

# DOSE: Drum One-Shot Extraction from Music Mixture

Suntae Hwang<sup>1</sup>Seonghyeon Kang<sup>1</sup>Kyungsu Kim<sup>1</sup>Semin Ahn<sup>2</sup>Kyogu Lee<sup>1,3</sup><sup>1</sup> Music and Audio Research Group (MARG), Department of Intelligence and Information, Seoul National University<sup>2</sup> Department of Mechanical Engineering, Seoul National University<sup>3</sup> Interdisciplinary Program in Artificial Intelligence & Artificial Intelligence Institute, Seoul National University  
{iamsuntae1, shkang17, kyungsu.kim, susemi2399, kglee}@snu.ac.kr

**Abstract**—Drum one-shot samples are crucial for music production, particularly in sound design and electronic music. This paper introduces Drum One-Shot Extraction, a task in which the goal is to extract drum one-shots that are present in the music mixture. To facilitate this, we propose the Random Mixture One-shot Dataset (RMOD), comprising large-scale, randomly arranged music mixtures paired with corresponding drum one-shot samples. Our proposed model, Drum One-Shot Extractor (DOSE), leverages neural audio codec language models for end-to-end extraction, bypassing traditional source separation steps. Additionally, we introduce a novel onset loss, designed to encourage accurate prediction of the initial transient of drum one-shots, which is essential for capturing timbral characteristics. We compare this approach against a source separation-based extraction method as a baseline. The results, evaluated using Fréchet Audio Distance (FAD) and Multi-Scale Spectral loss (MSS), demonstrate that DOSE, enhanced with onset loss, outperforms the baseline, providing more accurate and higher-quality drum one-shots from music mixtures. The code, model checkpoint, and audio examples are available at <https://github.com/HSUNEH/DOSE>

**Index Terms**—Drum, One-Shot, Music Source Separation, Neural Audio-Codec, Generative Model

## I. INTRODUCTION

Drum sounds are fundamental components of contemporary music production, playing a crucial role across various genres such as electronic music, hip-hop, and pop. In these genres, music producers often construct rhythmic patterns by sequencing individual drum one-shot samples, allowing for precise control over the timbral and temporal characteristics of the rhythm. Given the significance of drum sounds in music production, there has been substantial research interest in synthesizing drum one-shot samples using advanced technologies, including recent developments in deep learning [1]–[5]. These novel approaches utilize conditional inputs such as drum type or acoustic features, aiming to facilitate intuitive sample creation for music producers without requiring extensive knowledge of signal processing techniques.

In many practical scenarios, music producers work with existing recordings to create remixes, cover versions, or other productions, necessitating the extraction of high-quality drum samples directly from music mixtures. We define this task as *Drum One-Shot Extraction*. A conventional approach to this task involves applying music source separation techniques to isolate the drum track, followed by

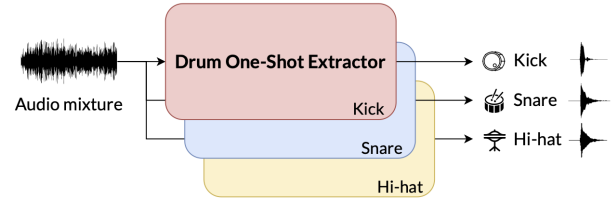


Fig. 1. Illustration of our approach. Given an audio mixture as input, each *Drum One-Shot Extractor (DOSE)* model extract one-shot audio samples for kick, snare, and hi-hat drums.

the identification and extraction of segments containing isolated one-shot samples. We refer to this type of method as the “separation-based approach”. However, this approach cannot guarantee the identification of isolated one-shot segments and is dependent on separation algorithms, potentially introducing artifacts and compromising sample quality.

To address these limitations, we propose a novel *generation-based one-shot extraction* approach that circumvents the intermediate separation step and directly generates drum one-shot samples from the input music mixture. Our method, **DOSE**, leverages recent advancements in neural audio codec language modeling to perform end-to-end generation of drum one-shots. DOSE employs separate decoder-only Transformers for each drum type (kick, snare, and hi-hat) to achieve better extraction performance compared to using a single model for all drum types.

The architecture of DOSE closely follows that of MusicGen [6], utilizing the same core components of neural audio codec and decoder-only transformer. The DAC [7] encoder encodes both the music mixture and the one-shot audio into discrete acoustic tokens. The transformer is then trained to autoregressively generate the acoustic tokens of the one-shot samples conditioned on the acoustic tokens of the music mixture. The generated tokens are decoded into waveform audio via the DAC decoder.

For training and evaluation of DOSE, we introduce the Random Mixture One-shot Dataset (RMOD), a novel paired dataset comprising synthetically generated music mixtures and their corresponding drum one-shot samples. RMOD consists of 360,000 pairs of mixture audio and corresponding drum one-shot samples, created by randomly mixing drum tracks, which are synthesized using one-shot samples, with instrument tracks.

We conducted a comprehensive quantitative evaluation of DOSE and baseline methods on the one-shot drum sample extraction task. In addition to RMOD, we utilized the Groove MIDI Dataset [8] in our evaluation, which offers more realistic drum performances. Our experimental results demonstrate that DOSE outperforms the baseline method, which is a separation-based approach implemented

This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) [No. RS-2023-00219429, 50%], Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [No. RS-2022-II220320, 2022-0-00320, Artificial intelligence research about cross-modal dialogue modeling for one-on-one multi-modal interactions, 40%], [No. RS-2021-II212068, Artificial Intelligence Innovation Hub (Artificial Intelligence Institute, Seoul National University), 5%], and [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University), 5%]

using LarsNet[citation], across various objective metrics, including Fr chet Audio Distance (FAD) [9] and Multi-Scale Spectral Similarity (MSS)) [10].

## II. RELATED WORK

**Drum One-Shot Generation.** Generating drum one-shot samples has received considerable attention with the development of neural audio synthesis methods. Early approaches leveraged models such as Variational Autoencoders (VAE) [11] and Generative Adversarial Networks (GAN) [12], which introduced latent-space exploration and controllable synthesis for drum sounds. Subsequent works like NeuroDrum [2], DrumGAN [4], and StyleWaveGAN [5] further improved controllability and timbral diversity by conditioning on various audio features (e.g., brightness, boominess, depth).

Score-based diffusion models [13] have recently emerged as a powerful paradigm for audio synthesis, offering flexible sampling strategies. CRASH [14], for instance, leverages stochastic differential equations (SDEs) to generate high-resolution percussive sounds (44.1 kHz) in a controllable manner, matching the fidelity of GAN-based methods while enabling techniques such as class mixing to create “hybrid” sounds. These advancements illustrate the growing breadth of approaches for drum/percussive sound generation, providing more interactive and fine-grained creative possibilities for music producers.

**Drum Source Separation.** In traditional music source separation, the goal is to split a mixture into individual instrument stems, including drums, bass, vocals, and others [15]–[18]. More specific to drum-oriented tasks, Mezza et al. [19] introduced a new challenge called Drum Source Separation, aiming to decompose a drum track into separate stems (kick, snare, hi-hat, etc). Their proposed method, LarsNet, accomplishes drum-component separation, allowing for individual extraction of each drum type.

**Neural Audio Codec Language Modeling.** Neural audio codecs such as SoundStream [20], EnCodec [21], and DAC [7] have received notable attention for their ability to compress audio into discrete tokens with minimal perceptual loss. When combined with transformer models, these tokenizers can be leveraged for autoregressive audio generation tasks. MusicGen [6] and MusicLM [22] are prime examples, demonstrating high-quality music generation via codec-based language models.

## III. METHOD

We propose DOSE, a deep learning model designed to extract drum one-shot sounds (kick, snare, and hi-hat) from complex audio mixtures. Inspired by MusicGen [6], DOSE employs a decoder-only transformer to process discrete audio representations. To emphasize accurate transient predictions, we introduce a novel onset loss during training.

### A. Autoregressive Acoustic Token Generation

DOSE generates drum one-shot sounds by autoregressively predicting acoustic tokens conditioned on mixture-audio tokens. This involves converting audio waveforms into discrete tokens using Descript Audio Codec (DAC) [7], predicting drum one-shot tokens, and decoding them into waveforms.

DAC processes mono audio input  $x \in \mathbb{R}^{T_{in} \cdot f_s}$ , tokenizing it into a discrete code

$$q \in \{1, \dots, N\}^{(T_{in} \cdot f_c) \times K},$$

where  $K$  is the number of codebooks,  $N$  is the codebook size, and  $f_c \ll f_s$  is the codec frame rate.

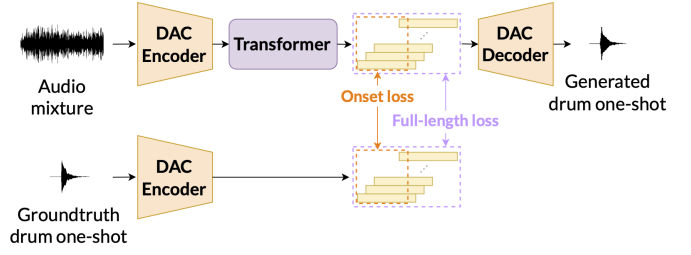


Fig. 2. Proposed Method. The input audio mixture is encoded into a sequence of discrete tokens using a frozen DAC encoder, which are then fed into a decoder-only transformer. The transformer is trained to autoregressively predict the groundtruth drum one-shot tokens by minimizing two losses: onset loss and full-length loss. Finally, the predicted token sequence is decoded into drum one-shot audio using the DAC decoder.

The decoder-only transformer autoregressively predicts drum one-shot tokens

$$\hat{q} \in \{1, \dots, N\}^{(T_{out} \cdot f_c) \times K},$$

conditioned on the mixture-audio tokens. The DAC decoder then converts these tokens back to audio waveforms. DOSE is trained separately for each drum type (kick, snare, and hi-hat), following the structure of decoder-only language models [6] and employing the interleaving delay pattern from [23].

### B. Training Loss

Our training objective combines two cross-entropy losses: (1) *full-length cross-entropy*, computed over all predicted tokens in the drum one-shot, and (2) *onset loss*, emphasizing the attack or transient portion by focusing on tokens with indices  $t + k \leq K + 1$ , where  $t$  is the codec frame index and  $k$  is the codebook index. These transient regions strongly influence perceived timbre [24].

The final loss used for training is summation of the full-length loss and the onset loss:

$$\mathcal{L} = \mathcal{L}_{full-length} + \mathcal{L}_{onset}.$$

This setup biases the model to focus on transient regions, improving the perceptual fidelity of generated one-shot samples.

## IV. DATASET

Existing drum sound datasets are not well-suited for one-shot drum extraction, as they often lack proper alignment between mixed audio tracks and their corresponding drum one-shot samples (e.g., kick, snare, hi-hat). This limitation makes it challenging for models to learn one-shot drum extraction compared to drum stem separation tasks, for which more established datasets exist.

To address this gap, we developed the Random Mixture One-shot Dataset (RMOD), a large-scale dataset that includes numerous pairs of randomly mixed music loops and the corresponding drum one-shot samples. These pairs serve as the basis for training and evaluating one-shot drum extraction models.

The overall data generation process for RMOD is illustrated in Figure 3. Drum one-shot samples were first used to create drum loops, which were then mixed with instrumental loops such as guitar, piano, and bass to form complete musical mixtures.

For the drum one-shot samples, we leveraged publicly available datasets [25]–[27]. Instrumental loops were sourced from the Logic Pro [28] library. In total, we collected 3,375 kick samples, 1,801 snare samples, 1,278 hi-hat samples, 454 piano samples, 1,161 guitar samples, 1,782 bass samples, and 202 vocal samples. Using augmentation and mixing techniques described in next sections, we

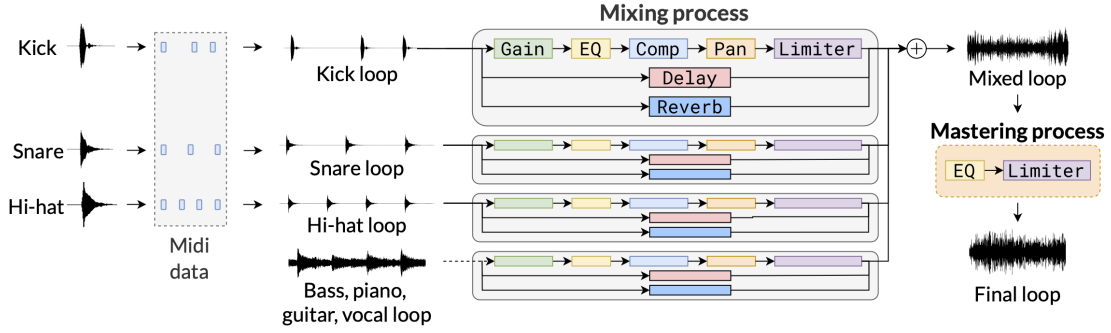


Fig. 3. Dataset generation process. First, kick, snare, and hi-hat loops are synthesized from one-shot drum audio samples using randomly generated MIDI notes. Next, optional bass, piano, guitar, and vocal loops are selected. The drum loops and other musical loops are then processed through independent mixing chains, which apply gain, EQ, compression, panning, limiting, delay, and reverb effects. Finally, all tracks are combined and passed through a mastering chain consisting of EQ and limiter effects.

generated one million pairs of randomly mixed music mixtures and their corresponding drum one-shot samples for training, with an additional 10,000 pairs each for validation and testing.

The RMOD dataset has been made publicly available through a dedicated repository, along with detailed documentation to facilitate its use by the broader research community.

#### A. Drum One-Shot Augmentation

To increase the diversity of the RMOD dataset and reflect real-world production techniques, we applied drum layering as a data augmentation method. Two drum one-shot samples were randomly selected and layered with amplitude weights of (0.8, 0.2), (0.7, 0.3), and (0.6, 0.4), corresponding to decibel reductions of approximately (-1.94 dB, -13.98 dB), (-3.01 dB, -10.46 dB), and (-4.44 dB, -7.96 dB). This weighted sum approach generated diverse drum combinations. This method effectively expanded the dataset, providing the model with a richer and more diverse set of training examples.

#### B. Loop Generation Process

To efficiently generate a large and diverse dataset, RMOD does not aim to closely mimic the structure of real music tracks. Instead, we employed a process of random mixing to create four-second loops, which allows for the rapid creation of a large number of training samples.

**MIDI Generation:** We began by creating MIDI (Musical Instrument Digital Interface) note sequences to control the timing and placement of drum sounds within each loop. Using `miditoolkit` [29] Python library, each four-second loop was divided into 1,920 equally spaced grid points, with kick drum sounds randomly occurring 2 to 4 times, snare drums 2 to 4 times, and hi-hats 14 to 18 times. This random variability in timing and placement provided the model with a wide range of drum sequences, improving its generalization to real-world scenarios.

**Audio Rendering from MIDI:** Once the MIDI sequences were created, each note was mapped to a corresponding drum one-shot sample from RMOD. If a new onset occurred for the same drum class while a previous sample was still playing, the earlier sound was sliced to prevent overlap within that class. This behavior reflects real-world scenarios, such as rapid hi-hat strikes, and does not affect overlapping sounds from different drum classes. For instrumental tracks, we used guitar, piano, bass, and vocal loop samples to introduce additional variability. Each instrument had a 30% chance of being excluded when creating the mixed loops, allowing for diverse instrument configurations in the dataset. To introduce further variation, instrumental

loops were randomly sliced into four- or two-second segments. Pitch shifting was then applied using `librosa` [30] Python library, with bass loops shifted between -6 and +2 semitones, and other instruments shifted between -12 and +12 semitones.

#### Mixing and Mastering Simulation:

To reduce the domain gap between the dataset and professionally produced music, we applied a series of digital audio effects during the loop generation process. In the mixing stage, each instrument and drum track was processed with gain adjustments, equalization (EQ), compression, panning, and limiting, with additional effects such as reverb and delay applied using parallel processing chains. This processing introduced sufficient variability in the audio characteristics, enhancing the dataset’s diversity to improve the model’s ability to generalize to real-world musical contexts.

During the mastering stage, the mixed audio was subjected to final EQ and limiting adjustments, producing variable outputs reflective of diverse audio environments. The parameters of these effects were randomized to ensure the dataset covered a broad range of production styles, reducing potential domain mismatch during inference. All audio files were exported in 16-bit, 44.1 kHz stereo WAV format to maintain consistency and compatibility with standard audio processing tools. All digital audio effects were implemented using the `pedalboard` [31] and `pymixconsole` [32] Python libraries.

## V. EXPERIMENTS

In this section, we describe the models under comparison and the datasets used for evaluation. We then report experimental results and analyses.

#### A. Compared Models

We evaluate the following models:

- **Reconstruction:** This baseline measures how much distortion arises purely from passing a ground-truth one-shot sample through the DAC encoder and decoder. Since the DOSE also employs DAC to generate outputs, this *reconstruction* score serves as an upper bound on the achievable performance of DOSE.
- **LarsNet:** We employ LarsNet [19], a drum source-separation model that decomposes the drum mixture into drum stems (kick, snare, hi-hat). Because we have the drum MIDI data, the onset times of individual drum hits are known. After separating the mixture with LarsNet, we align and slice out a single one-shot drum hit.

TABLE I  
PERFORMANCE COMPARISON OF MODELS ON RMOD AND GROOVE MIDI DATASETS USING MSS, FAD\_vgg, AND FAD\_clap METRICS.

Dataset	Model	MSS ( $\downarrow$ )			FAD_vgg ( $\downarrow$ )			FAD_clap ( $\downarrow$ )		
		kick	snare	hihat	kick	snare	hihat	kick	snare	hihat
RMOD	Reconstruction	1.832	1.880	1.530	0.207	0.360	0.621	0.133	0.131	0.135
	LarsNet	5.153	6.191	5.840	4.414	6.641	3.004	1.119	1.361	1.158
	DOSE (w/o onset loss)	4.435	4.262	3.767	1.153	1.566	1.105	0.487	0.718	0.470
	DOSE	<b>3.700</b>	<b>4.135</b>	<b>2.056</b>	<b>0.920</b>	<b>0.671</b>	<b>0.238</b>	<b>0.439</b>	<b>0.636</b>	<b>0.324</b>
RMOD (drum only)	Reconstruction	1.832	1.880	1.530	0.207	0.360	0.621	0.133	0.131	0.135
	LarsNet	4.841	5.932	5.695	2.738	5.553	2.503	0.887	1.207	1.028
	DOSE (w/o onset loss)	4.518	4.316	3.793	2.409	1.649	0.937	0.497	0.706	0.448
	DOSE	<b>3.708</b>	<b>3.966</b>	<b>1.972</b>	<b>0.978</b>	<b>0.705</b>	<b>0.217</b>	<b>0.435</b>	<b>0.605</b>	<b>0.301</b>
Groove MIDI Dataset	Reconstruction	2.243	1.461	1.476	2.364	3.276	1.935	0.236	0.306	0.234
	LarsNet	5.079	4.411	4.752	4.674	3.286	<b>2.467</b>	1.259	1.098	1.309
	DOSE (w/o onset loss)	3.606	4.041	<b>3.356</b>	3.453	3.799	4.491	0.690	1.170	<b>0.624</b>
	DOSE	<b>3.347</b>	<b>3.984</b>	3.631	<b>3.369</b>	<b>1.819</b>	5.152	<b>0.655</b>	<b>1.089</b>	0.933

- **DOSE:** Our proposed *generation-based* method that bypasses source separation. We also conduct an ablation study by comparing the DOSE model trained *with* and *without* the onset loss, to assess its impact on generating high-fidelity one-shot samples.

### B. Datasets

We use three datasets to evaluate the above models:

- **RMOD:** The *Random Mixture One-shot Dataset* introduced in this paper, which includes one million training samples and 10,000 samples each for validation and testing. Each sample is a 4-second mixture along with paired drum one-shots.
- **RMOD Drums-only:** A subset of the RMOD dataset containing only drum tracks. This subset is used to further evaluate how well each model performs when mixtures primarily consist of drums, reducing the interference from other instruments.
- **Groove MIDI Dataset:** A dataset that contains real human-performed drum MIDI files [33]. To convert these MIDI tracks to audio, we used *Logic Pro* drum kits *not* used in building the RMOD dataset. This ensures more realistic drum performances and out-of-domain generalization testing for each model.

### C. Metrics

We use two objective metrics to evaluate the quality of extracted or generated drum one-shot samples:

**Fréchet Audio Distance (FAD).** FAD [9] measures the distance between two distributions of audio embeddings. In our experiments, we employ two embedding models : VGGish [34], CLAP [35]. Both versions compare embeddings from the generated audio against a reference distribution, derived from ground-truth drum one-shots in the test set.

**Multi-Scale Spectral Similarity (MSS).** MSS [36] evaluates how closely a generated audio sample matches a reference audio sample in terms of its time-frequency representation. To compute the multi-scale spectral similarity (MSS), we first transform the generated and reference signals into spectrograms at multiple scales, specifically using FFT window lengths of 2048, 1024, 512, 256, 128, and 64. We then compute the mean squared error (MSE) between the spectrograms at each scale and aggregate them.

### D. Results and Discussion

Table I summarizes the performance of each model across RMOD, RMOD Drums-only, and the Groove MIDI Dataset. We report both

MSS and FAD (with VGG and CLAP embeddings). The following key observations emerge:

- Reconstruction exhibits a lower bound of the codec’s inherent loss, as it simply measures the quality of passing the ground-truth one-shot through the DAC encoder-decoder.
- DOSE significantly outperforms LarsNet, particularly for high-frequency percussion (e.g., hi-hat). The version of DOSE trained with onset loss further improves transient accuracy, reducing spectral errors and improving perceptual quality.
- However, DOSE’s performance in extracting hi-hat sounds declines on the Groove MIDI Dataset. This degradation is likely due to the presence of additional percussive elements (e.g., toms, crash cymbals, rims) that are absent in RMOD’s focused kick–snare–hi-hat configuration. These extra percussion sources introduce complex spectral overlaps and transient interferences, making it more challenging for DOSE to accurately extract hi-hat sounds.

## VI. CONCLUSION

In this paper, we introduced the task of Drum One-Shot Extraction, aiming to generate drum one-shot samples from a given music mixture. To tackle this task, we proposed the Random Mixture One-Shot Dataset (RMOD), containing one million training samples and 10,000 validation and test samples, each consisting of a music mixture paired with the corresponding drum one-shot samples.

We also introduced Drum One-Shot Extractor (DOSE), a neural audio codec-based model designed to generate high-quality drum one-shots from complex music inputs. Using objective metrics such as Fréchet Audio Distance (FAD) and Multi-Scale Spectral Loss, DOSE outperformed the baseline model, LarsNet, across multiple datasets, demonstrating its ability to generate perceptually accurate drum sounds.

A limitation of this work is the lack of paired data from real commercial music, which we aim to address in future work. Additionally, we plan to extend this approach to other instruments, enabling the generation of high-quality one-shots for a broader range of instruments, further enhancing music production possibilities.

# REFERENCES

- [1] C. Aouameur, P. Esling, and G. Hadjeres, “Neural Drum Machine: An interactive system for real-time synthesis of drum sounds,” in *Proc. Int. Conf. on Computational Creativity*, 2019.
- [2] A. Ramires, P. Chandna, X. Favory, E. Gómez, and X. Serra, “Neural percussive synthesis parameterised by high-level timbral features,” in *ICASSP 2020 – IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 786–790, 2020.
- [3] J. Drysdale, M. Tomczak, and J. Hockman, “Adversarial synthesis of drum sounds,” in *Proc. 23rd Int. Conf. on Digital Audio Effects (DAFx2020)*, pp. 167–172, 2020.
- [4] J. N. Hurler, S. Latner, and G. Richard, “DrumGAN: Synthesis of drum sounds with timbral-feature conditioning using generative adversarial networks,” in *Proc. 21st Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, 2020.
- [5] A. Lavault, A. Roebel, and M. Voiry, “StyleWaveGAN: Style-based synthesis of drum sounds with extensive controls using generative adversarial networks,” in *Proc. 19th Sound and Music Computing Conf. (SMC)*, 2022.
- [6] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, “Simple and controllable music generation,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [7] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar, and K. Kumar, “High-fidelity audio compression with improved RVQGAN,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [8] J. Gillick, A. Roberts, J. Engel, D. Eck, and D. Bamman, “Learning to groove with inverse sequence transformations,” in *Proc. Int. Conf. on Machine Learning (ICML)*, 2019.
- [9] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, “Fréchet Audio Distance: A metric for evaluating music-enhancement algorithms,” *arXiv:1812.08466*, 2018.
- [10] X. Wang, S. Takaki, and J. Yamagishi, “Neural source-filter waveform models for statistical parametric speech synthesis,” *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 28, pp. 402–415, 2019.
- [11] D. P. Kingma and M. Welling, “Auto-Encoding Variational Bayes,” in *Proc. ICLR*, 2014.
- [12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, “Generative adversarial networks,” *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.
- [13] Y. Song, J. Sohl-Dickstein, D. P. Kingma, *et al.*, “Score-based generative modeling through stochastic differential equations,” *arXiv:2011.13456*, 2020.
- [14] S. Rouard, C. Sony, G. Hadjeres, and C. Sony, “CRASH: Raw-audio score-based generative modeling for controllable high-resolution drum sound synthesis,” 2023.
- [15] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, “OPEN-UNMIX—A reference implementation for music source separation,” *J. Open Source Softw.*, vol. 4, no. 41, p. 1667, 2019.
- [16] R. Hennequin, A. Khilif, F. Voituret, and M. Moussallam, “Spleeter: A fast and efficient music-source-separation tool with pre-trained models,” *J. Open Source Softw.*, vol. 5, no. 50, p. 2154, 2020.
- [17] A. Défossez, N. Usunier, L. Bottou, and F. Bach, “Demucs: Deep extractor for music sources with extra unlabeled data remixed,” *arXiv:1909.01174*, 2019.
- [18] Y. Luo and N. Mesgarani, “Conv-TasNet: Surpassing ideal T-F magnitude masking for speech separation,” *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 27, no. 8, pp. 1256–1266, 2019.
- [19] A. I. Mezza, R. Giampiccolo, A. Bernardini, and A. Sarti, “Toward deep drum source separation,” *Pattern Recognit. Lett.*, vol. 183, pp. 86–91, 2024.
- [20] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, “SoundStream: An end-to-end neural audio codec,” *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 30, pp. 495–507, 2021.
- [21] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, “High-fidelity neural audio compression,” *arXiv:2210.13438*, 2022.
- [22] A. Agostinelli, T. I. Denk, Z. Borsos, *et al.*, “MusicLM: Generating music from text,” *arXiv:2301.11325*, 2023.
- [23] E. Kharitonov, A. Lee, A. Polyak, *et al.*, “Text-free prosody-aware generative spoken-language modeling,” *arXiv:2109.03264*, 2021.
- [24] R. C. Thayer Jr., “The effect of the attack transient on aural recognition of instrumental timbres,” *Psychology of Music*, vol. 2, no. 1, pp. 39–52, 1974.
- [25] A. Salimi and A. Hindle, “Drum and Percussion Kits,” Zenodo, 2020.
- [26] J. Sintes, “Yojul/one-shot-hip-hop-drums · datasets at Hugging Face,” 2020.
- [27] A. Ramires, P. Chandna, X. Favory, E. Gómez, and X. Serra, “Freesound One-Shot Percussive Sounds,” Zenodo, 2020.
- [28] Apple Inc., *Logic Pro*, version 10.7.9 [Computer software], 2024.
- [29] *miditoolkit*—Python Software Foundation, <https://github.com/YatingMusic/miditoolkit>.
- [30] B. McFee, C. Raffel, D. Liang, *et al.*, “librosa: Audio and music signal analysis in Python,” in *Proc. 14th Python in Science Conf.*, pp. 18–25, 2015.
- [31] P. Sobot, *Pedalboard*, Zenodo, 2021.
- [32] C. J. Steinmetz, J. Pons, S. Pascual, and J. Serrà, “Automatic multitrack mixing with a differentiable mixing console of neural audio effects,” *arXiv:2010.10291*, 2020.
- [33] J. Gillick, A. Roberts, J. Engel, D. Eck, and D. Bamman, “Learning to groove with inverse sequence transformations,” in *Proc. Int. Conf. on Machine Learning (ICML)*, 2019.
- [34] S. Hershey, S. Chaudhuri, D. P. Ellis, *et al.*, “CNN architectures for large-scale audio classification,” in *ICASSP 2017*, pp. 131–135.
- [35] Y. Wu, K. Chen, T. Zhang, *et al.*, “Large-scale contrastive language-audio pre-training with feature fusion and keyword-to-caption augmentation,” in *ICASSP 2023*, pp. 1–5.
- [36] X. Wang, S. Takaki, and J. Yamagishi, “Neural source-filter waveform models for statistical parametric speech synthesis,” *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 28, pp. 402–415, 2020.