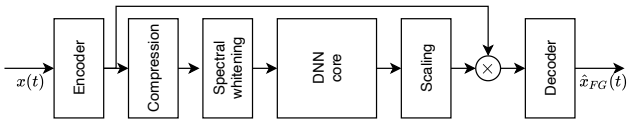

Fig. 2. The common DNN framework used in the experiments. The block *DNN core* is either a U-Net or a convolutional block, depending on the tested system.

where q is the effective magnitude scaling computed as $q = \log(\alpha + |c|)/|c|$, with $\alpha = 1$.

3) The compressed values are split into the real and imaginary parts and stacked from all input audio channels into the *channels* dimension of the network tensors.

4) Spectral whitening: For each frequency bin, subtract the mean and divide by standard deviation over the training data.

5) DNN core: Separation filters (masks) are produced for cross-channel filtering.

6) Scaling: Global scalar offset and scaling are applied to the filters to allow exceeding the range of the last activation.

7) The separation filters are applied on the original encoded representation.

8) Decoder: The encoded result is transformed back to time domain producing an estimate of the target signal $\hat{x}_{FG}(t)$. We use a convolutional STFT synthesis filter bank.

9) An estimate of the non-target component is obtained by a subtraction from the input mixture: $\hat{x}_{BG}(t) = x(t) - \hat{x}_{FG}(t)$.

Applying the proposed sampling frequency conversion for the models implemented in this framework is straightforward. First, the frame length is adjusted according to sampling frequency and the filters of the STFT-based encoder and decoder are designed for the new length. After the encoder, the data is of the same form with only a different number of elements in the dimension corresponding to frequency. The spectral whitening layer needs to be assigned with new per-frequency parameters, which can be computed, e.g., with one pass over target mixtures. As the remaining operations fulfil the assumption of being agnostic to the absolute frequency and apply their function to the whole input regardless of its size, the conversion is now ready.

C. U-Net

The U-Net core follows closely the structure of Spleeter [10] with the following changes. The up-sampling path uses factor 2 nearest-neighbor up-sampling, followed by a 5×5 convolution, instead of the transposed strided convolutions used in the original model. The number of filters in the last up-sampling convolution and the output 5×5 convolution are equal to the number of separation filters. The output activation is tanh.

D. Convolutional network (CNN)

The second DNN core used in the experiments is a fully-convolutional network developed for dialogue separation [2]. Variants of this model have been tested in earlier investigations [3], [11], [12]. The model consists of a stack of 24 convolutional blocks, each having a frequency-domain padding

in reflection mode, a 3×5 (time $\times$ frequency) 2D convolution with 32 filters, a ReLU activation, and a layer normalization for the *channels*. The last block uses tanh activation and the number of filters is equal to the number of separation filters.

III. Model Variants

Our experiments make use of the two core architectures in these variants:

- *48 kHz*: This is the reference condition, in which the model is both trained with and applied on 48 kHz data.
- *8 kHz*: The model is both trained with and applied on 8 kHz data. The test items are down-sampled for processing, and the model outputs are up-sampled back to 48 kHz.
- $F_{train} \rightarrow F_{test}$ *kHz*: The model is trained with data having sampling frequency $F_{train} = 8$ kHz and the parameters are copied to a model for data with sampling frequency of $F_{test} \in \{44.1, 48\}$ kHz. The spectral whitening layer parameters are obtained from the training mixtures at F_{test} kHz. Note that F_{train} and F_{test} do not necessarily have an integer ratio.

A. Data

The data used in our experiments consists of almost 21 hours of stereo audio from real broadcast content. All items have foreground signals (speech or dialogue) and matching background signals (music, effects, non-speech). The native sampling frequency of the data is 48 kHz, a full copy of the data is down-sampled to 8 kHz. The training data (14 h 18 min) and the test set (1 h) are independent and the same as in [11]. The validation data consists of 5 h 36 min of audio from the same item pool as the training data. The training examples are used in full length. Online data augmentation is used for each example with a random offset in the beginning (max. 10 ms) to avoid identical time ranges, 33% chance of being down-mixed into mono (in stereo experiments), a random gain in the range $[-6, +6]$ dB, and a signal-to-background mixing ratios modification by $[-6, +6]$ dB. The order of the training examples and the augmentation are randomized in each epoch.

The main experiments use models for separating stereo components from stereo input signals. For verifying that the models do not rely only on spatial information, the computational evaluation is performed additionally for models separating mono components from mono inputs. For the mono tests, the entire data set is converted to mono and the down-mixing augmentation is disabled.

B. Training

We use time-domain mean absolute error (MAE) as loss and ADADELTA [13] as the optimizer. The training is run until the validation loss does not improve in 10 epochs, and the model with the lowest validation loss is used in the evaluation.

The stereo training of the U-Net converged on epochs 96 and 85 for the 8 kHz and 48 kHz models, and of the CNN on epochs 28 and 30 for the 8 kHz and 48 kHz models (for mono, epochs 61 and 136 for the U-Net and 41 and 43 for the CNN). The number of trainable parameters in the stereo models with the U-Net core is 9,827,330, and with the CNN core 359,438 and does not depend on the sampling frequency. These numbers exclude the parameters of the spectral whitening operation. On our system, the average per-epoch training duration for the U-Net core is 42 s for the 8 kHz version and 6 min 45 s for the 48 kHz version, meaning 9.6 times faster training on the lower sampling frequency data. For the CNN, the average per-epoch training durations are 6 min and 42 min, meaning 7 times faster training with the lower sampling frequency.

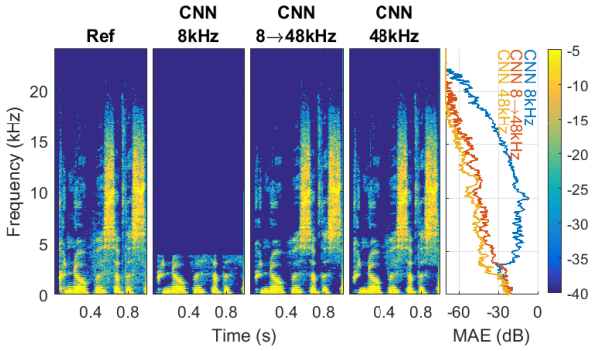

Fig. 3. Examples magnitude spectrograms of a 1 s excerpt. The frequency-dependent MAE is between the reference magnitude spectrum *Ref* and the speech estimates that the CNN variants separated from the input mixture (input SI-SIR = 8.5 dB).

IV. Evaluation

Fig. 3 visualizes example magnitude spectra of the outputs of the three CNN model variants. While obviously *CNN 8 kHz* does not have any content above 4 kHz, both *CNN 8→48 kHz* and *CNN 48 kHz* are able to separate the target with little errors in high band. In particular, *CNN 8→48 kHz* exhibits no major difference with *CNN 48 kHz* in high band separation performance, even if that frequency range was not seen during training. A more formal evaluation follows.

A. Computational evaluation

The computational evaluation uses the scale-invariant signal-to-distortion (SI-SDR), signal-to-artifacts (SI-SAR), and signal-to-interference (SI-SIR) [14], [15] evaluated on the separated foreground signal with the 44.1 kHz or 48 kHz reference signal. Additionally, a measure intended for basic audio quality evaluation by predicting the result of a MUSHRA [16] listening test, the 2f-model [17], [18] is computed. The measures are computed for both the input mixture signal and the model output in order to compute the change in the measure resulting from the models' processing. The results of the stereo experiments are given in Table I and of the mono experiments in Table II.

B. Listening test

A multi-stimulus listening test [16] is conducted to evaluate the perceptual quality. The test items are the same as in [11], taken from the stereo test item pool, and they consist of speech on top of various backgrounds. Each item is 8 s long. The

TABLE I

COMPUTATIONAL EVALUATION OF STEREO MODELS. THE ROWS ΔSI-SDR, ΔSI-SIR, AND Δ2F SHOW THE MEAN PER-ITEM IMPROVEMENT AND THE STANDARD DEVIATION FOR THE GIVEN MEASURE WITH RESPECT TO THE INPUT MIXTURE. THE ROW SI-SAR SHOWS THE MEAN PER-ITEM ABSOLUTE VALUE AND THE STANDARD DEVIATION OF THE PROCESSED CONDITIONS. THE INPUT MIXTURES WERE ARTIFACT-FREE. THE VALUES FOR THE INPUT MIXTURE WERE SI-SDR 6.5±7.0 DB, SI-SIR 6.5±7.0 DB, AND 2F-MODEL 20.6±11.9.

	CNN 8 kHz	CNN 8→48 kHz	CNN 8→44.1 kHz	CNN 48 kHz	U-Net 8 kHz	U-Net 8→48 kHz	U-Net 8→44.1 kHz	U-Net 48 kHz
ΔSI-SDR (dB)	3.0±5.3	**8.3±4.1**	**8.3±4.0**	**8.2±4.3**	2.4±5.0	6.9±4.1	5.9±4.3	7.2±4.5
ΔSI-SIR (dB)	**18.5±8.4**	**18.9±8.8**	**18.7±8.8**	17.7±8.4	15.8±8.0	16.5±8.4	17.2±9.5	16.7±7.4
SI-SAR (dB)	9.9±4.2	**15.6±5.9**	**15.6±5.9**	**15.8±6.1**	9.4±4.5	14.5±6.5	13.2±6.0	14.6±6.3
Δ2f	3.8±8.3	15.7±6.7	15.8±6.7	**16.6±7.0**	3.8±8.2	14.1±7.3	11.2±7.5	13.1±7.6

TABLE II

COMPUTATIONAL EVALUATION OF MONO MODELS. SEE THE CAPTION OF TABLE I FOR THE DESCRIPTION OF THE ROWS. THE VALUES FOR THE INPUT MIXTURE WERE SI-SDR 7.7±7.0 DB, SI-SIR 7.7±7.0 DB, AND 2F-MODEL 20.8±12.0.

	CNN 8 kHz	CNN 8→48 kHz	CNN 48 kHz	U-Net 8 kHz	U-Net 8→48 kHz	U-Net 48 kHz
ΔSI-SDR (dB)	1.6±5.1	**6.6±3.7**	**6.8±3.8**	1.0±4.8	5.3±3.8	5.7±4.2
ΔSI-SIR (dB)	14.1±6.8	14.7±7.2	**15.4±7.2**	12.7±7.5	13.6±7.8	12.7±7.4
SI-SAR (dB)	9.8±4.2	**15.5±6.1**	**15.5±6.0**	9.4±4.5	14.5±6.5	**15.2±6.6**
Δ2f	-1.5±8.2	9.0±5.8	**10.4±5.7**	-1.4±8.6	8.4±6.7	**10.1±6.0**

separated components are mixed to simulate a dialogue enhancement application by attenuating the background estimate 20 dB. The *hidden reference* is obtained from the original component signals used to create the input mixtures, and the *low anchor* is a 4 kHz low-pass filtered version of this. Since this reflects the maximum quality a model operating on 8 kHz signals could reach, the outputs from the *8 kHz* variants are omitted from the test. The test conditions are *8→48 kHz* and *48 kHz* for both U-Net and CNN cores.

The task given to the listeners was: *"... to rate the overall quality of the conditions in comparison with the given reference."* The test was taken by 12 expert listeners in their offices using their own high-quality headphones, and no result was post-screened. Figure 4 shows the test results.

V. DISCUSSION

The performance differences between the variants *48 kHz* and *8→48 kHz* are minimal in all computational measures as well as in the listening test results. Considering the variance across items and the confidence intervals, no significant difference can be observed between the two variants. This observation is valid for both U-Net and CNN cores, and for both mono and stereo models. On the other hand, it is clear especially in the 2f-model output that the *8 kHz* variants perform worse than the models for 48 kHz audio.

The results for the 44.1 kHz sampling frequency target data in Table I show that for the CNN core the conversion *8→44.1 kHz* works equally well as *8→48 kHz*, despite the sampling frequencies not having an integer ratio. The U-Net core architecture converted to 44.1 kHz shows slight differences in the computational evaluation suggesting a slightly better absolute separation performance, but worse artifact-related performance than for the models processing 48 kHz data. A possible reason for this difference may be related to the pooling and un-pooling operations in the U-Net.

Especially from the listening test (Figure 4), it is clear that the relevant model parameters learned from data with low sampling frequency can be transferred to a model for processing data on considerably higher sampling frequency. This property is valid for networks using layers that are frequency-axis agnostic, i.e., apply the same operation regardless of the absolute frequency. Considering ConvTasNet [5], the internal 1D convolutions use the dimension corresponding to frequency as the *channels* dimension, and a different number of frequency bins would result into a different number of filter channels preventing this trivial conversion. Considering OpenUnmix [19], the fully-connected layer across frequencies prevents trivial parameter transfer across sampling frequencies.

In the experiments, the statistics for the spectral whitening layer were learned from one pass through the mixture signals in the training. In practice, a smaller data set could be used for this, or one could use online estimation. We wanted to eliminate this factor from the evaluation and used the true statistics of the training data.

We used STFT filter bank for demonstrating the observation, since it has an analytic form across sampling frequencies and transform lengths. Possibly also other filter banks can be similarly used as long as the spacing in the time/frequency grid remains constant, as discussed in Sec. II-A. Finally, the application of dialogue separation was considered, for which speech is the target signal, which is particularly rich below 4 kHz. The transferability of the learned parameters might work to different extents in other source separation tasks with different target types. This should be investigated in future.

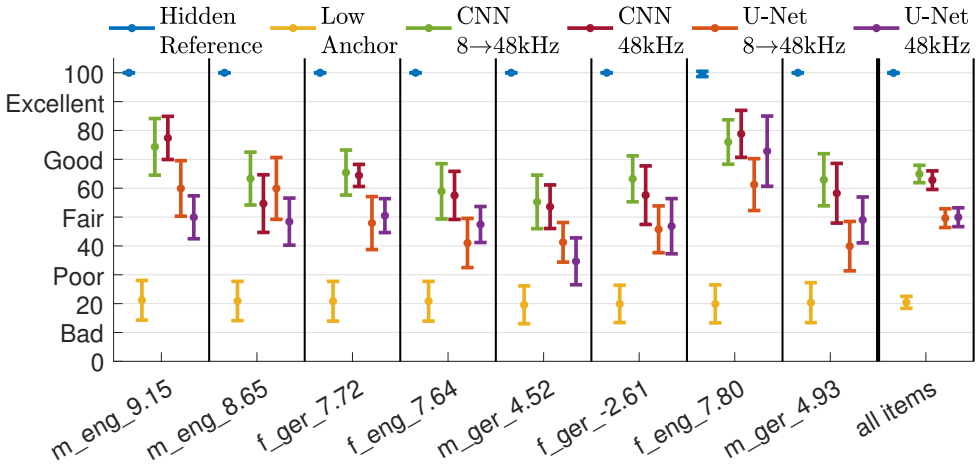

Fig. 4. Listening test results shown with 95% confidence intervals (Student's t-distribution, 12 listeners) per item and test condition, and overall on the right. The items include female (f) and male (m), as well as German (ger) and English (eng) samples, and different input SI-SIR (reported at the end of the each item name). The constant order of the conditions is shown by the legend. Best viewed in colors.

VI. Conclusions

We have observed that in some DNN architectures for audio source separation, the relevant model parameters are independent of the sampling frequency of the audio. This allows, e.g., using training data with multiple sampling frequencies for training a single model, or training a model with one sampling frequency and using it on another. In an experiment, we transferred the parameters from a model trained with 8 kHz sampling frequency data to a model for processing 48 kHz sampling frequency data, and no significant perceptual quality difference was observed with respect to a native 48 kHz model. The training speed was up to 7-10 times faster, depending on the core DNN architecture.

References

[1] K. Subramani and P. Smaragdis, "Point cloud audio processing," in *Proc. of 2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, USA, Oct. 2021.

[2] J. Paulus, M. Torcoli, C. Uhle, J. Herre, S. Disch, and H. Fuchs, "Source separation for enabling dialogue enhancement in object-based broadcast with MPEG-H," *Journal of the Audio Engineering Society*, vol. 67, no. 7/8, pp. 510–521, July/Aug. 2019.

[3] M. Torcoli, C. Simon, J. Paulus, D. Straninger, A. Riedel, V. Koch, S. Wirts, D. Rieger, H. Fuchs, C. Uhle, S. Meltzer, and A. Murtaza, "Dialog+ in broadcasting: First field tests using deep-learning based dialogue enhancement," in *International Broadcasting Convention (IBC) Technical Papers*, 2021.

[4] K. Saito, T. Nakamura, K. Yatabe, Y. Koizumi, and H. Saruwatari, "Sampling-frequency-independent audio source separation using convolution layer based on impulse invariant method," in *Proc. of 29rd European Signal Processing Conference*, Dublin, Ireland, Aug. 2021, pp. 321–325.

[5] Y. Luo and N. Mesgarani, "Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 8, pp. 1256–1266, Aug. 2019.

[6] X. Li and R. Horaud, "Multichannel speech enhancement based on time-frequency masking using subband long short-term memory," in *Proc. of 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, USA, Oct. 2019, pp. 298–302.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., 2015, pp. 234–241.

[8] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, "Singing voice separation with deep U-Net convolutional networks," in *Proc. of 18th International Society for Music Information Retrieval Conference*, Suzhou, China, Oct. 2017.

[9] L. Prétet, R. Hennequin, J. Royo-Letelier, and A. Vaglio, "Singing voice separation: A study on training data," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, Brighton, UK, May 2019, pp. 506–510.

[10] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: a fast and efficient music source separation tool with pre-trained models," *Journal of Open Source Software*, vol. 5, no. 50, p. 2154, June 2020.

[11] M. Strauss, J. Paulus, M. Torcoli, and B. Edler, "A hands-on comparison of DNNs for dialog separation using transfer learning from music source separation," in *Proc. of Interspeech 2021*, Brno, Czechia, Aug. 2021, pp. 3900–3904.

[12] M. Torcoli, J. Paulus, T. Kastner, and C. Uhle, "Controlling the remixing of separated dialogue with a non-intrusive quality estimate," in *Proc. of 2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, USA, Oct. 2021.

[13] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

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

[15] J. L. Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR – Half-baked or well done?" in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, Brighton, UK, May 2019, pp. 626–630.

[16] International Telecommunication Union, "Recommendation ITU–R BS.1534-3 Method for the subjective assessment of intermediate quality level of audio systems," 2015.

[17] T. Kastner and J. Herre, "An efficient model for estimating subjective quality of separated audio source signals," in *Proc. of 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, USA, Oct. 2019, pp. 95–99.

[18] M. Torcoli, T. Kastner, and J. Herre, "Objective measures of perceptual audio quality reviewed: An evaluation of their application domain dependence," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1530–1541, 2021.

[19] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, "Open-Unmix - A reference implementation for music source separation," *Journal of Open Source Software*, vol. 4, no. 41, 2019.