The Inverse Drum Machine: Source Separation Through Joint Transcription and Analysis-by-Synthesis

Bernardo Torres, Geoffroy Peeters, Gaël Richard

***Abstract*—We present the Inverse Drum Machine (IDM), a novel approach to Drum Source Separation that leverages an analysis-by-synthesis framework combined with deep learning. Unlike recent supervised methods that require isolated stem recordings, our approach operates on drum mixtures with only transcription annotations. IDM integrates Automatic Drum Transcription and One-shot drum Sample Synthesis, jointly optimizing these tasks in an end-to-end manner. By convolving synthesized one-shot samples with estimated onsets, akin to a drum machine, we reconstruct the individual drum stems and train a Deep Neural Network on the reconstruction of the mixture. Experiments on the StemGMD dataset demonstrate that IDM achieves separation quality comparable to state-of-the-art supervised methods that require isolated stems data, while significantly outperforming matrix decomposition baselines.**

***Index Terms*—audio source separation, deep learning, signal processing, analysis-by-synthesis**

IN Western popular music, the rhythmic foundation typically relies on percussion instruments from a standard drum kit comprising kick drum, snare drum, and hi-hat, while additional elements such as cymbals, tom-toms, and auxiliary percussions provide timbral complexity and rhythmic variation. Music producers and engineers often need to adjust individual drum instruments separately for remixing, rebalancing, effects processing, or creating educational materials [1], [2]. Ideally, music production would utilize isolated recordings of each drum instrument (known as "stems"), allowing for precise control during mixing. However, these instruments are usually played simultaneously and by the same performer, resulting in recordings in which all elements are mixed into a single audio stream. Obtaining these separated stems during recording requires multiple microphones (leading to microphone bleeding) or asking musicians to play in unnatural conditions [3]. The need for tools that can extract individual drum stems from already mixed recordings has led to growing interest in Drum Source Separation (DSS).

In the professional music production industry, commercial tools such as DrumsSSX[1] and Fadr[2] offer drum stem separation capabilities. These solutions, however, are proprietary and still have limitations in separation quality and flexibility.

DSS is challenging due to the acoustic properties of percussion sounds. Many percussive sounds lack a clear harmonic structure and exhibit broad spectral characteristics that often overlap in both time and frequency domains. Additionally, there exists a wide diversity of percussive instruments, classified roughly into membranophones (e.g., kick drum, snare drum, toms) and idiophones (e.g., cymbals, hi-hats) [4]. While "drum", more strictly defined, refers to membranophones, we use the term broadly to encompass all percussive instruments in this work for simplicity. Each instrument can produce several sounds depending on the playing technique, further complicating the separation task. In addition to these complexities, drum-related music information research has received relatively limited attention, possibly reflecting Western music's historical emphasis on melodic and harmonic elements rather than rhythmic foundations found in traditions such as Indian tabla playing or African percussion-centered music.

In a broader context, Music Source Separation (MSS) has gained significant attention due to the impressive performance of deep learning methods. These approaches have achieved state-of-the-art results by training Deep Neural Networks (DNNs) in a supervised manner using ground-truth separated stems [5]–[7]. However, MSS benchmarks typically focus on isolating just four classes: vocals, drums, bass, and other instruments, considering the drum section as a single stem.

Recent works have capitalized on multitrack synthetic datasets generated by drum machines and programming software [7], [8] for training DSS models. Simultaneously, the MSS literature has shown promising results from joint separation and transcription approaches [9]. Complementing these developments, data-driven analysis-by-synthesis methods have demonstrated advances in reconstruction-based techniques that don't require ground-truth data [10]–[12].

In this work, we propose an analysis-by-synthesis, multitask approach to DSS. By modeling drum tracks as sequences of one-shot samples triggered at precise times, similar to a drum machine, we train a DNN to recover both the transcription and one-shot samples from the reconstruction of the mixture. Drum source separation is then achieved via audio synthesis from the "inverted" drum machine. Due to the strong inductive bias in our approach, we can bypass the need for isolated stems during training while achieving results comparable to state-of-the-art methods.

Conceptually, our approach is similar of drum replacement in music production, where drum tracks undergo manual onset detection and are replaced with one-shot samples from a sample library. We propose instead to perform this process in an automated, data-driven manner while learning to synthesize the actual one-shot samples from the mixture.

Our method, IDM, combines approaches from Automatic Drum Transcription (ADT), unsupervised source separation, and drum synthesis literature [11]–[14] to tackle DSS without requiring separated stems. Our main contributions are:

- A novel approach to DSS that leverages an analysis-by-

B. Torres, G. Peeters, and G. Richard are with the Laboratoire de Traitement et Communication de lInformation (LTCI), Télécom Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France. (e-mail: {bernardo.torres, geoffroy.peeters, gael.richard}@telecom-paris.fr)

[1]https://fuseaudiolabs.de/#/pages/product?id=300867907

[2]https://fadr.com/drum-stems

synthesis framework combined with deep learning, jointly optimizing ADT and One-shot drum Sample Synthesis (OSS) end-to-end.
- Comprehensive evaluation using objective metrics that highlight different aspects of separation quality, demonstrating superior results compared to learning-free baselines and comparable performance to deep learning approaches that require ground-truth multitrack data.
- An open-source implementation of our method, enabling reproducibility and further research in the field of drum source separation[3].

I. Related Work

This section presents relevant prior research in four interconnected areas that form the foundation of our approach: Automatic Drum Transcription (ADT), One-shot drum Sample Synthesis (OSS) , Drum Source Separation (DSS), and Data-driven analysis-by-synthesis.

A. Automatic Drum Transcription (ADT)

ADT aims to detect and classify percussion events within audio recordings, producing a symbolic representation (typically onset times and instrument classes) of the drum performance [15]. While ADT encompasses various formulations, this work focuses on Drum Transcription of Drum-only recordings (DTD), where the input consists exclusively of percussion instruments without melodic content.

Approaches to DTD have evolved from traditional signal processing methods to machine learning techniques. Non-negative Matrix Factorization (NMF) and its variants have had lots of success [16] modeling spectral patterns of drum sounds without requiring labeled training data. More recent methods leverage deep learning, with both supervised [17], [18] and unsupervised [11] approaches showing significant performance improvements. Most DTD research has focused on a limited set of drum instruments, typically kick drum, snare drum, and hi-hat [2], [16], while some recent works have expanded the instrument vocabulary [11], [18]–[20].

B. One-shot Drum Sample Synthesis

Percussion instruments typically produce sounds with broad spectral characteristics and exponentially decaying modes [21]. Drum sound synthesis has a rich history in commercial music production and academic research. Traditional approaches include physical modeling [21], [22], which simulates the acoustic properties of percussion instruments; spectral modeling [23], [24], which represents sounds as sinusoids, filtered noise, and a transient component; and sample-based synthesis, which triggers pre-recorded audio samples.

Recent advances in generative deep learning have tackled drum sound synthesis research from a data-driven perspective, using convolutional neural networks [25] and generative adversarial networks [26]–[28]. Other techniques employ diffusion models [29] and Differentiable Digital Signal Processing (DDSP) [14], [30]. With particular relevance to this work, Shier et al. [14] proposed pairing a DDSP sinusoidal plus noise synthesizer with a transient enhancing Temporal Convolutional Network (TCN) to generate drum sounds.

[3]https://github.com/bernardo-torres/inverse-drum-machine

C. Drum Source Separation (DSS)

Traditional DSS approaches have primarily relied on NMF-based methods, which model drum events as spectral templates repeatedly triggered over time [31]. These approaches decompose a spectrogram into a product of template and activation matrices, representing the spectral characteristics of different drum sounds and their temporal occurrences, respectively.

Conceptually, our approach shares several similarities with works based on Non-negative Matrix Factor Deconvolution (NMFD) [32] which incorporate transcription information into the initialization process [2], [33]. However, our method differs in several crucial aspects: (1) we operate directly in the time domain, providing immediate access to one-shot samples without requiring magnitude spectrogram inversion; (2) unlike NMFD, which updates spectral templates at inference time through multiplicative update rules, our approach incorporates a dedicated training stage that optimizes neural network parameters via gradient descent.

In DSS literature, NMFD methods have also been used in a cascaded manner [33] to reduce cross-talk between instruments, and tested for dual-channel input [34]. Vande Veire et al. [13] introduced Sigmoidal NMFD, which enforces impulse-like behavior in the updated activations by factoring them into an onset and an amplitude component, and updating them using gradient descent.

Despite their effectiveness in controlled settings, NMFD-based methods have limited flexibility and cannot learn from data since they operate at test time. Moreover, while these approaches have demonstrated reasonable performance on datasets with few instrument classes, they have not been extensively evaluated on large-scale datasets with numerous percussion types.

A significant advancement in DSS came with the development of the StemGMD dataset [7], which leveraged MIDI transcriptions from the GMD dataset [35], captured from performances on an electronic drum kit, to synthesize individual drum stems for nine drum classes. With it, LarsNet was introduced, the first deep neural network specifically designed for drum source separation. LarsNet separates five stems from a stereo drum mixture: kick drum, snare drum, tom-toms, hi-hat, and cymbals. A recent benchmark of deep MSS architectures trained on StemGMD [8] shows that there is room for improvement by using larger, more complex models, in particular those that are trained to perform waveform synthesis directly (rather than relying on time-frequency masking). Unlike NMFD, these DNN models require supervised training with ground-truth separated stems and often have a high parameter count, making them computationally expensive and data-intensive.

D. Data-driven Analysis by Synthesis

A growing body of research combines data-driven methods with analysis-by-synthesis approaches. Differentiable Digital

Signal Processing (DDSP) combines DNNs with Digital Signal Processing (DSP) [36], enabling end-to-end training of models that incorporate signal processing operations within their architectures.

Applications of this paradigm include estimation of synthesizer parameters [14], [22], [37], oscillator frequency estimation [38]–[40], estimation of audio effects [41], and others [42]. A typical scenario is training DNNs to predict DSP parameters for synthesis by comparing reconstructed audio to a target. Spectral loss functions, such as multi-resolution Short-Time Fourier Transform (STFT) loss, are commonly used instead of waveform loss to avoid phase alignment issues.

In source separation, analysis-by-synthesis has been employed to estimate the parameters of musical sources from mixture inputs. These parameters drive source models to resynthesize individual sources that are summed and compared to the original mixture [12], [43]. This approach has proven particularly effective for separating singing voices in choir ensembles [12], [44], where multi-pitch information guides the separation process.

In the context of ADT, reconstruction of drum tracks has been used for evaluating transcription performance through listening tests [18]. For unsupervised ADT, Choi and Cho [11] resynthesize tracks from estimated transcriptions using randomly selected one-shot samples from a class-sorted collection, training a DNN using an onset-enhanced reconstruction loss. Our work extends their method to DSS.

II. Method

In this work, we propose an analysis-by-synthesis approach to decompose drum mixtures into transcription information and elementary one-shot samples. As illustrated in Figure 1, the proposed architecture processes audio mixtures through several stages. Initially, a Feature Extraction module derives frame-level representations from the input mixture. These features are subsequently transformed by a Synthesis Conditioning module into relevant parameters for synthesis (transcription and one-shot conditioning). The conditioning is used by a one-shot synth to synthesize drum samples which are sequenced by the estimated transcription. A multitask training stage enables learning of neural network parameters for all modules.

The following sections explain the proposed multitask framework and then go into detail on the Feature Extraction, Synthesis Conditioning, and Decoder modules, which are also detailed in Figure 2. For presentation clarity, we left most implementation details to Section II-G.

A. Multitask learning for Drum Source Separation

Our approach encompasses three interconnected tasks:

1) Automatic Drum Transcription (ADT): The precise estimation of the onset times of each drum instrument is achieved by training a transcription head to predict onset activations.

2) One-shot drum Sample Synthesis (OSS): High-quality one-shot samples for each drum instrument are generated by a TCN conditioned on instrument type and mixture embedding.

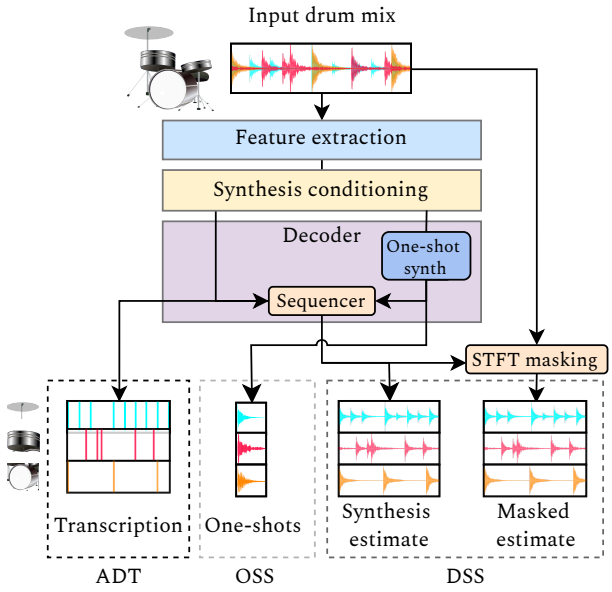

Fig. 1. The proposed separation model processes the audio input through several stages. First, a Feature Extraction module extracts learned frame-level features. These features are transformed by a Synthesis Conditioning module into relevant synthesis parameters: transcription onsets, velocities, a mixture embedding, and individual track gains. A decoder module synthesizes one-shot samples for each drum instrument conditioned on the mixture embedding and reconstructs individual drum tracks by sequencing these one-shots with the obtained transcription. The framework encompasses three interconnected tasks: Automatic Drum Transcription (ADT), One-shot drum Sample Synthesis (OSS), and Drum Source Separation (DSS). DSS is obtained either from the decoder output (*synthesis estimate*) or after time-frequency masking (*masked estimate*).

3) Drum Source Separation (DSS): Individual drum tracks are extracted from the mixture by sequencing the synthesized one-shot samples with the estimated transcription.

To train our analysis-synthesis framework, we crop random mixtures of length T from a large database of drum-only recordings. A training step begins by applying random gain to the cropped mixture, obtaining input x, from which transcription activations and a conditioning vector ($\mathbf{c}$) are extracted. Then, individual one-shot samples ($\mathbf{w}$) for each drum instrument are synthesized from $\mathbf{c}$ and sequenced into full tracks using the activations. The individual tracks are mixed to obtain a reconstructed mixture $\hat{x}$, and reconstruction loss $\mathcal{L}_{\text{recon}}$ is applied between $\hat{x}$ and x. The entire framework is trained end-to-end, with the addition of a transcription loss $\mathcal{L}_{\text{trans}}$ and a mixture embedding loss $\mathcal{L}_{\text{emb}}$ (see Section II-E for details). We use no additional data augmentations, and our total parameter count is of 465K.

B. Feature Extraction

Our Feature Extraction module learns features $\mathbf{z}$ which contain relevant information for the transcription and mixture embedding. It begins by computing Log-Mel Spectrograms from the input waveform x at 16 kHz. Then, a ConvNeXt encoder [45] processes the spectrogram to extract frame-level representations $\mathbf{z} \in \mathbb{R}^{D_z \times M}$, where $D_z = 32$ denotes the dimension of the frame features and M represents the number of spectrogram frames.

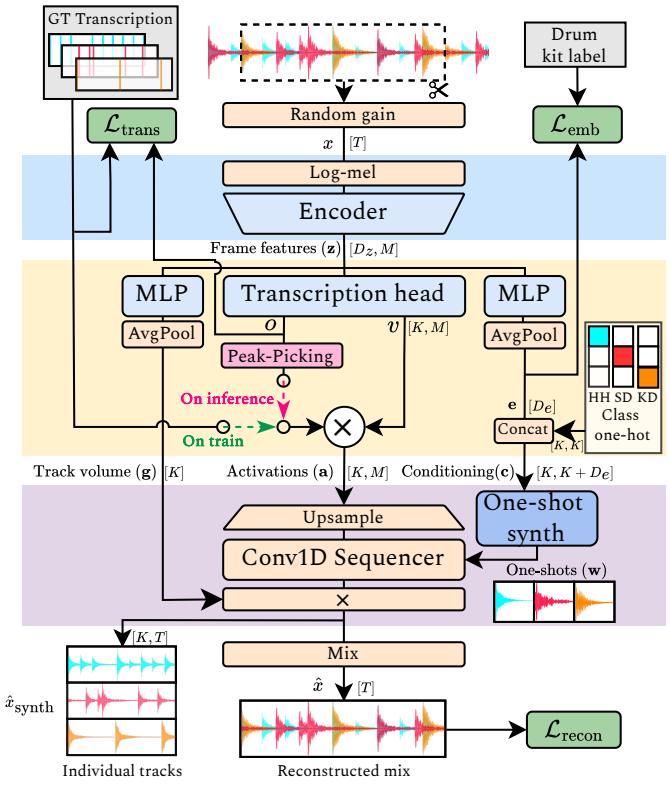

Fig. 2. Detailed diagram of the analysis-by-synthesis pipeline, containing the Feature Extraction (blue), Synthesis Conditioning (yellow), and Decoder (purple) modules. Blocks in blue indicate trainable components, blocks in orange indicate differentiable operations, and blocks in pink indicate non-differentiable operations. Loss functions are represented as green components, and grey components represent external information used during training

C. Synthesis Conditioning

The Synthesis Conditioning module transforms the frame features into parameters that guide the reconstruction process: transcription onsets, velocities, individual track gains, and a conditioning vector. Estimated onsets and velocities are multiplied and upsampled to obtain an activation signal used to trigger one-shot samples, composing the individual tracks for K predefined drum classes. The conditioning vector is used as conditioning for the one-shot synthesis model, guiding the generation of one-shot samples based on the instrument class and a mixture embedding, which controls the timbral variations within a given class.

1) Transcription head: The transcription head transforms features $\mathbf{z}$ into a factorized representation of the drum component activation signal, composed of:

- An onset signal $\mathbf{o} \in \mathbb{R}^{K \times M}$, identifying when each instrument is played.
- A velocity signal $\mathbf{v} \in \mathbb{R}^{K \times M}$, capturing the intensity of each onset.

This architecture draws inspiration from the "Onsets and Frames" [18], [46] and Sigmoidal NMFD [13] papers. The temporal resolution is set by the spectrogram's hop size.

2) Activation signal: We compute the activation signal $\mathbf{a} \in \mathbb{R}^{K \times T}$ by element-wise multiplication of onsets and velocities, followed by upsampling from the rate of the frames m to the rate of the signal time t:

$$\mathbf{a}_k[t] = \text{upsample}(\mathbf{o}_k[m] \odot \mathbf{v}_k[m]) \tag{1}$$

We upsample using zero insertion [11]. While this approach induces aliasing, we found that if the onset signal is sparse enough, the synthesis artifacts are negligible. We therefore separate training and inference behavior to minimize these effects during training:

During **training**, we pass ground-truth onsets instead of $\mathbf{o}$ to compose the activation $\mathbf{a}$. Note that the onset signal is still used for training the transcription head from the transcription loss, but we effectively stop the gradient flowing from the reconstruction process to $\mathbf{o}$. The gradient passed to the velocity branch, however, remains intact. During **inference**, we apply peak picking to extract sparse onsets from $\mathbf{o}$.

3) Mixture embedding: To represent the overall timbral characteristics of the drum-kits (not of the specific instruments), we derive a mixture embedding $\mathbf{e} \in \mathbb{R}^{D_e}$ from frame features $\mathbf{z}$. While this layer could be learned without supervision, in this work we train it to match one-hot labels constructed from drum kit annotations using a classification loss $\mathcal{L}_{\text{emb}}$. While this branch behaves as a "drum kit classifier" and restricts the model to synthesize only from drum kits observed during training, it serves as proof-of-concept for a disentangled class/timbre conditioning mechanism.

4) Conditioning vector: Recall that K is the number of drum instruments. A class conditioning vector $\mathbf{c}$ is composed by broadcasting $\mathbf{e}$ to $[K, D_e]$ and concatenating ($\oplus$) the K-dimensional one-hot encodings of the different classes in the feature dimension, resulting in size $[K, K + D_e]$. $\mathbf{c}$ can be interpreted as the stack of individual class|mixture conditioning vectors for all classes of a particular mixture ($\{\mathbf{c}_k\}_{k=1}^{K}$, where $\mathbf{c}_k = \mathbf{e} \oplus \text{one-hot}(k)$).

D. Decoder

The Decoder module synthesizes one-shot samples for each drum instrument and reconstructs individual drum tracks by sequencing these one-shots with the activations from the Synthesis Conditioning module. The decoder is composed of two main components: a one-shot synth and a sequencer.

1) One-shot Synth: Conditional One-shot drum Sample Synthesis is performed with a Temporal Convolutional Network (TCN), similar to the work of Shier et al. [14]. However, instead of using TCN to enhance the transient of a sinusoidal-plus-noise signal [24], we take a more direct path. We eliminate the sinusoidal-plus-noise input, instead feeding 1 second zero-padded white noise directly to the TCN. By expanding the network's receptive field and adding a time envelope, we found the TCN capable of synthesizing both sinusoidal, noise, and transient components. Figure 3 shows the architecture of the one-shot synthesis model, with specific implementation details reported in Sections II-G6 to II-G8.

To guide the synthesis process, we condition the TCN using FiLM [47] on a conditioning vector $\mathbf{c}$ composed of two disentangled dimensions: a mixture embedding $\mathbf{e}$ and the instrument class one-hot encoding. To model the natural

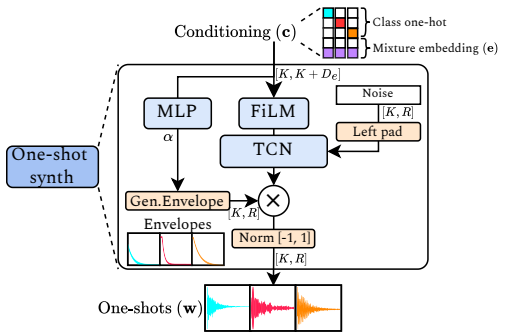

Fig. 3. One-shot synth model architecture. White noise is fed into a Temporal Convolutional Network (TCN) conditioned via Feature-wise Linear Modulation (FiLM) on a conditioning vector $\mathbf{c}$, which has disentangled instrument class/timbre dimensions. Depending on input $\mathbf{c}$, the TCN allows for synthesizing various drum instruments from the different drum kits. The TCN output is shaped by a parametrized exponential envelope. The final output is normalized to $[-1, 1]$ amplitude range.

energy decay typical of percussion instruments, an exponential amplitude envelope is multiplied to the TCN output, controlled by a parameter α dependent on $\mathbf{c}$.

Unlike conventional OSS approaches [14], [27], [29], we train the synthesizer model on mixture reconstructions rather than isolated one-shot samples, making it inherently reliant on accurate onset placement.

2) Sequencer: The sequencer serves as the bridge between transcription and synthesis, transforming drum instrument activations and one-shot samples into full individual tracks. Let $\mathbf{w} \in \mathbb{R}^{K \times \mathrm{R}}$ represent a set of time-domain one-shot signals (one second in duration, R denoting the sampling rate), $\mathbf{g} \in \mathbb{R}^{K}$ denote a vector of global track gains. The reconstructed mixture $\hat{\mathbf{x}}_{\text{synth}} \in \mathbb{R}^{T}$ is obtained by:

$$\hat{\mathbf{x}}_{\text{synth}} = \sum_{k=1}^{K} \mathbf{g}_k \, \text{Sequencer}(\mathbf{w}_k, \mathbf{a}_k) \tag{2}$$

The sequencer is implemented as a convolutional operator using frequency domain multiplication [11].

E. Training Objective

The system is trained end-to-end using a combination of reconstruction, transcription, and mix embedding losses. The reconstruction loss $\mathcal{L}_{\text{recon}}$ is computed between the synthesized mixture and the input cropped mixture. We set $\mathcal{L}_{\text{recon}}$ to be the Multi-Resolution STFT Loss [36], [48]:

$$\mathcal{L}_{\text{recon}}(x, \hat{x}) = \sum_{\gamma \in \Gamma} \Big\| |\mathbf{X}^{\gamma}| - |\hat{\mathbf{X}}^{\gamma}| \Big\|_1 + \Big\| \log(|\mathbf{X}^{\gamma}|) - \log(|\hat{\mathbf{X}}^{\gamma}|) \Big\|_1, \tag{3}$$

where $(x, \hat{x})$ are the time domain input and the reconstructed signals and $\mathbf{X}^{\gamma} = \text{STFT}^{\gamma}(x)$ is the STFT of x at scale γ. The set of scales Γ, expressed in number of samples of the analysis window, is set to $[2048, 1024, 512, 256]$, and the hop sizes are set to a quarter of the analysis window size.

Transcription loss $\mathcal{L}_{\text{trans}}$ is the Binary Cross-Entropy loss between the estimated onsets and the ground-truth onsets. To obtain ground-truth targets for training, we create a target signal of the same rate as the onset signal, setting frames closest to the onset time to 1 and the rest to 0. The mixture embedding loss $\mathcal{L}_{\text{emb}}$ is the Cross-Entropy between the estimated mixture embedding $\mathbf{e}$ and the one-hot drum kit label. The training objective is the sum of the three losses:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{trans}} + \mathcal{L}_{\text{emb}} \tag{4}$$

F. Time-frequency masking

While the separated sources can be directly obtained by synthesis, previous works showed that improved results may be obtained after using them for spectral masking of the mixture [12]. In this work, we employ α-Wiener Masking [49], which leverages our synthesized individual tracks to create Time-Frequency (TF) masks applied to the original mixture. The underlying principle is that each stem's spectrogram contributes proportionally to the energy in each TF bin of the mixture's spectrogram.

G. Implementation details

1) Log Mel Spectrogram: We compute Log Mel Spectrograms with 250 mel frequency bands, a window size of 1024 samples, and a hop size of 256 samples (16ms), corresponding to 62.5Hz activation rate.

2) Encoder Architecture: Our ConvNeXt encoder begins with a transition convolutional layer downsampling the feature dimension by 4, followed by LayerNorm [50]. Four hierarchical stages follow with feature dimensions $[4, 8, 16, D_z]$, each starting with a channel downsampling block (LayerNorm + 3×3 convolution). Each stage contains multiple residual blocks with 7×7 depthwise separable convolutions, LayerNorm, GELU activations, and pointwise transformations. Unlike the original ConvNeXt, we maintain temporal resolution between stages. As demonstrated in [20], the specifics of the architecture might not have a big impact on ADT performance. We thus opted for a modern, well-validated off-the-shelf convolutional architecture.

3) Transcription Head: The frame features $\mathbf{z}$ are processed through a 1D convolutional layer with ReLU activation, followed by parallel onset and velocity branches with linear layers. The onset branch outputs sigmoid activations, while the velocity branch uses exponentiated sigmoid activation [36]. We did not include a sequence model as we found it unnecessary for achieving reasonable transcription performance.

4) Peak Picking: We apply peak picking to the frame-rate onset signal during inference to extract sparse onsets and remove low amplitude activations. We follow the heuristic from [11], [51].

5) Mixture embedding: The mixture embedding is derived from frame features through LayerNorm, temporal average pooling, and a two-layer Multi-Layer Perceptron (MLP) with ReLU activation.

6) TCN Architecture: Our TCN [14] consists of 10 dilated causal convolutional layers with residual connections and GELU activations. Input is projected to a latent dimension of 48, processed through the dilated residual blocks (dilation doubling at each layer, starting from 1), and projected back to audio dimensions. The total receptive field is ≈ 767 ms. Kernel size is 15 throughout, with instance normalization applied to all layers and output normalized to $[-1, 1]$.

7) FiLM Conditioning: Conditioning is implemented as a stack of linear layers mapping from $\mathbf{c}$ to shift and scale parameters applied after the GELU activation in each TCN layer:

$$\text{FiLM}(\mathbf{x}_{ij}, \mathbf{c}) = \gamma_{ij}(\mathbf{c}) \cdot \mathbf{x}_{ij} + \beta_{ij}(\mathbf{c}), \tag{5}$$

where $\mathbf{x}_{ij}$ is the jth feature map of TCN layer (after GELU) i, $\mathbf{c}$ is the conditioning vector, and $\gamma_{ij}(c)$ and $\beta_{ij}(c)$ are the shift and scale parameters learned by the (i, j)th linear layer.

8) Envelope: A 2 layer MLP with ReLU activation and exponentiated sigmoid estimates envelope parameter α from the conditioning vector. The envelope is applied to the TCN output by element-wise multiplication:

$$\text{Gen. envelope}(\alpha) = e^{-20 \cdot \alpha \cdot \frac{t}{\text{R}}} \quad \text{for} \quad t \in [0, \text{R}] \tag{6}$$

9) TF Masking: Given a mixture complex STFT $\mathbf{X}$ and an estimated source STFT $\hat{\mathbf{X}}_{\text{synth},i}$, time-frequency masks $\mathbf{M}_i$ and masked spectrograms $\hat{\mathbf{X}}_{\text{mask},i}$ are computed as:

$$\mathbf{M}_i = \frac{|\hat{\mathbf{X}}_i|^\alpha}{\sum_j |\hat{\mathbf{X}}_j|^\alpha + \epsilon} \quad \text{and} \quad \hat{\mathbf{X}}_{\text{mask},i} = \mathbf{M}_i \odot \mathbf{X}, \tag{7}$$

where $\odot$ denotes element-wise multiplication and ϵ is a small constant to prevent division by zero. We denote $\hat{x}_{\text{mask}}$ the waveform estimates obtained after inverse STFT of $\hat{\mathbf{X}}_{\text{mask},i}$ using the mixture phase. α is set to 1 in order to remain consistent with [7], [33].

III. Experiments

This section describes our experimental methodology for training and testing the proposed drum source separation framework. Our primary experimental configuration utilizes a drums-only database with ground-truth transcription annotations and separated stems as a test-set. The following subsections detail the data, experimental protocols and baseline methods employed in our work.

A. Datasets

Our experiments are conducted using the StemGMD dataset [7], which builds upon the MIDI data from the GMD dataset [35] by synthesizing isolated drum stems for 9 canonical instruments across 10 distinct drum kits. StemGMD has approximately 136 hours of mixture data. Perfect transcription annotations were extracted directly from the associated MIDI files. Table I presents the mapping between the original instrument nomenclature, the names utilized in the audio files (`9-Class`), and the two instrument groupings employed in our experimental design. Training utilizes the `9-Class` configuration, while evaluation is performed on both `9-Class` and `5-Class` groupings. To maintain comparability with `LarsNet`, which withheld 4 kits during training, we utilize only 6 of the 10 drum kits available in the dataset. Consequently, all training and evaluation protocols are implemented exclusively on these 6 kits.

TABLE I
DRUM CLASS CONFIGURATIONS. INSTRUMENT NAMES CORRESPOND TO COMMON DRUM KIT TERMINOLOGY, WHILE `9-Class` AND `5-Class` REFER TO THE GROUPINGS USED IN OUR EXPERIMENTS. `9-Class` NAMING IS CONSISTENT WITH THE STEMGMD DATASET. OUR MODEL IS TRAINED ON THE `9-Class` CONFIGURATION AND IS ABLE TO SEPARATE AN INPUT MIXTURE INTO 9 DRUM CLASSES. THE `5-Class` CONFIGURATION, USED FOR EVALUATION ONLY, GROUPS SIMILAR INSTRUMENTS TOGETHER.

Instrument Name	`9-Class` (train/test)	`5-Class` (test)
Kick Drum	kick	Kick
Snare Drum	snare	Snare
Closed Hi-Hat	hihat_closed	Hi-Hats
Open Hi-Hat	hihat_open	
High-Mid Tom	hi_tom	Toms
Low Tom	mid_tom	
High Tom	low_tom	
Crash	crash_left	Cymbals
Ride	ride	

We follow the provided train, validation, and test splits on the MIDI track level. Contrary to LarsNet [7], which utilizes a small subset of the test partition named *eval session*, we evaluate our models on the entire test split, which comprises $\sim$ 10 h of mixture data. We found that the *eval session* removed substantially the toms and cymbals from the test set, and by increasing its size we achieve a more balanced evaluation.

B. Experimental setup

Our training protocol utilizes randomly cropped audio segments of $T = 8$ seconds, normalized to the amplitude range $[-1, 1]$. We apply random gain augmentation to the cropped mixtures, with gain values sampled from a uniform distribution in the range $[0.3, 1.0]$ with a probability of 0.8. Cropped tracks are resampled and converted to mono on-the-fly.

For the training phase, we configure the number of drum instruments as $K = 9$, enabling separation of the input mixtures into 9 distinct drum classes (as enumerated in Table I). The training stage consists of 800 epochs with a batch size of 33, utilizing the Adam optimizer with a learning rate of 5×10^{-3} and gradient clipping with a threshold value of 0.5. We implement model checkpointing based on the mean validation loss $\mathcal{L}_{\text{total}}$ across tracks computed on the validation set (no cropping is performed on the validation data). We sort the validation tracks by length and keep the shortest 550 tracks for a more efficient validation.

C. Baselines

We evaluate our approach against both learning-free baselines and a deep supervised baseline. Table II summarizes our

TABLE II
COMPARISON OF DIFFERENT METHODS FOR DRUM SOURCE SEPARATION, BASELINES AND PROPOSED MODEL. TRANSC. = TRANSCRIPTION.

Method	External information used		Input for eval. (kHz/channels)
	Training	Inference	
`Oracle`	–	Isolated stems	16/mono
`NMFD 1A`	–	Transc. + one-shots	16/mono
`NMFD 1B`	–	Transc. + one-shots	16/mono
`NMFD 3`	–	Transc.	16/mono
`LarsNet`	Isolated stems	–	44.1/stereo
`LarsNet Mono`	Isolated stems	–	44.1/mono
`IDM` (ours)	Transc.	–	16/mono
`IDM+ onsets`	Transc.	Transc.	16/mono

experimental comparison and the varying levels of external information utilized during training and inference phases.

1) Learning-free: The learning-free baseline methodology was proposed in [2], [33]. This approach is based on NMFD [32], wherein the mixture's spectral magnitude is modeled as the product of template and activation matrices.

Their work employs informed NMFD variants where drum transcription information and spectral templates from drum one-shots are utilized for NMFD algorithm initialization. We replicate experimental cases 1A, 1B, and 3 from their work for direct comparison with our method. In all three cases, the activation matrix is initialized with ground-truth transcription information with values of 1 where onsets occur and a small constant ϵ elsewhere.

Following [7], we utilize a StemGMD partition containing isolated one-shots synthesized with identical drum kits for template initialization for cases 1A and 1B (*single hits* partition). In case 3, spectral templates are initialized with random values.

Both 1A and 1B use transcription and template initialization, but in 1A, spectral templates remain fixed while only the activation matrix is updated. We denote these baselines by `NMFD 1A`, `NMFD 1B`, and `NMFD 3`, respectively.

We adapted the STFT parameters to accommodate our 16 kHz sample rate while maintaining time resolution and window size consistency with the original implementation. In contrast to findings reported in [7], we observed a drop in performance when increasing the number of iterations. Consequently, we conducted a grid search on the validation set, varying template length and iteration count for each case. Our hyperparameter search determined that 50 iterations resulted in optimal performance for the case with fixed templates (`NMFD 1A`), while 20 iterations were best for `NMFD 1B` and `NMFD 3`. For the template length, we found that 40, 10, and 7 yielded the best results respectively. We increased ϵ to 10^{-10} to address convergence issues observed with smaller values. The number of templates was configured to match the number of drum instruments (9). All other implementation details were preserved as specified in the original publication. `NMFD` baselines were reimplemented using TorchNMF[4].

2) Supervised DNN: We benchmark our method against the deep learning model `LarsNet`, which was trained for drum separation in a fully supervised manner using isolated stems [7] and its available open source[5]. `LarsNet` was trained on StemGMD multitrack data, utilizing 6 of the 10 available drum kits. The model has 49.1M parameters, operates at 44.1 kHz, and was trained on stereo files to separate stems in the `5-Class` configuration. `LarsNet` was designed to output soft masks for each drum class, which are applied to the mixture spectrogram to obtain separated stems. To ensure a fair comparison, we provide the model with 44.1 kHz stereo inputs and use its default soft mask to obtain 44.1 kHz stereo separated estimates. Then, to compute the performance metrics, we downsample the separated stems to 16 kHz and convert them to mono. When evaluated in the *masking* configuration, we use these estimates to compute α-Wiener masks and apply them to the mixture spectrogram at 16kHz. We also report metrics when removing spatial information from the input (`LarsNet Mono`).

3) Onsets on inference time: Due to our modular synthesis approach, we can override the class onset predictions $o_k[m]$ during inference with the ground-truth (while keeping velocities $v_k[m]$). This provides an upper performance bound by isolating transcription limitations from synthesis capabilities. This approach offers insights into the separation performance with perfect onset detection. We add `+ onsets` to the model name when reporting these results.

D. Evaluation

We implement a comprehensive evaluation framework combining transcription and separation metrics. It is noteworthy that the DSS task encompasses multiple drum classes with predominantly sparse activations (such as tom-toms), requiring careful consideration in the evaluation protocol. All metrics are computed per class, using full tracks as input, and aggregated across tracks. For all metrics except Predicted Energy in Silence (PES), evaluation is restricted to active stems, defined as stems containing at least one onset during the track duration. This methodological decision results in variable sample sizes (number of tracks) for the different classes. The "Overall" metrics reported are computed as the aggregation of scores across all tracks, flattened and independent of class. We found this methodology to best represent the actual class distribution in the dataset.

During evaluation, input mixtures are generated dynamically through summing constituent stems. For the `5-Class` evaluation configuration, we sum the stems within the same instrument groups to establish the target stems (e.g., `hihat_closed` and `hihat_open` are combined into a single `Hi-Hats` stem). We also sum model outputs accordingly.

1) Transcription: We extract discrete onsets from the estimated onset signal through peak picking, following the methodology detailed in Section II-G4. Precision and recall metrics are then computed for each drum class using the *mir_eval* python package [52]. The precision metric quantifies

[4]https://github.com/yoyolicoris/pytorch-NMF

[5]https://github.com/polimi-ispl/larsnet

the proportion of detected onsets that correspond to actual events, while recall measures the proportion of actual events that are successfully detected by the system.

Transcription evaluation is performed exclusively on our models and the `NMFD` baselines. We emphasize that our transcription performance assessment is not intended as a direct comparison with state-of-the-art drum transcription systems (which typically incorporate multiple datasets and extensive augmentation strategies), but rather as a measure of our system's intrinsic transcription capabilities and associated limitations. Moreover, even though DTD is considered a relatively simple task in ADT literature [15], [20], most works do not consider the full range of drum instruments present in our dataset.

2) Source separation: Our separation evaluation methodology is designed to assess both direct synthesis and masking-based separation capabilities. For each input mixture x, the model generates synthesized estimates $\hat{\mathbf{x}}_{\text{synth}}$ directly produced by our generative model.

We also compute masked estimates $\hat{\mathbf{x}}_{\text{mask}}$ through time-frequency masking of the input mixture (Section II-F). While masking techniques have been extensively employed in source separation literature, they exhibit known limitations including spectral leakage artifacts. Synthesis, conversely, provides a more direct assessment of the model's capacity to generate authentic drum sounds but may suffer from phase misalignment with the ground-truth. We therefore evaluate synthesis and masking outputs independently.

For each active stem, we compute both waveform and perceptual metrics between model outputs and ground-truth stems, with waveform metrics being the standard in source separation literature [6], [53].

The Scale-Invariant Signal-to-Distortion Ratio (SI-SDR) [54] quantifies separation quality in the waveform domain:

$$\text{SI-SDR} = 10 \log_{10} \left(\frac{\|\alpha s\|^2}{\|\alpha s - \hat{s}\|^2} \right), \tag{8}$$

where $\alpha = \langle s, \hat{s} \rangle / \|s\|^2$ is the optimal scaling factor between the target source s and its estimate $\hat{s}$. SI-SDR is computed exclusively for masked outputs ($\hat{x}_{\text{mask}}$).

However, spectral similarity often correlates more strongly with human perceptual judgments than signal-based methods [55]. We therefore employ the Log Spectral Distance (LSD) as proxy for perceptual similarity, computed as:

$$\text{LSD}(s, \hat{s}) = \frac{1}{T} \sum_{t=1}^{T} \left(\frac{1}{F} \sum_{f=1}^{F} \Big[\log(|\hat{\mathbf{S}}|^2 + \epsilon) - \log(|\mathbf{S}|^2 + \epsilon) \Big]^2 \right)^{\frac{1}{2}}, \tag{9}$$

where $\mathbf{S}$=STFT(s) with window length of 2048 and hop length of 512, and s and $\hat{s}$ are the estimated and ground-truth signals.

Predicted Energy in Silence (PES) [56] is employed to quantify how accurately models predict silence when the corresponding ground-truth stem is silent (inactive). PES is defined as the average energy of estimated stems during silent ground-truth frames. The estimated and ground-truth signals are segmented using non-overlapping windows of size 512, retaining only frames corresponding to inactive ground-truth segments. Energy (in decibels) of estimated stems is computed as $10 \times \log_{10}(\sum_n (x(n)^2) + 10^{-8})$. We set our minimum threshold for silent frames to -60dB, clipping values below. PES is reported exclusively for synthesized outputs, as we observed that masking-based approaches introduce cross-talk artifacts that manifest as unwanted energy during silence.

For SI-SDR, higher is better, while for LSD and PES, lower is better.

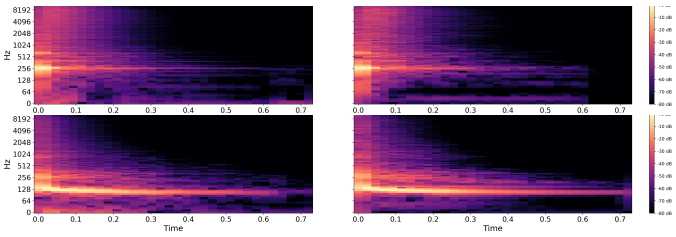

Fig. 4. Log-magnitude spectrograms of synthesized, one-second-long one-shot samples (left) and real (right) for a snare (top) and a tom (bottom) instrument. The real samples are taken from the StemGMD *single hits* partition with the second-highest velocity. Y axis is scaled in log frequency for better visibility, and warm colors represent higher intensity.

IV. Results

In this section, we present a comprehensive evaluation of our proposed Inverse Drum Machine (IDM) method and baselines, analyzing both transcription (Section IV-B) and separation (Section IV-C) performance. In addition to the objective metrics shown in this paper we have made available in an accompanying website[6] examples of the separated stems for all models discussed, along with the synthesized one-shot samples of our model.

For box plots, the central mark indicates the median of scores across tracks, the edges of the box are the 25th and 75th percentiles, and the whiskers extend to the full range of the data points.

A. One-Shot sample synthesis

Our model effectively learned to synthesize drum one-shots from 9 instruments across 6 drum kits. We provide a qualitative analysis of a few synthesized one-shots and refer the reader to the supplementary material for a full set of samples. Figure 4 shows examples of synthesized one-shots for a snare and a tom-tom, with a reference one-shot taken from the StemGMD *single hits* partition (second highest velocity). The generated samples demonstrate the model's ability to generate realistic drum sounds, capturing transient and steady-state components of each instrument.

B. Transcription Results

As shown in Figure 5, our model achieves high precision and recall values (both in the 90s percentile range) across most instruments. This strong performance is partly due to the controlled experimental conditions, as the test set utilizes the

[6]https://bernardo-torres.github.io/projects/inverse-drum-machine/

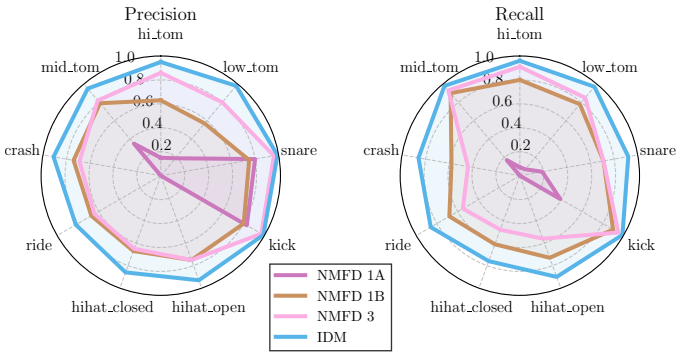

Fig. 5. Transcription performance comparison between our model and NMFD baselines.

same drum kits as the training set. Nevertheless, our model significantly outperforms NMFD baseline approaches, which have been shown to outperform unsupervised deep learning methods in previous work [11]. These results also highlight a disadvantage of NMFD-based methods: even when provided with ground-truth transcription at the algorithm's initialization, the basis optimizations lead to smoothed activations that compromise transcription accuracy.

The results reveal some variations across drum instruments. We observe particularly challenging detection patterns for closed hi-hats and ride cymbals. Despite these challenges, the overall transcription accuracy remains robust, providing a solid foundation for the subsequent separation task.

C. Separation Results

1) Masking-based separation: The results for masking-based separation are presented in Figure 6. The proposed IDM model outperforms all NMFD baselines by a substantial margin across all metrics, despite having access to transcription information during inference. Compared to the supervised LarsNet, our approach performs slightly worse on masked metrics, though we marginally outperform the LarsNet Mono configuration on LSD.

According to the SI-SDR scores, a notable challenge appears in separating cymbal and Hi-Hats, where the fixed one-shot duration of 1 second in our approach might be insufficient to model their longer decay characteristics [13], negatively affecting objective metrics without necessarily reflecting perceptual quality.

2) Direct synthesis results: Figure 7 presents the synthesis-based separation results using Log Spectral Distance (LSD) and silence prediction (PES) metrics. IDM demonstrates superior silence prediction capabilities compared to supervised baselines and achieves better LSD scores than LarsNet for most instruments except toms and cymbals. Direct synthesis produces lower LSD values than masked outputs for all classes except snare, suggesting our generative approach can produce spectrally accurate reconstructions without the artifacts introduced by masking.

3) Influence of onsets during inference: Including ground-truth onsets during inference (IDM+ onsets) consistently improves separation performance, occasionally surpassing

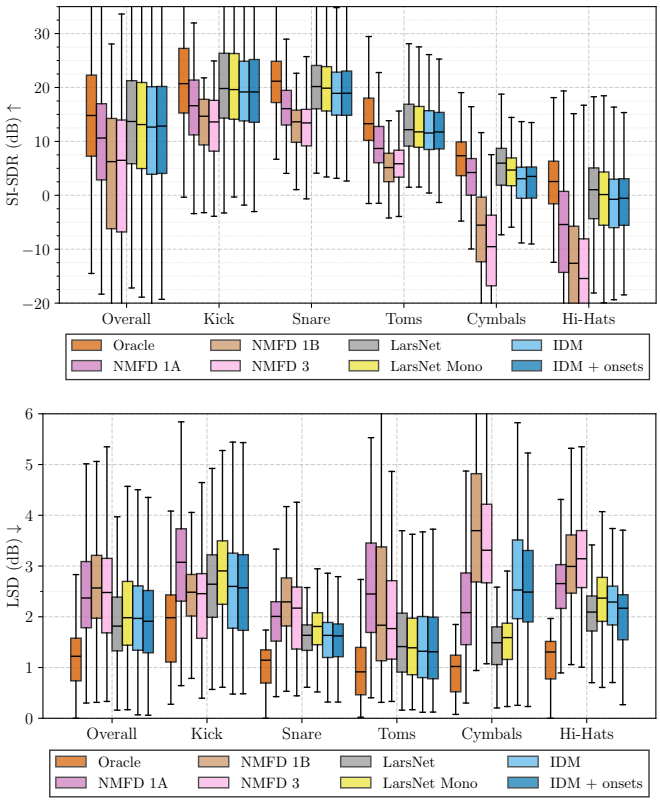

Fig. 6. Comparison of masking-based separation performance metrics. Top: Scale-Invariant Signal-to-Distortion Ratio (higher is better). Bottom: Log Spectral Distance (lower is better).

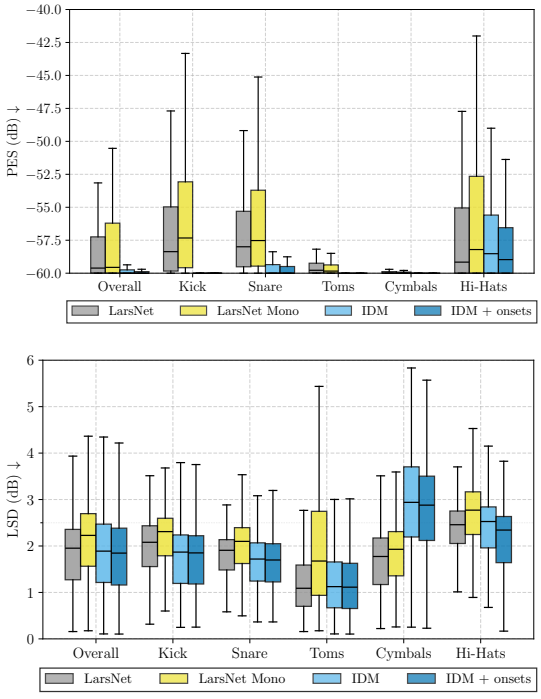

Fig. 7. Comparison of synthesis-based separation performance metrics. Top: Predicted Energy in Silence (PES). Bottom: Log Spectral Distance (LSD). For both metrics, lower is better.

TABLE III
PERFORMANCE METRICS BY INSTRUMENT CLASS

Class	Masked			Synthesized	
	SI-SDR↑	LSD↓	PES↓	LSD↓	PES↓
kick	18.80	2.55	-53.64	1.74	-54.98
snare	19.29	1.55	-56.12	1.66	-58.77
hihat_closed	-10.29	2.38	-55.54	2.39	-56.76
hihat_open	2.81	1.34	-59.83	1.46	-59.98
hi_tom	9.64	0.99	-59.55	0.83	-59.77
mid_tom	8.02	0.97	-59.69	0.73	-59.80
low_tom	8.68	1.46	-59.34	1.04	-59.68
ride	-5.35	2.01	-59.86	2.14	-59.88
crash_left	-2.12	2.91	-59.94	3.06	-59.99

`LarsNet Mono` or `LarsNet` results (e.g., on kick for masked LSD and Hi-Hats for synth LSD). This confirms that our method's optimal performance depends on accurate onset detection, though the relatively small improvement margin reflects our model's already strong transcription capabilities.

4) Results in 9-class configuration: We report model results in the full `9-Class` configuration in Table III for reference. The `9-Class` configuration provides a more detailed view of the model's performance across all drum instruments. Particularly notable is the disparity between open and closed hi-hats, which are often grouped in evaluation protocols such as that used by [7].

V. DISCUSSION

Our experimental results demonstrated that the proposed IDM approach successfully integrates transcription and synthesis for DSS. Without requiring access to ground-truth isolated stems during training, our model achieves performances comparable to the ones of supervised approaches that depend on such data while using 100 times less parameters (465K vs 49.1M). The ability to directly synthesize one-shot samples for each instrument represents an additional advantage, facilitating potential applications in music production and remixing scenarios.

The transcription component can be improved by leveraging more data, state-of-the-art techniques, data augmentation [18], [20], incorporating beat information [17], and increasing the temporal resolution of onset detection. External ADT models or even human input could be integrated to enhance separation results in practical applications.

Our evaluation protocol reveals limitations in traditional DSS assessment methods. Waveform metrics exhibit considerable variance across instrument classes, suggesting they may not optimally measure objective performance. The `9-class` results indicate significant performance disparities that challenge conventional class groupings, as exemplified by open hi-hats showing greater acoustic similarity to ride cymbals than to closed hi-hats. Finally, the sparsity of less frequently played instruments can bias track-level metrics due to the predominance of silence.

This approach could benefit musical traditions with significant percussive components, such as African or Latin American music [57], particularly in low-resource contexts where isolated stem data is scarce or unavailable.

VI. CONCLUSION

This paper presented the Inverse Drum Machine (IDM), a novel approach to Drum Source Separation (DSS) that integrates Automatic Drum Transcription and One-shot drum Sample Synthesis in an end-to-end framework. By leveraging an analysis-by-synthesis methodology, our approach achieves high-quality DSS without requiring isolated stems during training.

Experimental results demonstrated that IDM performs comparably to state-of-the-art supervised methods that require multitrack training data, while significantly outperforming matrix decomposition baselines. The modular synthesis framework and the ability to directly synthesize one-shot samples for each drum instrument provide additional flexibility for creative applications.

Future work may explore extending this approach to higher sample rates and enhancing synthesis quality. The successful conditional synthesis without exposure to ground-truth isolated targets enables potential applications with more diverse drum kits and real-world drum mixture data. Future implementations could replace the one-hot representation of drum kits with pre-trained embedding models or learned representations. Our approach could also potentially operate without transcription information by leveraging unsupervised learning approaches [11], while synthesis quality could be enhanced through adversarial learning techniques [27].

ACKNOWLEDGMENTS

This work was funded by the European Union (ERC, HI-Audio, 101052978). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

REFERENCES

[1] D. Fitzgerald, "Automatic drum transcription and source separation," Ph.D. dissertation, Dublin Institute of Technology, Dublin, Ireland, 2004.
[2] C. Dittmar and D. Gärtner, "Real-time transcription and separation of drum recordings based on NMF decomposition," in *Proc. Int. Conf. Digit. Audio Effects*, 2014, pp. 187–194.
[3] O. Gillet and G. Richard, "ENST-Drums: An extensive audio-visual database for drum signals processing," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, 2006, pp. 156–159.
[4] T. D. Rossing, *Science of Percussion Instruments*. Singapore: World Scientific, 2000, vol. 3.
[5] R. Hennequin, A. Khlif, 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. 56, p. 2154, 2020.
[6] A. Défossez, "Hybrid spectrogram and waveform source separation," in *Proc. ISMIR Workshop Music Source Separation*, 2021.
[7] A. I. Mezza, R. Giampiccolo, A. Bernardini, and A. Sarti, "Toward deep drum source separation," *Pattern Recognit. Lett.*, vol. 183, pp. 86–91, 2024.
[8] ——, "Benchmarking music demixing models for deep drum source separation," in *Proc. IEEE Int. Symp. Internet Sounds*. Erlangen, Germany: IEEE, 2024, pp. 1–6.
[9] E. Manilow, P. Seetharaman, and B. Pardo, "Simultaneous separation and transcription of mixtures with multiple polyphonic and percussive instruments," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Barcelona, Spain: IEEE, 2020, pp. 771–775.

[10] K. W. Cheuk, Y.-J. Luo, E. Benetos, and D. Herremans, "The effect of spectrogram reconstruction on automatic music transcription: An alternative approach to improve transcription accuracy," in *Proc. Int. Conf. Pattern Recognit.* Milan, Italy: IEEE, 2020, pp. 9091–9098.

[11] K. Choi and K. Cho, "Deep unsupervised drum transcription," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, Delft, Netherlands, 2019, pp. 183–191.

[12] K. Schulze-Forster, G. Richard, L. Kelley, C. S. Doire, and R. Badeau, "Unsupervised music source separation using differentiable parametric source models," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 31, pp. 1276–1289, 2023.

[13] 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.

[14] J. Shier, F. Caspe, A. Robertson, M. Sandler, C. Saitis, and A. McPherson, "Differentiable modelling of percussive audio with transient and spectral synthesis," in *Proc. Forum Acusticum*, 2023.

[15] 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 Trans. Audio Speech Lang. Process.*, vol. 26, no. 9, pp. 1457–1483, 2018.

[16] H. Lindsay-Smith, S. McDonald, and M. Sandler, "Drumkit transcription via convolutive NMF," in *Proc. Int. Conf. Digit. Audio Effects*, York, UK, 2012.

[17] R. Vogl, M. Dorfer, G. Widmer, and P. Knees, "Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, Suzhou, China, 2017, pp. 150–157.

[18] 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.

[19] R. Vogl, G. Widmer, and P. Knees, "Towards multi-instrument drum transcription," in *Proc. Int. Conf. Digit. Audio Effects*, Aveiro, Portugal, 2018, pp. 57–64.

[20] 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.

[21] P. R. Cook, "Physically informed sonic modeling (PhISM): Percussive synthesis," in *Proc. Int. Comput. Music Conf.*, 1996, pp. 228–231.

[22] H. Han and V. Lostanlen, "Wav2shape: Hearing the shape of a drum machine," in *Proc. Forum Acusticum*, 2020, pp. 647–654.

[23] J. O. S. III and X. Serra, "PARSHL: An analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation," in *Proc. Int. Comput. Music Conf.*, 1987.

[24] T. S. Verma and T. H.-Y. Meng, "Extending spectral modeling synthesis with transient modeling synthesis," *Comput. Music J.*, vol. 24, no. 2, pp. 47–59, 2000.

[25] A. Ramires, P. Chandna, X. Favory, E. Gómez, and X. Serra, "Neural Percussive Synthesis Parameterised by High-Level Timbral Features," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Barcelona, Spain: IEEE, 2020, pp. 786–790.

[26] C. Aouameur, P. Esling, and G. Hadjeres, "Neural drum machine: An interactive system for real-time synthesis of drum sounds," *arxiv Preprint arXiv:1907.02637*, 2019.

[27] J. Nistal, S. Lattner, and G. Richard, "DRUMGAN: Synthesis of drum sounds with timbral feature conditioning using generative adversarial networks," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, Utrecht, Netherlands, 2020, pp. 590–597.

[28] J. Drysdale, M. Tomczak, and J. Hockman, "Adversarial synthesis of drum sounds," in *Proc. Int. Conf. Digit. Audio Effects*, Vienna, Austria, 2020, pp. 167–172.

[29] S. Rouard and G. Hadjeres, "CRASH: Raw audio score-based generative modeling for controllable high-resolution drum sound synthesis," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, J. H. Lee, A. Lerch, Z. Duan, J. Nam, P. Rao, P. van Kranenburg, and A. Srinivasamurthy, Eds., 2021, pp. 579–585.

[30] R. Simionato and S. Fasciani, "Sines, transient, noise neural modeling of piano notes," *Frontiers in Signal Processing*, vol. 4, p. 1494864, 2025.

[31] O. Gillet and G. Richard, "Transcription and separation of drum signals from polyphonic music," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 16, no. 3, pp. 529–540, 2008.

[32] P. Smaragdis, "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs," in *Proc. Intl. Conf. on Independent Component Analysis and Blind Signal Separation*. Grenada, Spain: Springer, 2004, pp. 494–499.

[33] C. Dittmar and M. Müller, "Reverse engineering the amen break - score-informed separation and restoration applied to drum recordings," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 24, no. 9, pp. 1535–1547, 2016.

[34] C.-Y. Cai, Y.-H. Su, and L. Su, "Dual-channel drum separation for low-cost drum recording using non-negative matrix factorization," in *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.* Tokyo, Japan: IEEE, 2021, pp. 17–22.

[35] J. Gillick, A. Roberts, J. Engel, D. Eck, and D. Bamman, "Learning to groove with inverse sequence transformations," in *Proc. Int. Conf. Mach. Learning*, Long Beach, USA, 2019, pp. 2269–2279.

[36] J. H. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *Proc. Int. Conf. Learn. Representations*, 2020.

[37] N. Masuda and D. Saito, "Synthesizer sound matching with differentiable DSP." in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, 2021, pp. 428–434.

[38] J. Engel, R. Swavely, L. H. Hantrakul, A. Roberts, and C. Hawthorne, "Self-supervised pitch detection by inverse audio synthesis," in *Proc. Workshop Self-Supervision Audio Speech 37th Int. Conf. Mach. Learn.*, 2020.

[39] B. Hayes, C. Saitis, and G. Fazekas, "Sinusoidal frequency estimation by gradient descent," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Rhodes, Greece, 2023, pp. 1–5.

[40] B. Torres, G. Peeters, and G. Richard, "Unsupervised harmonic parameter estimation using differentiable DSP and spectral optimal transport," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Seoul, South Korea: IEEE, 2024, pp. 1176–1180.

[41] C. Peladeau and G. Peeters, "Blind estimation of audio effects using an auto-encoder approach and differentiable digital signal processing," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Seoul, South Korea: IEEE, 2024, pp. 856–860.

[42] B. Hayes, J. Shier, G. Fazekas, A. McPherson, and C. Saitis, "A review of differentiable digital signal processing for music and speech synthesis," *Frontiers in Signal Processing*, vol. 3, p. 1284100, 2024.

[43] M. Kawamura, T. Nakamura, D. Kitamura, H. Saruwatari, Y. Takahashi, and K. Kondo, "Differentiable digital signal processing mixture model for synthesis parameter extraction from mixture of harmonic sounds," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Singapore, Singapore: IEEE, 2022, pp. 941–945.

[44] G. Richard, P. Chouteau, and B. Torres, "A fully differentiable model for unsupervised singing voice separation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Seoul, South Korea: IEEE, 2024, pp. 946–950.

[45] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* New Orleans, USA: IEEE, 2022, pp. 11 966–11 976.

[46] C. Hawthorne, E. Elsen, J. Song, A. Roberts, I. Simon, C. Raffel, J. H. Engel, S. Oore, and D. Eck, "Onsets and frames: Dual-objective piano transcription," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, E. Gómez, X. Hu, E. Humphrey, and E. Benetos, Eds., Paris, France, 2018, pp. 50–57.

[47] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "FiLM: Visual Reasoning with a General Conditioning Layer," *arXiv preprint arXiv:1709.07871*, no. arXiv:1709.07871, 2017.

[48] 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.

[49] A. Liutkus and R. Badeau, "Generalized Wiener filtering with fractional power spectrograms," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Brisbane, Australia: IEEE, 2015, pp. 266–270.

[50] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[51] S. Böck, F. Krebs, and M. Schedl, "Evaluating the online capabilities of onset detection methods," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, F. Gouyon, P. Herrera, L. G. Martins, and M. Müller, Eds., Porto, Portugal, 2012, pp. 49–54.

[52] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, "MIR_EVAL: A transparent implementation of common MIR metrics," in *Proc. Int. Soc. Music Inf. Retrieval Conf.*, Taipei, Taiwan, 2014, pp. 367–372.

[53] Y. Mitsufuji, G. Fabbro, S. Uhlich, and F.-R. Stöter, "Music demixing challenge2021," *arXiv preprint arXiv:2108.13559*, 2021.

[54] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR–half-baked or well done?" in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Brighton, UK: IEEE, 2019, pp. 626–630.

[55] M. Torcoli, T. Kastner, and J. Herre, "Objective Measures of Perceptual Audio Quality Reviewed: An Evaluation of Their Application Domain

Dependence," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 29, pp. 1530–1541, 2021.

[56] K. Schulze-Forster, C. S. J. Doire, G. Richard, and R. Badeau, "Weakly informed audio source separation," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.* IEEE, 2019, pp. 273–277.

[57] L. Maia, P. D. de Tomaz Jr., M. Fuentes, M. Rocamora, L. W. P. Biscainho, M. V. M. da Costa, and S. Cohen, "A novel database of brazilian rhythmic instruments and some experiments in computational rhythm analysis," in *Proc. Audio Eng. Soc. Latin Amer. Conf.*, Montevideo, Uruguay, 2018.