D-COMPOSER: LANGUAGE MODELING FOR SIMULTANEOUS DRUM TRANSCRIPTION AND SOUND EVENT DECOMPOSITION

Patrick O'Reilly **Saumya Pailwan** **Annie Chu**
Jason Smith **Bryan Pardo**
Northwestern University, Evanston, USA
patrick.oreilly2024@u.northwestern.edu

ABSTRACT

Drum beats are often conceptualized and produced as arrangements of repeated single-drum sound events or *one-shots*. Drum Analysis-by-Synthesis (AbS) models combine this structural assumption with reconstruction-based training to perform MIR tasks such as drum transcription, drum source separation, and one-shot extraction, offering a flexible alternative to more traditional task-specific supervision. However, leading AbS approaches struggle to generalize to a broad range of drum timbres (e.g., electronic drum kits), rely on high-quality paired audio-transcription training data, and do not model intra-class timbral variation within a single arrangement (e.g., multiple kick sounds). To overcome these limitations, we propose **D-Composer**, an autoregressive language modeling framework that decomposes drum recordings into labeled sound events and their onset times. In drum transcription and one-shot extraction tasks, D-Composer performs comparably to or better than a state-of-the-art AbS system and task-specific baselines while generalizing to a broader range of drum timbres. Our results demonstrate that when equipped with abundant synthetic data and compact acoustic token representations, language modeling provides a simple recipe for unifying transcription and sound event decomposition.

1. INTRODUCTION

Drums are a cornerstone of many musical traditions. Within popular Western genres, drums are typically conceptualized (and correspondingly notated and produced) as arrangements of repeated single-drum sound events or *one-shots* spanning drum classes such as kick, snare, and hi-hat [1–3]. While this discrete structure is important to the creation of music [4–6], audio recording and mixing processes combine drum sound events into representations (e.g, PCM) that make it extremely difficult to access individual one-shots. "Factorized" music recording formats that preserve arrangement structure at the note level (e.g., MPEG-4 Structured Audio [7]) have struggled with adoption, in part because there is no reliable way to automatically extract and label note-level audio from recorded tracks.

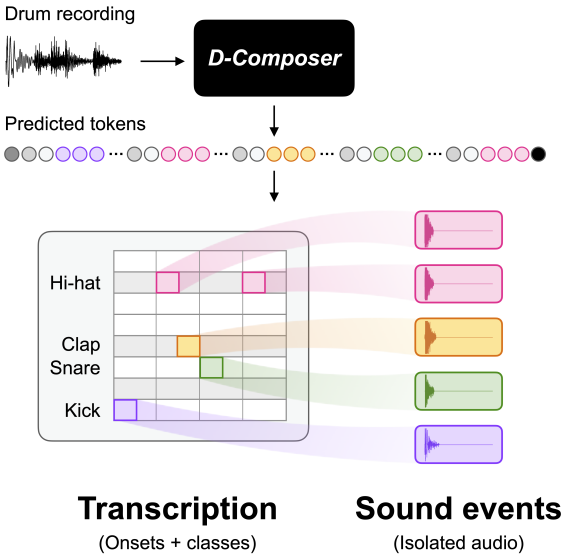

Figure 1. D-Composer maps drum audio to a stream of tokens (see Section 3.2) encoding the onset time, class, and isolated audio for all sound events, effectively turning any drum recording into a playable piano roll.

MIR research has sought to develop methods for recovering this latent structure, including tasks such as *drum transcription* (the mapping of a drum recording to a score-like representation indicating the onset time and class of each sound event), *drum source separation* (the mapping of a drum recording to one recording per drum class, each capturing all sound events of that class), and *drum one-shot extraction* (the mapping of a drum recording to one representative sound event recording per drum class). The resulting task-specific models can recover individual aspects of the latent structure (e.g., event times) but cannot fully decompose a drum recording into a manipulable arrangement of discrete sound events such as a piano roll [2, 8], limiting creative applications.

Recent work in drum Analysis-by-Synthesis (AbS) has shown that a single model can be trained to perform transcription, separation, and one-shot extraction simultaneously by reconstructing drum audio from a predicted tran-

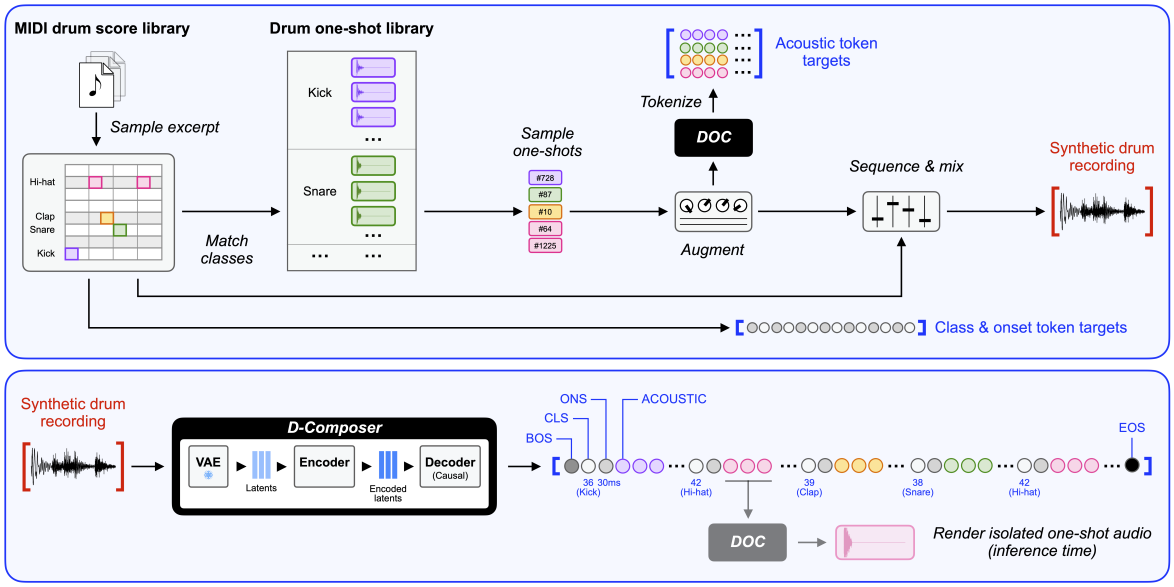

Figure 2. Training of the proposed D-Composer system, including data preparation (top) and target formatting (bottom). Using MIDI drum scores and one-shot data, we construct synthetic drum recordings as model inputs (bracketed in red) and corresponding drum class, onset, and acoustic tokens for each constituent one-shot as targets (bracketed in blue).

script [9] – closing the gap towards full decomposition. However, leading AbS approaches struggle to generalize to a broad range of drum timbres (e.g., electronic drum kits), rely on high-quality paired audio-transcription training data, and do not model intra-class timbral variation within a single arrangement (e.g., multiple kick sounds).

To overcome these limitations, we propose **D-Composer**, an autoregressive language modeling framework that fully decomposes drum recordings into arrangements of discrete sound events. Given a drum recording, D-Composer produces a structured sequence of tokens representing each sound event's onset time (in quantized 10ms intervals), the sound event's drum class (as a MIDI note), and the isolated sound event itself (as a sequence of acoustic tokens that can be mapped to audio through a purpose-built low-bitrate codec). Notably, whereas AbS performs analysis and synthesis in sequence, our framework organizes analysis and synthesis into a *single interleaved process* that produces a combined symbolic-acoustic output.

In drum transcription and one-shot extraction experiments on both real and synthetic recordings, D-Composer performs comparably to or better than a state-of-the-art AbS system and task-specific baselines while generalizing to a broader range of drum timbres, validating the effectiveness of our approach in "spelling out" drum beats as sequences of sound events. Our contributions are as follows:

- A language modeling framework (D-Composer) that fully decomposes a drum beat recording into a sequence of discrete sound events
- An ultra-low-bitrate **d**rum **o**ne-shot **c**odec (**DOC**) that compresses 44.1kHz stereo drum sound events at 0.1kbps, an order of magnitude smaller than leading high-fidelity codecs [10, 11]
- Transcription and one-shot extraction experiments demonstrating that the proposed system's decompositions accurately locate and reconstruct a drum recording's constituent sound events

Through this work, we aim to provide a tool for mapping drum beats to piano roll-like representations in which individual sound events can be easily accessed and manipulated to adjust timing and arrangement, extract individual sound sources, or perform other music production tasks. We provide listening examples and code at `https://d-composer.github.io`.

2. RELATED WORK

Recent works have applied deep learning methods to drum production and analysis tasks, including drum transcription [8,12–18], symbolic and audio-domain drum beat generation [19–21], drum source separation [22], drum one-shot extraction [2], and drum one-shot synthesis [23–26]. Earlier drum transcription methods were tailored to acoustic drum kit timbres and class taxonomies, limiting their usefulness for timbrally distinct genres such as hip-hop. Recent works have sought to improve the generalization of transcription systems by moving from small annotated datasets of acoustic kit recordings to larger augmented and synthetic datasets [14, 15, 27], and from frame-level discriminative modeling to conditional generative modeling [17]. Similarly, recent work in drum source separation and one-shot extraction has used large-scale, high-quality simulated drum data [22] and generative modeling over synthetic datasets [2] to achieve strong performance.

As deep learning-based methods for drum analysis and production have improved, Analysis-by-Synthesis (AbS) has emerged as a promising alternative to single-task supervised learning. Initial work in AbS leveraged

reconstruction-based training and sparsity regularization to learn *unsupervised* drum transcription [13]. More recently, Torres et al. proposed the Inverse Drum Machine (IDM), an AbS method that leverages reconstruction-based training and transcript supervision to unify drum transcription, source separation, and one-shot extraction within a single model [9]. IDM achieves strong performance across these tasks at a size of only 500,000 parameters, demonstrating the benefits of AbS training. However, IDM's design presents three major trade-offs: it predicts dense, sample-level transcripts that require post-processing to sparsify; it yields a single one-shot recording per drum class, preventing the modeling of intra-class timbral variations; and it is trained on simulated drum recordings spanning a narrow range of acoustic drumkit timbres. **In this work, we propose an alternative framework for training models to simultaneously identify and isolate sound events.** By moving from audio-domain AbS to language modeling over symbolic-acoustic token streams, our framework naturally produces sparse, event-level transcripts; enables modeling of intra-class timbral variations by rendering all sound events individually; and leverages synthetic data to cover diverse acoustic and electronic drum timbres.

Beyond AbS, our proposed method parallels recent works in audio understanding for language models that ground analysis tasks in timestamped transcript-like predictions [28–30]. In particular, our work can be seen as related to the timestamped audio captioning method of Kumar et al. [29], which maps an audio recording to a readout of sound event onset times, classes, and descriptive captions. However, our method differs in that descriptive captions are replaced with compact acoustic token sequences that can be used to recover isolated sound events themselves, enabling a range of musical interactions.

The Cerberus method of Manilow et al. also performs simultaneous transcription and separation of musical mixtures, demonstrating the value of joint supervision across multiple tasks [31]. Whereas Cerberus predicts two-dimensional piano roll-like transcript representations in a single step and performs mask-based separation of full instrument classes, we decompose drum recordings into isolated sound events and predict transcripts via language modeling over an event-based token representation. In the transcription space, our "decomposition" framing also has precedent in the work of Wu & Lerch [32], who perform drum transcription via NMF.

Finally, our deep learning-based mapping of audio recordings to piano roll-like representations has precedent in the Recomposer work of Ellis et al. [33]. However, whereas Recomposer performs transcript-guided resynthesis of general sound mixtures, our approach decomposes a drum beat into isolated sound events and a transcript that arranges them, enabling us to synthesize mixtures through simple summing.

3. METHOD

We next describe the components of the proposed D-Composer system. As illustrated in Figure 2, D-Composer

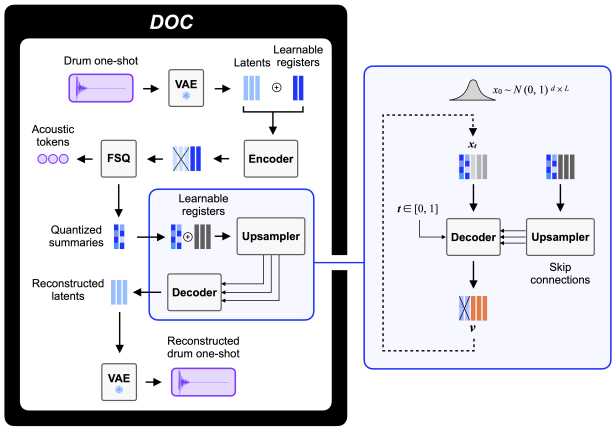

Figure 3. The proposed Drum One-Shot Codec (DOC). FSQ outputs are grouped to obtain the acoustic tokens (shown as purple circles) used by the D-Composer system.

leverages a simple autoregressive language model (Section 3.4) to map drum recordings to sequences of interleaved transcript tokens and acoustic tokens that respectively locate and characterize each sound event (Section 3.2). As ground-truth decompositions are scarce for in-the-wild drum data, we propose to synthesize suitable training data from more widely available drum score and one-shot data sources (Section 3.3). To function efficiently, our system relies on a low-bitrate drum one-shot codec or DOC (Section 3.1) to encode isolated sound events as short sequences of acoustic tokens and to correspondingly decode such sequences to audio. The compact representation afforded by DOC allows us to maintain reasonable decomposition sequence lengths even for drum recordings that contain many sound events (as each event requires the prediction of a separate DOC token subsequence). The following sections cover each system component in more detail.

3.1 Drum One-Shot Codec (DOC)

Our method predicts sound events as discrete token sequences, leveraging a codec to convert between tokens and drum one-shot audio. Existing high-fidelity codecs [10,11] require hundreds of tokens to represent a 3-second one-shot, meaning a decomposition of a drum recording containing 50 sound events would require on the order of 10^4 tokens. While feasible for contemporary language modeling approaches, this sequence length imposes significant compute and memory requirements. We therefore train a custom drum one-shot codec (DOC) to map between 3-second one-shot audio recordings and token sequences of length 32, reducing sequence lengths by an order of magnitude and allowing for training on consumer GPUs.

The design of DOC is illustrated in Figure 3. Similar to Pasini et al. [11], we pool an input representation (obtained from a frozen pretrained VAE) to a fixed-length sequence of summary vectors using learnable registers; these summary latents are then quantized per-dimension using Finite Scalar Quantization (FSQ) [34] and passed to an upsampling module that restores the input sequence length via

additional learnable registers. Following recent works that show the benefit of generative decoding for highly compressed latent representations [11, 35, 36], we use a flow-matching [37] decoder to reconstruct the input latents by refining a corresponding sequence of noise vectors, conditioned on the flow timestep t, the summary latents, and the intermediate activations from the upsampler.

While our approach is compatible with any VAE or invertible representation, we opt for the MelodyFlow VAE [38] due to its encoding speed and ability to render 48kHz stereo audio. Our encoder, upsampler, and decoder are standard non-causal transformers [39] with rotary positional encodings [40], 12 layers of dimension 512, and 8 heads per layer for a total of 140m parameters. We use a single summary vector of dimension 128, 5 quantization levels per dimension, and group sets of 4 adjacent dimensions to obtain $\frac{128}{4} = 32$ product codes [41] with vocabulary size $5^4 = 625$. DOC is trained end-to-end using only the standard flow-matching loss, achieving roughly full utilization of the FSQ codebook.

3.2 Interleaved Token Representation

We predict drum decompositions as interleaved sequences of DOC acoustic tokens (representing the isolated audio of each sound event) and transcript tokens (representing the onset time and class of each sound event). These structured token sequences can be mapped to manipulable representations such as piano rolls by decoding predicted acoustic tokens to obtain one-shots and arranging decoded one-shots according to predicted transcript tokens (see Figure 1).

Following a beginning-of-sequence (BOS) token, each sound event in a decomposition is represented by a drum class (CLS) token, an onset time (ONS) token, and 32 acoustic tokens that can be mapped to audio via DOC (see Section 3.1); after all sound events, a token sequence is terminated with an end-of-sequence (EOS) token. Sound events are ordered ascending by ONS with a secondary sort on CLS where two events share an onset. ONS tokens are drawn from a vocabulary of 1000 onsets in $[0, 9990]$ms quantized to 10ms intervals, CLS tokens from a vocabulary of 12 MIDI notes following a simplified general MIDI percussion mapping [42] (see Table 1), and acoustic DOC tokens from a vocabulary of size 625. Our token representation is illustrated in Figure 2. Similar to event-based symbolic music tokenizations [43, 44], our representation is naturally sparse (we do not expend tokens on silence) and supports variable numbers of sound events.

3.3 Training Data

Training D-Composer requires drum beat recordings with known ground-truth decompositions. To this end, we synthesize training data by sequencing one-shot recordings according to drum MIDI scores, with the former providing timbral diversity and the latter realistic drum arrangements.

We obtain drum scores from the Lakh MIDI [45] and Groove MIDI [19] datasets, normalizing notes to match the aforementioned drum vocabularies. We obtain one-shots from the Logic Pro stock library [1], the Native Instruments Battery4 library [2], and the Drum and Percussion Kits dataset [46]. To match one-shots with scores, we map each one-shot recording to a MIDI note from our drum class vocabulary using available MIDI tags (in the case of some Logic Pro data) or classes inferred from filenames. The resulting one-shot distribution is shown in Table 1.

MIDI	Class	Count
36	kick	9007
37	rim	220
38	snare	6643
39	clap	1196
42	hat	4331
43	tom	4191
49	cymbal	9845
54	tambourine	350
56	cowbell	129
67	perc	4479
70	shaker	759
71	other	3708
	Total	**44858**

Table 1. Drum vocabulary with representative MIDI notes and aggregated one-shot counts.

The preparation of our training data is illustrated in Figure 2. To construct a training example, we first sample an excerpt from a MIDI drum score. For all notes present, we sample up to four one-shots in the corresponding drum class to allow for intra-class timbral variation. Using labels present in the Logic Pro and Battery4 data, we require that 40% of training instances sample all one-shots from within the same "drum kit" family to ensure exposure to timbrally coherent mixtures. We augment one-shots of each class with random speed change, distortion (hyperbolic tangent and clipping), filtered delay, reverb, and pass-filtering to further boost timbral diversity. Similarly, we apply random time shift, timing jitter, speed change, and velocity jitter to the MIDI excerpt. We then sequence one-shots according to the MIDI excerpt, randomly selecting between sampled variations of each class at every occurrence. The sequenced one-shots are gain-scaled according to the corresponding MIDI velocities and summed, and the resulting mixture is further augmented with reverb, equalization, and dynamic range compression to simulate realistic recording conditions. Target CLS and ONS tokens are extracted from the augmented MIDI sequence, while target acoustic tokens are obtained by passing the selected one-shots through both the class-level and mixture-level audio transformations – effectively assuming linearity in our audio processing – and then encoding with DOC.

3.4 D-Composer Language Model

Our proposed D-Composer system uses a straightforward encoder-decoder transformer language model trained end-to-end with teacher-forced cross-entropy loss. A non-causal transformer encoder maps a drum beat input – represented, as in DOC, by a sequence of MelodyFlow latents

[1] https://www.apple.com/logic-pro/

[2] https://www.native-instruments.com/en/products/komplete/drums/battery-4/

Codec	SR (Hz)	Stereo	Bitrate (kbps)	#Tok	MSL ↓
DAC [10]	44.1k	✗	7.7	2322	0.31
CoDiCodec [11]	44.1k	✓	2.3	512	0.30
DOC (ours)	48k	✓	0.1	32	0.37
MelodyFlow [38]	48k	✓	–	–	0.26

Table 2. Codec compression (as measured by **Bitrate** and encoded token sequence length **#Tok**) and reconstruction performance (as measured by multi-scale spectral loss **MSL**) on 3-second drum one-shot recordings. DOC operates on continuous latents obtained from the MelodyFlow VAE, which therefore bounds its reconstruction quality.

– to a sequence of continuous vectors that are passed to a causal decoder transformer via cross-attention. The decoder then predicts a token sequence representing a decomposition of the input recording into isolated sound events, their class labels, and their onset times, as detailed in Section 3.2. We use the same underlying transformer architecture (positional encoding, etc.) as DOC, with 12 encoder and decoder layers of dimension 512 with 8 heads each for a total of 90m parameters.

4. EXPERIMENTS

We validate our proposed method's ability to faithfully decompose drum recordings into discrete sound events through a transcription evaluation (measuring the accuracy of predicted drum classes and sound event onsets), a one-shot extraction evaluation (measuring the quality of sound event isolation), and a reconstruction evaluation (measuring the "round-trip" distance between a sequenced/summed decomposition and its source audio). Additionally, we compare the reconstruction quality and compression factor of DOC against leading audio codecs.

4.1 Models

DOC: We train for 80k steps at batch size 32 on a 90% split of one-shots from our combined dataset (see Section 3.3) trimmed/padded to 3 seconds. We sample flow timesteps from the squared uniform distribution $U[0,1]^2$, train with EMA decay 0.9999, and use AdamW [47] with base learning rate 10^{-4} and exponential decay 0.999996 per step. Total training time on 2× RTX 4090 GPUs is 14 hours. At inference, we decode acoustic tokens using 5 linearly-spaced flow steps and Euler sampling.

D-Composer: We train for 100k steps at batch size 16. Following the method discussed in Section 3.3, we train on 4-second beats synthesized from MIDI sequences of up to 50 events sampled from the Lakh MIDI dataset and Groove MIDI train split, with one-shots from the same training split as DOC. We use the same optimization configuration as DOC but omit EMA. Total training time on 2× RTX 4090 GPUs is 46 hours. At inference, we sample from all D-Composer models with nucleus 0.95 [48] and constrain predicted token sequences to ensure correct structure (i.e., allow only the sampling of an onset token after a class token, an acoustic token after an onset token, and so on).

Baselines: On transcription, one-shot extraction, and reconstruction tasks we compare against the state-of-the-art AbS baseline IDM [9]. We additionally compare against the task-specific ADTOF transcription model [8] and DOSE one-shot extraction model [2]. For our codec reconstruction experiments, we compare DOC against the high-fidelity codecs DAC [10] and CoDiCodec [11], as well as the underlying MelodyFlow VAE [38] (which provides an upper bound on DOC reconstruction quality).

4.2 Metrics

We measure transcription accuracy using the standard per-class F1 score with onset tolerance 50ms [17]. To facilitate comparisons, we format transcript predictions using the 5-class drum vocabulary of ADTOF (kick – BD, snare – SD, tom – TT, hi-hat – HH, cymbal – CY), onto which the IDM and D-Composer vocabularies can be mapped cleanly.

We measure one-shot extraction quality through the multi-scale spectral loss between predicted and ground-truth one-shots using the same 5-class mapping, which is supported by both IDM and D-Composer; for DOSE we report the supported subset of BD, SD, HH. Both IDM and DOSE predict a single one-shot per class, while D-Composer predicts a separate one-shot for every sound event in the recording; we therefore select a random representative one-shot prediction of each class from D-Composer's output. We use the MultiScaleSTFT-Loss implementation in the `audiotools` library [3] with MSE on magnitudes and spectrogram window lengths of $[2048, 1024, 512, 256, 128, 64]$ as in Hwang et al. [2].

Finally, we use the same multi-scale spectral loss to measure reconstruction performance of D-Composer and IDM on full drum beats (i.e. the round-trip distance between input drum recordings and the predicted reconstructions) and to evaluate the reconstruction performance of DOC and baseline codecs on drum one-shot recordings. We note that while our use of spectral losses to quantify model performance follows prior work [2, 9], such losses offer only a coarse assessment of audio similarity and do not fully correspond with human perceptual judgements [49]. We leave more fine-grained human evaluations to future work and encourage readers to listen to the examples provided on our supplementary webpage [4].

4.3 Evaluation Data

We evaluate D-Composer on three tasks across four datasets, totaling 4500 drum beat recordings. To ensure fair comparisons with DOSE and IDM, we downsample D-Composer outputs from 48kHz to 44.1kHz and downmix from stereo to mono. We normalize all input recordings to −16dB RMS, as we find that all evaluated transcription systems exhibit some sensitivity to loudness.

Transcription: we evaluate transcription accuracy on data from three sources: 1000 4-second excerpts from the Expanded Groove MIDI (E-GMD) dataset [27], which contains aligned transcripts and drum recordings synthesized

[3] `https://github.com/descriptinc/audiotools`

[4] `https://d-composer.github.io`

Model	Transcription F1 ↑			One-shot MSL ↓	Reconstruction MSL ↓			
	E-GMD	MDB	Synthetic	Synthetic	E-GMD	MDB	Synthetic	FSL10K
ADTOF [8]	.68/.70/.29/.27/.16	.95/.84/.10/.68/.29	.74/.55/.01/.37/.09	—	—	—	—	—
DOSE [2]	—	—	—	.22/.21/—/.23/—	—	—	—	—
IDM [9]	.51/.57/.27/.41/.11	.84/.72/.10/.43/.12	.29/.34/.02/.37/.05	.22/.21/.14/.23/.54	1.04	1.74	2.12	2.15
D-Composer	.55/.59/.20/.30/.13	.86/.74/.02/.48/.22	.78/.61/.02/.40/.15	.19/.18/.14/.21/.52	0.94	1.54	1.78	1.92

Table 3. Results of our transcription, one-shot extraction, and round-trip reconstruction experiments. Transcription F1 scores and one-shot spectral distances are reported by the ADTOF 5-class vocabulary of KD/SD/TT/HH/CY, while reconstruction spectral distance is computed on full drum recordings. Metrics for tasks or drum classes not supported by a model are indicated with a dash (–).

using acoustic and electronic timbres on a Roland TD-11 drum kit; 1000 4-second excerpts from the MDB-Drums dataset [50], which contains aligned transcripts and real-world drum recordings from the MedleyDB dataset [51]; and 2000 4-second excerpts synthesized in the manner described in Section 3.3 from unseen Groove MIDI sequences and one-shots, omitting both MIDI and one-shot augmentations and using a single representative one-shot for each drum class. These data sources therefore cover a wide range of drum timbres and production processes.

One-shot extraction: We use the same synthetic dataset as for transcription (2000 4-second excerpts), as the presence of a single timbre per class per recording allows for well-defined ground truth one-shots.

Reconstruction: We evaluate round-trip reconstruction on the three aforementioned datasets (E-GMD, MDB-Drums, synthetic) in addition to 500 4-second excerpts from the Freesound Loop Dataset [52] obtained by filtering for recordings with only "drum" instrument tags, representing a wide range of sonic palettes and audio qualities.

Codec comparisons: Finally, we compare codec reconstruction quality on 2000 unseen one-shots from our combined dataset, trimmed/padded to 3 seconds. As in our D-Composer evaluations, audio is resampled to 44.1kHz and downmixed to mono prior to computing metrics.

We note that our training data does not overlap with ADTOF. We match the train/test split of GMD scores and Logic Pro one-shots used by IDM's StemGMD [22] in our own data. Our one-shots overlap with the training data of DOSE via the Drum and Percussion Kits dataset [46].

5. DISCUSSION

Our experimental results demonstrate D-Composer's ability to accurately locate and isolate sound events within drum recordings, showing the viability of our language modeling approach for drum decomposition.

The results of our transcription evaluation are presented in Table 3. D-Composer performs comparably to the state-of-the-art AbS IDM model on the heavily acoustic MDB-Drums dataset and partially acoustic E-GMD dataset, and achieves higher average accuracy than both IDM and the task-specific ADTOF system on synthetic data that draws from a broader range of electronic drum timbres. These results do not demonstrate the superiority of D-Composer as a transcription engine – performance is heavily affected by training domain, and both ADTOF and IDM achieve strong performance at smaller model scales. Rather, these results demonstrate that by training to predict symbolic-acoustic readouts from synthetic data, our framework teaches the model to accurately localize sound events in real drum recordings.

The results of our one-shot extraction evaluation are presented in Table 3. D-Composer achieves lower average MSL than both the AbS IDM model and task-specific DOSE model, indicating its ability to isolate representative one-shot timbres from mixture recordings.

The results of our round-trip reconstruction evaluation are presented in Table 3. D-Composer achieves lower average reconstruction MSL than the AbS IDM model on both in-the-wild, heavily electronic data (FSL10K) and professionally-recorded, majority-acoustic data (MDB), indicating its ability to produce decompositions that accurately recover the timing and timbre of source recordings.

Finally, the results of our codec reconstruction experiment are presented in Table 2. For three-second one-shot recordings, DOC produces token sequences over 10 times shorter than state-of-the-art acoustic codecs at the expense of degraded audio quality, demonstrating a clear compression/fidelity trade-off. While our round-trip reconstruction and one-shot extraction experiments show that this fidelity is sufficient to compete with baseline systems in downstream tasks, the development of a higher-quality low-bitrate codec stands out as a direction for future work.

Taken together, these results demonstrate the benefits of our simple language modeling approach for decomposing sound scenes into constituent isolated sound events and corresponding arrangements. These decompositions can be mapped to piano roll-like representations, allowing for high-level arrangement edits of drum recordings (e.g., removing hi-hats, adjusting speed) and fine-grained adjustments (e.g., timing correction on individual onsets) – and potentially opening up a range of musical interactions in the process.

Future work should explore the development of interfaces for musical decomposition systems; improved low-bitrate acoustic codecs; modeling of expanded MIDI drum class vocabularies and additional attributes such as MIDI velocity; and methods for integrating real-world, annotation-free mixtures into the end-to-end training of decomposition systems via e.g. reinforcement learning.

6. ACKNOWLEDGEMENTS

This work was supported by NSF Awards Number 2222369 and 2300633. We would also like to thank Mark Cartwright for productive discussions and Fangze Li for assistance in formatting Figures 1-3.

7. ETHICS STATEMENT

This work proposes a creative tool for decomposing drum recordings into interpretable and manipulable representations, facilitating music production tasks (editing, transcription, one-shot extraction, drum source separation) and encouraging interactive exploration of musical materials. We address ethical implications and limitations of this work below.

The D-Composer and DOC models presented in this paper were trained on transcripts and one-shot recordings from corpora dominated by Western popular music and may perform poorly on out-of-domain timbres and rhythms from other musical traditions. This limitation may be addressed in future work by expanding the training data to reflect a broader set of musical traditions.

Our one-shot training data was sourced from royalty-free commercial libraries (Native Instruments Battery4 and Logic Pro), as existing public-domain one-shot datasets suffer from size and quality limitations. The licenses of our chosen libraries prohibit the redistribution of assets in "repackaged" form even in non-commercial contexts. While these licenses do not assert whether AI models trained on assets may constitute such a repackaging, we initially refrain from releasing model weights as we work to resolve this ambiguity. We emphasize that the models trained in this work are intended for educational and research purposes, and we do not propose their use for commercial purposes.

Finally, we acknowledge the environmental impact of the research presented in this work. We estimate the energy consumption of each D-Composer training run (46 hours on $2\times$ NVIDIA RTX 4090) at $\frac{450\mathrm{W}\times 2}{1000}\times 46\mathrm{h} = 41.4\mathrm{kWh}$, and the energy consumption of each DOC training run (14 hours on $2\times$ NVIDIA RTX 4090) at $\frac{450\mathrm{W}\times 2}{1000}\times 14\mathrm{h} = 12.6\mathrm{kWh}$ using the reported $450\mathrm{W}$ power draw of our cards. Model inference was performed on a single GPU for a small fraction of training time, and we therefore omit this cost from our estimates. At roughly 15 DOC runs and 5 D-Composer runs, we estimate the total energy consumption of our experiments at $396\mathrm{kWh}$, or 175% the mean training cost for ISMIR papers published in 2017–2023 as estimated by Holzapfel et al. [53]. We note that, while approximate, this energy consumption is likely orders of magnitude below that of leading academic music models [44,54], and is equivalent to about six charges of a Hyundai Ioniq 5 electric car.

8. REFERENCES

[1] M. Mauch and S. Dixon, "A corpus-based study of rhythm patterns," in *Proceedings of the 20th International Society for Music Information Retrieval Conference*, 2012.

[2] S. Hwang, S. Kang, K. Kim, S. Ahn, and K. Lee, "Dose: Drum one-shot extraction from music mixture," in *ICASSP*, 2025.

[3] O. Nieto, G. J. Mysore, C.-i. Wang, J. B. Smith, J. Schlüter, T. Grill, and B. McFee, "Audio-based music structure analysis: Current trends, open challenges, and applications," *Transactions of the International Society for Music Information Retrieval*, vol. 3, no. 1, 2020.

[4] R. Polak, N. Jacoby, T. Fischinger, D. Goldberg, A. Holzapfel, and J. London, "Rhythmic prototypes across cultures: A comparative study of tapping synchronization," *Music Perception*, vol. 36, no. 1, pp. 1–23, 2018.

[5] O. Lartillot, T. Eerola, P. Toiviainen, and J. Fornari, "Multi-feature modeling of pulse clarity: Design, validation and optimization." in *Proceedings of the 9th International Society for Music Information Retrieval Conference*, 2008, pp. 521–526.

[6] Z. Rafii and B. Pardo, "Repeating pattern extraction technique (repet): A simple method for music/voice separation," *IEEE transactions on audio, speech, and language processing*, vol. 21, no. 1, pp. 73–84, 2012.

[7] *Information technology – Coding of audio-visual objects – Part 3: Audio, Subpart 5: Structured Audio*, ISO/IEC JTC 1/SC 29/WG 11 Std. ISO/IEC 14 496-3, 1999, mPEG-4 Structured Audio (SAOL, SASL).

[8] M. Zehren, M. Alunno, and P. Bientinesi, "High-quality and reproducible automatic drum transcription from crowdsourced data," *Signals*, vol. 4, no. 4, pp. 768–787, 2023. [Online]. Available: https://www.mdpi.com/2624-6120/4/4/42

[9] B. Torres, G. Peeters, and G. Richard, "The inverse drum machine: Source separation through joint transcription and analysis-by-synthesis," *IEEE Transactions on Audio, Speech and Language Processing*, 2026.

[10] R. Kumar, P. Seetharaman, I. K. Alejandro Luebs, and K. Kumar, "High-fidelity audio compression with improved rvqgan," in *NeurIPS*, 2023.

[11] M. Pasini, S. Lattner, and G. Fazekas, "CoDiCodec: Unifying continuous and discrete compressed representations of audio," in *ISMIR*, 2025.

[12] R. Vogl, M. Dorfer, and P. Knees, "Recurrent neural networks for drum transcription," in *Proceedings of the 17th International Society for Music Information Retrieval Conference*, 2016.

[13] K. Choi and K. Cho, "Deep unsupervised drum transcription," in *ISMIR*, 2019.

[14] P. Melucci, P. Merialdo, and T. Akama, "Towards realistic synthetic data for automatic drum transcription," *arXiv preprint arXiv:2601.09520*, 2026.

[15] M. Cartwright and J. P. Bello, "Increasing drum transcription vocabulary unsing data synthesis," in *DAFx*, 2018.

[16] Y. Wang, J. Salamon, M. Cartwright, N. J. Bryan, and J. P. Bello, "Few-shot drum transcription in polyphonic music," in *ISMIR*, 2020.

[17] M. Yeung, K. Toyama, T. Teramoto, S. Takahashi, and T. Kojima, "Noise-to-notes: Diffusion-based generation and refinement for automatic drum transcription," in *ICASSP*, 2026.

[18] A. Santos, A. Cardoso, M. E. P. Davies, and R. B. Dannenberg, "Tapping into a new paradigm: A synthetic strategy for automatic drum tapscription," in *NIME*, 2025.

[19] J. Gillick, A. Roberts, J. Engel, D. Eck, and D. Bamman, "Learning to groove with inverse sequence transformations," in *ICML*, 2019.

[20] A. Caillon and P. Esling, "Rave: A variational autoencoder for fast and high-quality neural audio synthesis," *arXiv preprint arXiv:2111.05011*, 2021.

[21] P. O'Reilly, J. Barnett, H. F. García, A. Chu, N. Pruyne, P. Seetharaman, and B. Pardo, "The rhythm in anything: Audio-prompted drums generation with masked language modeling," in *ISMIR*, 2025.

[22] A. I. Mezza, R. Giampiccolo, A. Bernardini, and A. Sarti, "Toward deep drum source separation," p. 86–91, 2024. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2024.04.026

[23] A. Lavault, A. Roebel, and M. Voiry, "Stylewavegan: Style-based synthesis of drum sounds with extensive controls using generative adversarial networks," in *Proceedings of the Sound and Music Computing Conference (SMC)*, 2022.

[24] J. Drysdale, M. Tomczak, and J. Hockman, "Style-based drum synthesis with gan inversion," in *Extended Abstracts for the Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[25] J. Nistal, S. Lattner, and G. Richard, "Drumgan: Synthesis of drum sounds with timbral feature conditioning using generative adversarial networks," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[26] S. Rouard and G. Hadjeres, "Crash: Raw audio score-based generative modeling for controllable high-resolution drum sound synthesis," in *ISMIR*, 2021.

[27] L. Callender, C. Hawthorne, and J. Engel, "Improving perceptual quality of drum transcription with the expanded groove midi dataset," *arXiv preprint arXiv:2004.00188*, 2020.

[28] J. Jeong, P. Mousavi, M. Ravanelli, and C. Subakan, "Listen first, then answer: Timestamp-grounded speech reasoning," *arXiv preprint arXiv:2603.19468*, 2026.

[29] S. Kumar, P. Seetharaman, K. Chen, O. Nieto, J. Su, Z. Wang, R. Kumar, D. Manocha, N. J. Bryan, Z. Jin, and J. Salamon, "Tac: Timestamped audio captioning," *arXiv preprint arXiv:2602.15766*, 2026.

[30] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *ICML*, 2023.

[31] E. Manilow, P. Seetharaman, and B. Pardo, "Simultaneous separation and transcription of mixtures with multiple polyphonic and percussive instruments," in *ICASSP*, 2020.

[32] C.-W. Wu and A. Lerch, "Drum transcription using partially fixed non-negative matrix factorization with template adaptation," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, Oct. 2015, pp. 257–263. [Online]. Available: https://archives.ismir.net/ismir2015/paper/000199.pdf

[33] D. P. W. Ellis, E. Fonseca, R. J. Weiss, K. Wilson, S. Wisdom, H. Erdogan, J. R. Hershey, A. Jansen, R. C. Moore, and M. Plakal, "Recomposer: Event-roll-guided generative audio editing," *arXiv preprint arXiv:2509.05256*, 2025.

[34] F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen, "Finite scalar quantization: Vq-vae made simple," in *ICLR*, 2024.

[35] Y. Wang, Z. Tang, Y. Wang, A. Hinsvark, Y. Liu, Y. Li, K. Peng, J. Ao, M. Ma, M. Seltzer, Q. He, and X. Liu, "Scaling speech tokenizers with diffusion autoencoders," in *ICLR*, 2026.

[36] Z. Lahrichi, G. Hadjeres, G. Richard, and G. Peeters, "S-presso: Ultra low bitrate sound effect compression with diffusion autoencoders and offline quantization," *arXiv preprint arXiv:2602.15082*, 2026.

[37] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *ICLR*, 2023.

[38] G. L. Lan, B. Shi, Z. Ni, S. Srinivasan, A. Kumar, B. Ellis, D. Kant, V. Nagaraja, E. Chang, W.-N. Hsu, Y. Shi, and V. Chandra, "High fidelity text-guided music editing via single-stage flow matching," *arXiv preprint arXiv:2407.03648*, 2024.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.

[40] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, no. C, Mar. 2024. [Online]. Available: https://doi.org/10.1016/j.neucom.2023.127063

[41] R. Langman, A. Jukić, K. Dhawan, N. R. Koluguri, and B. Ginsburg, "Spectral codecs: Spectrogram-based audio codecs for high quality speech synthesis," 2024.

[42] *General MIDI Level 1 Specification*, MIDI Manufacturers Association Std., 1994.

[43] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer," in *ICLR*, 2018.

[44] J. Thickstun, D. Hall, C. Donahue, and P. Liang, "Anticipatory music transformer," *TMLR*, 2024.

[45] C. Raffel, "Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching," Ph.D. dissertation, Columbia University, 2016.

[46] A. Salimi and A. Hindle, "Drum and percussion kits," https://doi.org/10.5281/zenodo.3994999, 2020.

[47] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[48] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," in *ICLR*, 2020.

[49] P. Manocha, A. Finkelstein, R. Zhang, N. J. Bryan, G. J. Mysore, and Z. Jin, "A differentiable perceptual audio metric learned from just noticeable differences," in *Interspeech*, 2020.

[50] C. Southall, C.-W. Wu, A. Lerch, and J. Hockman, "Mdb drums: An annotated subset of medleydb for automatic drum transcription," in *ISMIR*, 2017.

[51] 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 *ISMIR*, Taipei, Taiwan, Oct. 2014.

[52] A. Ramires, F. Font, D. Bogdanov, J. B. L. Smith, Y.-H. Yang, J. Ching, B.-Y. Chen, Y.-K. Wu, H. Wei-Han, and X. Serra, "The freesound loop dataset and annotation tool," in *ISMIR*, 2020.

[53] A. Holzapfel, A.-K. Kaila, and P. Jääskeläinen, "Green mir?: Investigating computational cost of recent music-ai research in ismir," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2024.

[54] J. Liu, Y. Dong, H. Liu, Y. Cheng, Z. Guo, H. Liang, W. Zhan, Y. Sun, X. Li, F. Yu, and M. Sun, "Shao: Scaling acoustic token language models toward high-fidelity music generation," *arXiv preprint arXiv:2605.01790*, 2026.