Benchmarking Music Demixing Models for Deep Drum Source Separation

Alessandro Ilic Mezza
DEIB
Politecnico di Milano
Milan, Italy
alessandroilic.mezza@polimi.it

Riccardo Giampiccolo
DEIB
Politecnico di Milano
Milan, Italy
riccardo.giampiccolo@polimi.it

Alberto Bernardini
DEIB
Politecnico di Milano
Milan, Italy
alberto.bernardini@polimi.it

Augusto Sarti
DEIB
Politecnico di Milano
Milan, Italy
augusto.sarti@polimi.it

***Abstract*—Traditionally, drum source separation has been tackled using nonnegative spectro-temporal factorization methods. Only recently, deep learning showed unprecedented performance in separating five stems from a drum mixture, namely, kick drum, snare drum, toms, hi-hat, and cymbals. The literature, however, still lacks a thorough comparison of the techniques readily available in the context of music source separation. In this paper, we conduct a first benchmarking analysis of music demixing models tailored for deep drum source separation. We evaluate a range of state-of-the-art neural network architectures, including HT-Demucs, MDX23C, and BS-RoFormer, trained using StemGMD, a large-scale dataset of isolated single-instrument drum stems. Besides demonstrating that said architectures outperform the state-of-the-art method for drum source separation, we discuss their strengths and weaknesses, giving insights into their performance and ultimately offering valuable guidance for researchers and practitioners willing to develop drum demixing models for different applications, among which those related to music making, personalized listening, and online music education stand out.**

***Index Terms*—deep learning, drum source separation, music demixing.**

I. Introduction

Drum source separation (DSS) is the task of separating audio stems from a drum mixture [1]–[3]. With many potential applications in the context of music production and music technology, there is an ever-increasing demand for tools and methods able to take advantage of DSS. Suffice to think of the great impact that DSS may have on tasks such as re-mastering, re-mixing, sampling, and automatic music and drum transcription [4]. At the same time, DSS holds the promise of being instrumental in many Internet of Sounds (IoS) [5] and Internet of Musical Things (IoMusT) applications [6], such as those concerning personalized user experience, online music learning [7], and remote teaching [8]. Indeed, by complementing Music Demixing (MDX), DSS could provide users the possibility to personalize their listening experience on the fly by remixing an incoming audio stream, or extracting stems to isolate a particular drum pattern and facilitate the learning process. While research on causal low-latency sound demixing is still in the early stages [9], [10], computing platforms have developed to a point where deep learning models can now run in real time on embedded audio devices [11]. Therefore, in the future, MDX and DSS technologies may play a central role in networked music performance applications [12], e.g., by creating a dedicated virtual stage monitoring system for remote musicians interacting over the Internet [13], [14]. It is worth pointing out, in fact, that drum tracks are typically considered as a single stem, and transmitting a single-channel audio stream over a telecommunication network would avoid synchronization issues and greatly reduce the required bandwidth compared to the multi-channel case. Hence, downstream DSS can be exploited at the receiver end to still be able to process drum stems individually. Last but not least, many different web-based services nowadays are offering users the possibility to extract stems (e.g., vocals, bass, etc.) out of songs; while MDX services are well established, those providing DSS capabilities are still not widespread as further research is required to reach satisfying results.

Over the past few years, DSS has been tackled employing traditional signal processing techniques, such as Nonnegative Matrix Factorization (NMF) [1], [15] or Nonnegative Matrix Factor Deconvolution (NMFD) [2], [16], [17], obtaining, in most of the cases, positive results that yet presented inter-channel leaking and artifacts, hardly making them suitable for high-quality music applications. Recently, led by the remarkable performance of deep learning based methods in the context of MDX, deep DSS (DDSS) has started to gather the interest of researchers and practitioners. For instance, in [3], the authors proposed LarsNet, the first published deep learning model for accomplishing DSS. LarsNet is able to separate five stems from a stereo drum mixture, i.e., kick drum, snare drum, toms, hi-hat, and cymbals, obtaining unprecedented results when compared to traditional methods.

Under the hood, LarsNet happens to be similar to Spleeter by Deezer [18], i.e., a U-net based MDX model, posing the question of whether and how other MDX architectures can be applied to solve the task of DSS and what are their performance when trained on a large collection of drum mixtures. Thus, in this paper, we aim at bridging this research gap offering, for the first time, a benchmark of state-of-the-art MDX architectures trained for DSS. In particular, we consider

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 — program "RESTART").

TABLE I
SUMMARY OF THE DEMIXING ARCHITECTURES CONSIDERED IN THE PRESENT STUDY.

Model	# Subbands	Domain	Method	# Params	# Estimators
LarsNet [3]	—	Magnitude STFT	Masking	9,828,680	5
MDX23C (4096)	4	CaC	Synthesis	281,757,952	1
MDX23C (8192)	4	CaC	Synthesis	288,045,568	1
HT-Demucs	—	Hybrid	Synthesis	41,986,766	1
BS-HT-Demucs	4	Hybrid	Synthesis	42,014,474	1
BS-HT-Demucs	8	Hybrid	Synthesis	42,051,418	1
BS-Roformer	62	Complex STFT	Masking	24,591,276	5

HT-Demucs [19], its Band-Split (BS) version, MDX23C [20], and, finally, BS-RoFormer [21], i.e., the winner of the MDX track in the 2023 Sound Demixing (SDX) Challenge [22], [23]. Some architectures are fed magnitude STFTs, others complex-valued STFTs, and still others apply hybrid optimizations considering both STFTs and time-domain waveforms. Moreover, from the output perspective, we distinguish between *synthesis models*, i.e., methods that learn to directly produce the predicted stem waveform, and *masking models*, i.e., methods that learn a soft mask, which, once applied to the mixture STFT, yields an estimate of the desired drum stem. Moreover, we consider both architectures that rely on a single estimator as well as models that rely on N estimators, with N being the number of drum stems. Here, the latter category turns out to coincide with masking methods, since the corresponding MDX models were not designed to stack multiple stems along the channel dimension of their output tensors and, thus, yield them all at once.

The DDSS models considered in the present study are trained and evaluated on StemGMD [24], a corpus of isolated drum stems presented in [25], which, to the best of our knowledge, constitutes the only large-scale dataset available in the literature for accomplishing DSS. Our analysis reveals that MDX architectures, when applied to DDS, do not maintain the relative performance they exhibit when applied to MDX. Specifically, models operating in both the time and time-frequency domains are, in general, characterized by a higher Signal-to-Distortion Ratio (SDR) and appear to have better generalization properties.

Ultimately, this work aims to provide researchers and practitioners with valuable guidance for approaching drum source separation, with new insights into the pros and cons of state-of-the-art MDX architectures, paving the way toward a more thorough and aware design of DDSS methods.

II. LARSNET

Proposed in [3], LarsNet is the first published deep learning based method specifically tailored for DSS. LarsNet comprises five U-Nets, each trained to separate one of five audio stems from a stereo drum mixture, i.e., kick drum (KD), snare drum (SD), toms (TT), hi-hat (HH), and cymbals (CY). In particular, the TT stem includes tom-toms and floor toms; the HH stem comprises both open and closed hi-hat hits; and the CY stem encompasses all crash and ride cymbals.

Akin to Spleeter by Deezer [18], the U-Nets are fully-convolutional and operate in the time-frequency domain. Each network is fed a $F \times T$ patch of the magnitude STFT of the input stereo mixture, and outputs a stem-specific soft mask through a Sigmoid nonlinearity. The time-domain stem signal is then estimated by taking the inverse STFT (ISTFT) of the element-wise product between the mixture STFT and the soft mask thus obtained. We compute the STFT using a 4096-sample Hann window with 75% overlap, and set $F = 2048$, thus zeroing out only the Nyquist coefficient. Additionally, at inference time, we segment and batch input mixtures longer than $T = 512$.

The neural networks are trained using StemGMD, a recently released dataset of isolated single-instrument drum stems [24], [25]. The training involves the minimization of a L^1-loss function between the magnitude STFT of the ith ground truth stem $\mathbf{X}_i \in \mathbb{R}_{\geq 0}^{2\times F\times T}$ and that of the masked mixture, i.e.,

$$\mathcal{L}_{\text{LarsNet}} = \|\mathbf{X}_i - \hat{\mathbf{M}}_i \odot \mathbf{X}\|_1, \tag{1}$$

where $\odot$ denotes the Hadamard product, $\hat{\mathbf{M}}_i$ is the ith soft mask, and $\mathbf{X} \in \mathbb{R}_{\geq 0}^{2\times F\times T}$ is the magnitude STFT of the mixture, such that $\hat{\mathbf{X}}_i = \hat{\mathbf{M}}_i \odot \mathbf{X}$ is the magnitude STFT of the predicted stem.

III. MUSIC DEMIXING MODELS

In this section, we provide a summary of the state-of-the-art methods in the field of music source separation that will be tested further ahead in the context of drum source separation. The key features of each method are reported in Table I.

A. MDX23C

Proposed for the 2023 SDX challenge, MDX track, leaderboard C, where it achieved 3rd place [23], MDX23C is a modified version of the more known KUIELab-MDX-Net originally proposed for the 2021 MDX challenge [26]. Both MDX23C and KUIELab's models are, in fact, multi-source TDF-TDC-U-Nets [27]. Introduced in 2020, the original TDF-TDC-U-Net model was first used for singing voice separation [28].

As shown in Fig. 1b, it consists of a U-Net where all convolutional layers are replaced with TFC-TDF blocks. In such blocks, the output of a first time-frequency convolution (TFC) block (normalization, pre-activation, 2D convolutions) is fed to a stack of time-distributed fully-connected (TDF) layers processing each frame of the latent representation along

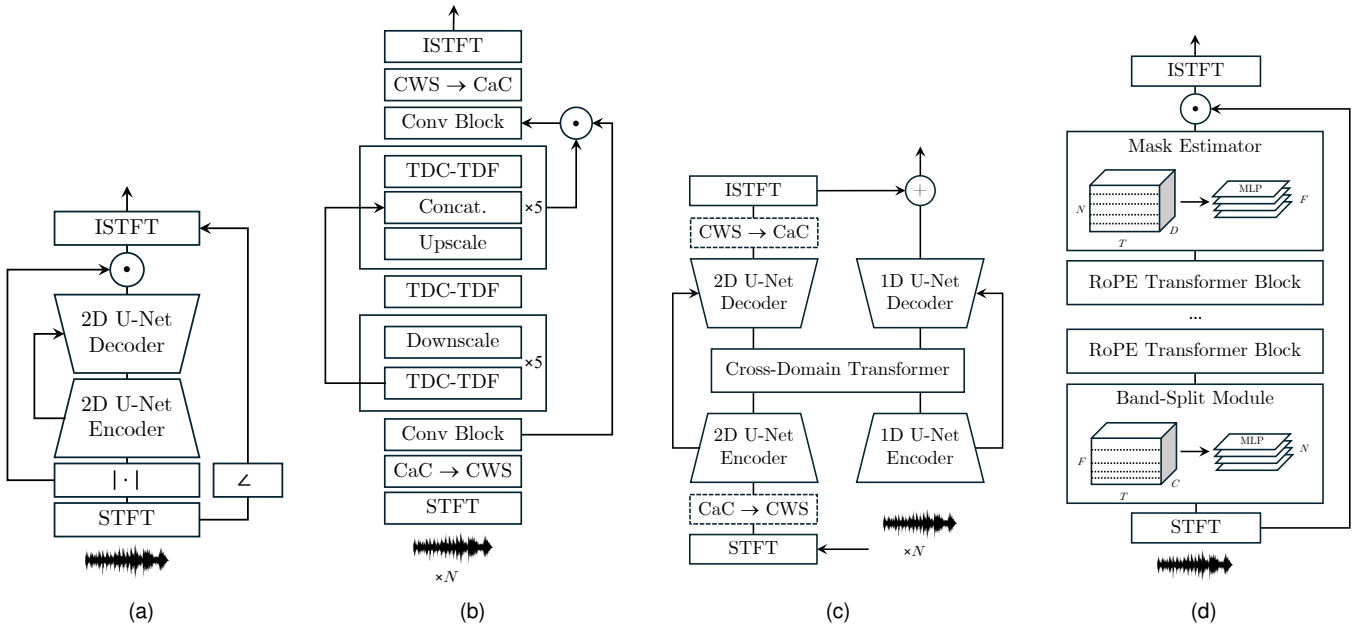

Fig. 1. Demixing architectures considered in our benchmark. (a) LarsNet; (b) MDX32C; (c) Band-Split HT-Demucs, which, without the dashed blocks, corresponds to HT-Demucs; (d) BS-RoFormer.

the axis corresponding to time with the same weights, before being processed again by a second TFC block.

The former operations are performed on real-valued tensors, where real and imaginary parts of the complex STFT are treated as channels, a procedure known as "Complex as Channels," or CaC for short [28]. Furthermore, KUIELab-MDX-Net includes subband processing as in [29], which proved beneficial in many music demixing works [21], [30]. In particular, MDX23C implements a CaC to Channel-Wise Subband (CWS) transformation module, where a tensor with C channels, T frames, and F bins is evenly partitioned into K subbands of size F/K, and rearranged such that the subbands are stacked along the channel dimension which is now of size CK. This way, CWS allows the model to learn different weights for each subband.

MDX23C features five encoder blocks and five decoder blocks. Each block either halves or doubles the number of frequency bins and lets the number of channels grow by 128. As typical in classic U-Nets, shortcut paths connect the feature maps at corresponding encoder-decoder levels via concatenation. Finally, an outer skip connection feeds the output of the convolutional frontend into the output of the last convolutional block through element-wise multiplication.

In this work, we consider two MDX23C variants with different time-frequency resolutions of its STFT frontend. Namely, we set $N_{\text{FFT}} = 4096$ and $N_{\text{FFT}} = 8192$ as the size of the frame-by-frame Fast Fourier Transforms (FFTs). We train both models employing a time-domain L^1-loss and a multi-resolution STFT loss using `auraloss` [31] such that

$$\mathcal{L}_{\text{MDX23C}} = \|x_i - \hat{x}_i\|_1 + \sum_{s=1}^{S} \|\mathbf{X}_i^{(s)} - \hat{\mathbf{X}}_i^{(s)}\|_1 , \qquad (2)$$

where x_i and $\hat{x}_i$ are the ground-truth and predicted stem waveforms, and $\mathbf{X}_i$ and $\hat{\mathbf{X}}_i$ are the ground-truth and predicted magnitude STFTs, respectively, with $s = 1, \ldots, S$ being the multi-resolution index.

B. HT-Demucs

Hybrid Transformer Demucs (HT-Demucs) [19] is built upon Demucs' and Hybrid Demucs' (H-Demucs) architectures, originally proposed in [32] and [33], respectively. Fig. 1c shows HT-Demucs, which, however, must be considered without the dashed blocks; these will be discussed further ahead. In particular, the original H-Demucs architecture comprises two U-Nets: one operating in the time domain and the other operating in the time-frequency domain. Each U-Net consists of five encoder layers and five decoder layers with the two innermost layers being shared between the two signal paths. The output from the spectral branch is transformed back via ISTFT before being combined with the output from the temporal branch, thereby providing the model final prediction.

Building upon the previous model, HT-Demucs preserves the outermost four layers from the original architecture while substituting the innermost layers in the encoder and decoder with a cross-domain transformer. This new architecture processes in parallel the 2D signal from the spectral branch and the 1D signal from the waveform branch. In contrast to the original H-Demucs, which relies on thorough parameter tuning to align temporal and spectral representations, the cross-domain transformer allows greater flexibility. HT-Demucs is trained using a time-domain L^1-loss, i.e.,

$$\mathcal{L}_{\text{HT-Demucs}} = \|x_i - \hat{x}_i\|_1 , \qquad (3)$$

TABLE II
SIGNAL-TO-DISTORTION RATIO FOR EACH DRUM KIT.

	Seen Drum Kits						Unseen Drum Kits			
	Brooklyn	**East Bay**	**Heavy**	**Portland**	**Retro Rock**	**SoCal**	**Bluebird**	**Detroit G.**	**Motown R.**	**Roots**
LarsNet	17.71	18.24	16.93	18.62	18.54	18.94	18.63	17.1	15.53	16.79
MDX23C ($N_{\mathrm{FFT}} = 4096$)	19.98	19.45	19.44	20.57	20.34	20.31	19.87	17.62	13.9	16.2
MDX23C ($N_{\mathrm{FFT}} = 8192$)	**20.98**	20.48	**20.55**	21.01	**21.06**	**21.02**	20.15	18.15	15.67	17.81
HT-Demucs	20.68	**20.59**	20.21	21.22	20.84	20.94	**20.71**	**19.23**	**17.64**	**19.89**
BS-HT-Demucs (4 Bands)	19.92	19.36	19.45	**21.26**	19.93	19.69	20.16	18.58	17.0	18.77
BS-HT-Demucs (8 Bands)	20.1	19.65	19.62	21.15	20.2	20.06	20.26	18.67	17.29	19.1
BS-Roformer	18.76	19.42	17.96	19.63	19.31	20.06	18.67	14.91	14.72	15.68

where x_i and $\hat{x}_i$ are the ith ground-truth and predicted stem waveforms, respectively.

Finally, when writing [19], the authors envisioned to couple HT-Demucs with subband processing as in [29]. In this work, we implement CWS as discussed in Sec. III-A, and we name the resulting model *Band-Split HT-Demucs*. This architecture is depicted in Fig. 1c, where now, contrary to the original HT-Demucs, the dashed blocks must be counted among the layers. In this work, we consider the classic full-band HT-Demucs architecture, as well as Band-Split (BS) variants with four and eight subbands, respectively.

C. Band-Split RoFormer

Motivated by the success of Band-Split RNN [30], the authors of [21] introduced Band-Split RoFormer, whose architecture is shown in Fig. 1d. Most notably, the model achieved the first place in the MDX track of the 2023 SDX Challenge [23]. Just like its RNN-based counterpart, BS-RoFormer uses a band-split frontend to decompose the input STFT representation into nonoverlapping subbands. In particular. The input STFT is first segmented into frequency bands of uneven size. Then, the resulting time-frequency patches are fed to a stack of dedicated Multi-Layered Perceptrons (MLPs) each consisting of RMSnorm [34] and a fully-connected linear layer that equalizes the number of latent features per band. The ensuing representation is fed to a stack of alternating *time-transformers* and *subband-transformers*. The former apply self-attention across the time dimension, whereas the latter attend along the subband axis. Furthermore, Rotary Positional Embeddings (RoPE) [35] are applied by rotating queries and keys depending on their position in the corresponding sequence. Our implementation uses six interleaved pairs of subband- and time-transformers with RoPE embeddings instead of twelve, resulting in a total of 25 M trainable parameters, almost four times less than the original MDX model [21]. The output of the last transformer block is thus passed to an MLP stack tasked with estimating a complex-valued mask that, when applied to the source STFT via element-wise multiplication, allows for reconstructing the target stem via ISTFT.

Similar to LarsNet, but differently from the other MDX models described above, the method consists of a collection of N BS-RoFormers, one for each stem, that were trained using the same loss function as of MDX23C, i.e., (2). Nevertheless, it is worth noticing that, in this case, $\hat{\mathbf{X}}_i^{(s)}$ is the magnitude STFT at resolution s of $\hat{x}_i = \mathrm{ISTFT}\big\{\hat{\mathbf{M}}_i \odot \mathrm{STFT}\{x\}\big\}$, while $\hat{\mathbf{M}}_i$ is the complex-valued soft mask predicted for the ith stem.

IV. FROM MDX ARCHITECTURES TO DDSS MODELS

The architectures presented in the previous sections, with the exception of LarsNet, were originally proposed for MDX and not DSS. We trained all of them using StemGMD.

StemGMD [24], [25] contains audio tracks of nine percussion instruments from a canonical acoustic drum kit, namely, kick drum, snare drum, high tom, low-mid tom, floor tom, open hi-hat, closed hi-hat, crash cymbal, and ride cymbal. The tracks are synthesized from Groove MIDI Dataset's MIDI recordings [36] as a 44.1 kHz stereo wav files using ten different Logic Pro X drum kits. In this work, following [3], we reduce the number of classes to five by aggregating all toms, all cymbals, and all hi-hats in three broader classes. In doing so, we have also adopted the conventions introduced by [3] that prescribed to use six out of ten drum kits for training and testing (*seen* drum kits) and hold out four drum kits exclusively for evaluation purposes (*unseen* drum kits). This leaves us with an *eval session* comprising 400 mixture files (40 for each drum kit).

Differently from LarsNet, which used aligned stems, i.e., every training pair consisted of a mixture and a corresponding stem coming from the same drumming performance, the new DDSS models are trained by sampling random chunks of isolated drums and creating the mixture simply by summing them. To do so, we discard every silence track from StemGMD. This way, when mixing random chunks, we ensure that all percussion instruments in the drum kit are represented, at least with one hit per training sample. Discarding silence leaves us with 4338 kick tracks, 5196 snare tracks, 4500 toms tracks, 3900 hi-hat tracks, and 2526 cymbals tracks.

Additionally, we perform data augmentation on every isolated clip using Spotify's Pedalboard [37] and the `audiomentations` Python library. These augmentation methods outnumber those originally proposed for LarsNet: `audiomentations` implements hyperbolic tangent distortion and MP3 compression augmentation, as well as additive white Gaussian noise corruption for hi-hat and cymbals. In turn, Pedalboard allows us to apply random compression, chorus, phaser, reverberation, bitcrash, and resampling in the range of 16 kHz to 44.1 kHz. Moreover, we implement

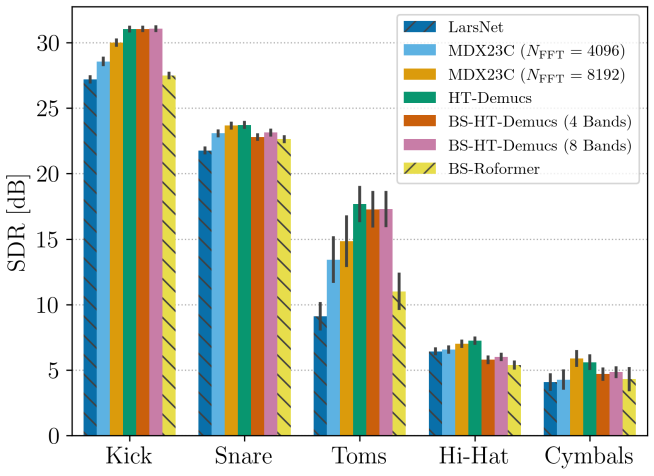

Fig. 2. Average Signal-to-Distortion Ratio (SDR) and standard error (SE) of the DDSS models considered in the present study. Hatched bars indicate methods that rely on multiple estimators.

channel swap[1] and polarity inversion with a probability of 50%, random gain variation in the range of 0.5 to 1.5, and random panning according to a constant-power pan-rule with a probability of 10%. Every target stem could be loaded as the mixture of several randomly selected tracks of the same instrument, possibly from different drum kits. This procedure is limited to three files at a time, where the second is sampled with a probability 20%, and the third with a probability of 10%. At each sampling step, the previous and new waveforms are averaged with uniformly distributed random weights adding up to one. The remaining augmentation methods are then applied to the resulting mixture.

We train each model on two Tesla V100 for about 50,000 iterations with variable gradient accumulation.[2]

V. Results and Discussion

Regarding the evaluation, we compute the SDR as it is particularly sensitive to inter-channel leakage artifacts, and is the metric of choice for past MDX challenges [23], [26]. For the ith stem, the SDR is given by

$$\text{SDR}_i = 10 \log_{10} \frac{\|x_i\|^2 + \epsilon}{\|x_i - \hat{x}_i\|^2 + \epsilon}, \tag{4}$$

where $\epsilon = 10^{-7}$ avoids numerical problems.

All the methods outperform LarsNet, as it can be appreciated by looking at Fig. 2, which shows the results of the proposed benchmark. Bars represent, for each stem, the average SDR computed over every drum kit in the eval session. Moreover, hatched bars refer to methods using N neural networks in parallel as opposed to plain bars that represent single-estimator models. As a first remark, it is interesting indeed to note that the latter outperforms the former, i.e., the methods in which every model has been trained specifically to extract one particular stem. For instance, HT-Demucs, on average, performs better than the other methods, suggesting that waveform synthesis, along with processing the signal in both the time and time-frequency domains, is a promising way to approach the task. In fact, we may think of drum hits as having a broad spectrum due to their impulsive nature, which makes the temporal resolution of onsets critical for achieving satisfactory results. It might be indeed the wide frequency response characterizing drums the reason why BS-HT-Demucs shows poorer performance compared to HT-Demucs, and, in general, band-split methods do not provide any evident benefit when it comes to DDSS.

Table II breaks down the results into seen and unseen drum kits. Notably, MDX23C with $N_{\text{FFT}} = 8192$ turns out to be the best model on many seen drum kits (Brooklyn, Heavy, Retro Rock, SoCal). HT-Demucs, while otherwise close-second, outperforms MDX23C when it comes to processing unseen kits. We may argue that, possibly due to the large number of trainable parameters and the limited timbre variety of StemGMD, MDX23C is prone to overfit and fails to generalize as well as the smaller HT-Demucs model.

We deem the overall results very interesting, especially because BS-RoFormer, although being the winner of the 2023 SDX Challenge, is not able to outperform the other models. Many can be the causes contributing to such a downgraded performance; we argue that this may be due to a combination of reduced model capacity, and, once again, limited timbral diversity characterizing StemGDM. These claims, however, must be further investigated in future works.

VI. Conclusions

We presented a first benchmark of state-of-the-art MDX architectures trained for accomplishing deep drum source separation. We took into account the best performing methods currently available in the literature, including top-ranking and winners of past SDX Challenges, such as BS-RoFormer and MDX23C. We provided a comparison of methods operating in the time-frequency domain and in both the time and time-frequency domains, methods trained to estimate soft masks and to directly synthesize stem waveforms, and again, methods relying on several estimators and employing a single neural network. Our analysis shows that all the state-of-the-art MDX architectures involved in the benchmark are able to outperform LarsNet, i.e., the state-of-the-art method for DDSS, and that hybrid models tend to yield better results and have better generalization capabilities. Going forward, we envision DDSS as a key tool for augmenting and enriching the music making process, while offering users a growing number of ways to personalize their listening experience.

[1]Notice that, in StemGMD, instruments are panned to the left or right depending on their typical position in a canonical drum kit [25].

[2]Audio examples and DDSS models are available online at https://polimi-ispl.github.io/benchmark-ddss.

References

[1] C. Dittmar and D. Gärtner, "Real-time transcription and separation of drum recordings based on NMF decomposition," in *International Conference on Digital Audio Effects (DAFx-14)*, pp. 187–194, 2014.

[2] L. Vande Veire, C. De Boom, and T. De Bie, "Sigmoidal NMFD: convolutional NMF with saturating activations for drum mixture decomposition," *Electronics*, vol. 10, no. 3, p. 284, 2021.

[3] A. I. Mezza, R. Giampiccolo, A. Bernardini, and A. Sarti, "Toward deep drum source separation," *Pattern Recognition Letters*, vol. 183, pp. 86–91, 2024.

[4] C.-W. Wu, C. Dittmar, C. Southall, R. Vogl, G. Widmer, J. Hockman, M. Müller, and A. Lerch, "A review of automatic drum transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 9, pp. 1457–1483, 2018.

[5] L. Turchet, M. Lagrange, C. Rottondi, G. Fazekas, N. Peters, J. Østergaard, F. Font, T. Bäckström, and C. Fischione, "The internet of sounds: Convergent trends, insights, and future directions," *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11264–11292, 2023.

[6] L. Turchet, C. Fischione, G. Essl, D. Keller, and M. Barthet, "Internet of musical things: Vision and challenges," *IEEE Access*, vol. 6, pp. 61994–62017, 2018.

[7] S. A. Ruthmann and D. G. Hebert, "Music learning and new media in virtual and online environments," *Creativities, technologies and media in music learning and teaching: an Oxford handbook of music education*, pp. 254–272, 2018.

[8] C. Alexandraki, N. Mimidis, Y. Viglis, A. Nousias, D. Milios, and K. Tsioutas, "Collaborative playalong practices in online music lessons: the MusiCoLab toolset," in *2023 4th International Symposium on the Internet of Sounds*, pp. 1–10, 2023.

[9] G. R. Dabike, M. A. Akeroyd, S. Bannister, J. Barker, T. J. Cox, B. Fazenda, J. Firth, S. Graetzer, A. Greasley, R. Vos, and W. Whitmer, "The need for causal, low-latency sound demixing and remixing to improve accessibility," in *SDX Workshop 2023*, Nov. 2023.

[10] S. Venkatesh, A. Benilov, P. Coleman, and F. Roskam, "Real-time low-latency music source separation using hybrid spectrogram-tasnet," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 611–615, 2024.

[11] D. Stefani and L. Turchet, "Real-time embedded deep learning on Elk Audio OS," in *2023 4th International Symposium on the Internet of Sounds*, pp. 1–10, 2023.

[12] C. Rottondi, C. Chafe, C. Allocchio, and A. Sarti, "An overview on networked music performance technologies," *IEEE Access*, vol. 4, pp. 8823–8843, 2016.

[13] J.-P. Cáceres and C. Chafe, "Jacktrip: Under the hood of an engine for network audio," *Journal of New Music Research*, vol. 39, no. 3, pp. 183–187, 2010.

[14] C. Drioli, C. Allocchio, and N. Buso, "Networked performances and natural interaction via LOLA: Low latency high quality A/V streaming system," in *Information Technologies for Performing Arts, Media Access, and Entertainment* (P. Nesi and R. Santucci, eds.), (Berlin, Heidelberg), pp. 240–250, Springer Berlin Heidelberg, 2013.

[15] C.-Y. Cai, Y.-H. Su, and L. Su, "Dual-channel drum separation for low-cost drum recording using non-negative matrix factorization," in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 17–22, 2021.

[16] P. Smaragdis, "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs," in *Independent Component Analysis and Blind Signal Separation: Fifth International Conference (ICA)*, pp. 494–499, 2004.

[17] C. Dittmar and M. Müller, "Reverse engineering the Amen break — score-informed separation and restoration applied to drum recordings," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 9, pp. 1535–1547, 2016.

[18] 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, 2020.

[19] S. Rouard, F. Massa, and A. Défossez, "Hybrid transformers for music source separation," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1–5, 2023.

[20] R. Solovyev, A. Stempkovskiy, and T. Habruseva, "Benchmarks and leaderboards for sound demixing tasks," *arXiv preprint arXiv:2305.07489v1*, 2023.

[21] W.-T. Lu, J.-C. Wang, Q. Kong, and Y.-N. Hung, "Music source separation with band-split rope transformer," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 481–485, 2024.

[22] J.-C. Wang, W.-T. Lu, Q. Kong, and Y.-N. Hung, "BS-RoFormer: The SAMI-ByteDance music source separation system for Sound Demixing Challenge 2023," in *SDX Workshop 2023*, Nov. 2023.

[23] G. Fabbro, S. Uhlich, C.-H. Lai, W. Choi, M. Martínez-Ramírez, W. Liao, I. Gadelha, G. Ramos, E. Hsu, H. Rodrigues, *et al.*, "The Sound Demixing Challenge 2023 – Music Demixing track," *Transactions of the International Society for Music Information Retrieval*, 2024.

[24] A. I. Mezza, R. Giampiccolo, A. Bernardini, and A. Sarti, "StemGMD: A large-scale audio dataset of isolated drum stems for deep drums demixing," Apr. 2023. Zenodo, 10.5281/zenodo.7860223.

[25] A. I. Mezza, R. Giampiccolo, A. Bernardini, and A. Sarti, "StemGMD: A large-scale multi-kit audio dataset for deep drums demixing," in *SDX Workshop 2023*, Nov. 2023.

[26] Y. Mitsufuji, G. Fabbro, S. Uhlich, F.-R. Stöter, A. Défossez, M. Kim, W. Choi, C.-Y. Yu, and K.-W. Cheuk, "Music Demixing Challenge 2021," *Frontiers in Signal Processing*, vol. 1, pp. 1–14, 2022.

[27] M. Kim, W. Choi, J. Chung, D. Lee, and S. Jung, "KUIELab-MDX-Net: A two-stream neural network for music demixing," *arXiv preprint arXiv:2111.12203*, 2021.

[28] W. Choi, M. Kim, J. Chung, D. Lee, and S. Jung, "Investigating U-Nets with various intermediate blocks for spectrogram-based singing voice separation," in *Proc. of the 21th International Society for Music Information Retrieval Conference*, 2020.

[29] H. Liu, L. Xie, J. Wu, and G. Yang, "Channel-wise subband input for better voice and accompaniment separation on high resolution music," in *Interspeech 2020*, pp. 1241–1245, 2020.

[30] Y. Luo and J. Yu, "Music Source Separation With Band-Split RNN," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 1893–1901, 2023.

[31] C. J. Steinmetz, "auraloss: Audio-focused loss functions in pytorch," in *Digital Music Research Network Workshop*, Dec. 2020.

[32] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Music source separation in the waveform domain," *arXiv preprint arXiv:2111.03600*, 2019.

[33] A. Défossez, "Hybrid spectrogram and waveform source separation," *arXiv preprint arXiv:1911.13254*, 2021.

[34] B. Zhang and R. Sennrich, "Root mean square layer normalization," in *Proc. of the 33rd Conference on Neural Information Processing Systems*, 2019.

[35] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, p. 127063, 2024.

[36] J. Gillick, A. Roberts, J. Engel, D. Eck, and D. Bamman, "Learning to groove with inverse sequence transformations," in *International Conference on Machine Learning (ICML)*, 2019.

[37] P. Sobot, "Pedalboard," July 2021. Zenodo, 10.5281/zenodo.7817838.