Audio Engineering Society

Conference Paper

Presented at the AES 2026 International Conference on
Automotive Audio
2026 July 29–31, Detroit, MI, USA

This paper was peer-reviewed as a complete manuscript for presentation at this conference. This paper is available in the AES E-Library (http://www.aes.org/e-lib), all rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

Improving Low-Latency Automatic Drum Transcription for Automotive Applications

Philipp Weyers[1], Christian Uhle[1,2], Meinard Müller[1,2], and Matthias Lang[1]

[1]*Fraunhofer Institute for Integrated Circuits (IIS), Germany*
[2]*International Audio Laboratories Erlangen - A joint institution of the Friedrich-Alexander Universität Erlangen-Nürnberg (FAU) and Fraunhofer IIS*

Correspondence should be addressed to Philipp Weyers (philipp.weyers@iis.fraunhofer.de)

ABSTRACT

In automotive audio systems, musical beats and drum events can control synchronized in-cabin experiences such as ambient lighting and music-driven visual effects. Compared to beat tracking, automatic drum transcription (ADT) offers richer control signals by detecting and classifying drum onsets of multiple drum classes (e.g., bass drum, snare drum, and hi-hat), enabling more precise and musically meaningful synchronization. Deploying ADT in vehicles, however, requires low latency, computational efficiency, and robust performance for various input signals. This paper investigates improvements to low-latency ADT suitable for automotive deployment, using the Separate-Tracks-Annotate-Resynthesize Drums (STAR Drums) dataset and a block-based processing strategy that achieves an average detection delay of around 60 ms. We explore three strategies: (1) lightweight architecture modifications inspired by recent advances in image classification, combined with a temporal convolutional network (TCN); (2) re-rendering STAR Drums to increase drum timbre diversity and augmenting the re-synthesized drum stems; and (3) refinement training with pseudo labels obtained from source-separated mixtures. Our results show that data augmentation and increased drum timbre diversity yield modest performance gains, whereas pseudo-label refinement provides the largest effect, with up to 18 % relative improvement in global F-measure. In the real-time eight-class setting, our best model achieves a global F-measure of 0.77 on MDB Drums, competitive with state-of-the-art offline systems, demonstrating that accurate and efficient ADT is feasible for automotive deployment.

1 Introduction

In the automotive sector, music information retrieval (MIR) offers numerous applications, including the control of in-car algorithms based on properties of the input signal. A genre classifier, for example, can adapt processing to different expectations of classical versus pop listeners. Furthermore, algorithms providing information about the beat can enhance the in-cabin experience by enabling synchronized ambient lighting and music-driven visual effects. While beat tracking can provide tempo estimations, automatic drum transcription (ADT) [1], which aims at identifying and classifying drum onsets in music signals, offers extended possibilities by providing transcriptions of multiple drum classes which can be regarded as a toolkit for creative visualizations. Bass drum and snare drum events may drive beat-synchronized ambient lighting, while

less frequently played classes such as toms and cymbals can trigger more distinctive visual cues, often coinciding with structural changes or climactic moments in the music. Furthermore, statistics derived from hi-hat activity, such as inter-onset intervals, can be used to adapt visualization characteristics to reflect, for example, high-energy tracks with dense hi-hat patterns versus more relaxed songs with sparse hi-hat activity.

Here, we focus on improving ADT for automotive applications, which, besides high transcription accuracy, demands a low transcription delay and low computational complexity suitable for embedded systems.

Similarly to other MIR disciplines, deep neural networks (DNNs) have increased ADT performance over the last decade, with state-of-the-art (SOTA) models reaching global F-measures around 0.8 for drum transcription in the presence of melodic instruments (DTM) when transcribing the whole drum kit [2, 3, 4]. However, many works [2, 3, 5, 6, 7] use context windows of several seconds and architectures with limited online capability, such as transformers or bi-directional recurrent neural networks (RNNs), posing conceptual constraints for real-time applications.

Several pre-DNN era works focused on real-time drum transcription of drum-only recordings (DTD), which is less demanding than DTM due to the absence of sounds of melodic instruments and singing. For instance, Dittmar and Gärtner [8] leveraged non-negative matrix factorization, while Kaliakatsos-Papakostas et al. [9] developed a computationally efficient transcription model using bandpass filtering and thresholding. Miron et al. [10] based their DTD transcription algorithm on K-means clustering. Later, Stables et al. [11] compared bi-directional and real-time-capable unidirectional RNNs for DTD and DTM, but limited the transcription to three classes (bass drum, snare drum, and hi-hat). More recently, Weber et al. [12] investigated dynamic few-shot learning (FSL) for DTM with low delay, transcribing up to ten classes, but their performance does not match that of recent SOTA works.

In this work, we address the existing gap in real-time DTM research by targeting performance that is competitive with SOTA offline algorithms, adopting a block-based processing strategy with low look-ahead, similar to [12]. Vatakis and Spence [13] investigated the just noticeable difference for audiovisual asynchrony in music and speech videos, resulting in values from 100 to 130 ms for guitar and piano. Rhythm instruments were not included. According to Turchet et al. [14], a delay up to 60 ms allows for synchronization between musicians in live performances, which we adopt as a conservative upper bound for the acceptable delay when creating visualizations for music. Assuming that the audio signal itself will be delayed by 10 to 20 ms due to DSP processing common in modern automotive audio systems, which reduces the effective transcription delay, we consider an average algorithmic delay of up to 70 ms suitable for automotive applications.

Additionally, we propose architecture modifications to the established combination of a convolutional neural network (CNN) followed by a RNN or attention-based network as used in [6, 2, 3, 5, 15, 7, 4]. The proposed modifications include replacing the RNN or self-attention layers with a TCN [16] for modeling temporal relationships. Furthermore, we incorporate several modifications proposed for image classification by Liu et al. [17] into the CNN. As a result, the proposed model solely consists of convolutional layers and allows for both low implementation complexity and low computational complexity.

Besides architecture modifications, several works have focused on providing higher quality training data for ADT [18, 19, 5]. In [5], the STAR Drums dataset, consisting of synthesized drum sounds alongside recordings of instruments and singing, proved to offer competitive training performance while at the same time featuring a fully automatic dataset creation pipeline and a separate drum stem. We leverage both features to experiment with increasing the diversity of the included re-synthesized drum timbres by re-rendering the dataset, and we apply augmentation to the drum stem during training. Additionally, we systematically investigate the use of pseudo labels in STAR Drums to refine models trained on synthetic data.

The main contributions of this work are:

- We further investigate and improve real-time DTM by proposing architecture changes inspired by recent progress in image classification, offering strong performance while at the same time reducing computational complexity.
- We explore possibilities to improve the training data provided in the STAR Drums dataset by re-rendering its content using more diverse drum timbres and by examining data augmentation strategies applied to the provided separate drum stem.

- We investigate refinement training using pseudo labels from the original mixtures, improving the generalization capabilities of models initially trained on accurately annotated synthesized data.

2 Methods

2.1 Pre-Processing

We use a 1024-point short-time Fourier transform (STFT) with a hop length of 512 samples at a sample rate of 48 kHz, resulting in a frame spacing of 10.7 ms. In line with previous works [25, 2, 4], we use monaural input signals. The input to the model consists of mel spectrograms with 96 frequency bands and an upper cut-off frequency of 16 kHz. The model processes blocks of 40 frames with a hop length of one frame.

2.2 Model Architecture

We follow the principles of established ADT architectures [2, 4, 5, 15, 7], using three network components:

1. A CNN that directly learns features from input spectrograms.
2. A network capable of modeling temporal patterns, e.g., a RNN or a self-attention network.
3. A dense network to reduce the output dimensionality to the number of classes.

As a baseline, we use a similar network as in [5], referred to as `CRNN`. It consists of four convolutional layers including convolutions with 3x3 kernels and 128 convolutional channels, instance normalization, exponential linear unit (ELU) activation, and 2x1 max-pooling, with the first dimension being frequency and the second dimension time. The CNN is followed by three unidirectional GRU layers with 60 hidden units and a dense network with two fully connected layers with tanh activation and a hidden dimension of 128, reducing the output to the number of classes.

In addition to `CRNN`, we take inspiration from Liu et al. [17], who modified convolutional networks based on recent advancements in transformer and conformer architectures. We create a second network architecture by using the proposed ConvNeXt architecture as the basis for the CNN, including the following components: A stem layer downsamples the input to the network with a 4x1 kernel and stride of 4. To avoid losing information along the time axis, kernel and stride of the stem layer act only along the frequency axis, in contrast to [17], where the authors process images rather than time series. The following layers consist of ConvNeXt-blocks, as shown in Fig. 1 and proposed in [17]. Each ConvNeXt-block features depthwise convolution (a grouped convolution with the number of groups equaling the number of channels) with a large kernel size of 7x7, an inverted bottleneck with the hidden dimension being four times the input and output dimension, gaussian error linear unit (GELU) activation, and layer normalization.

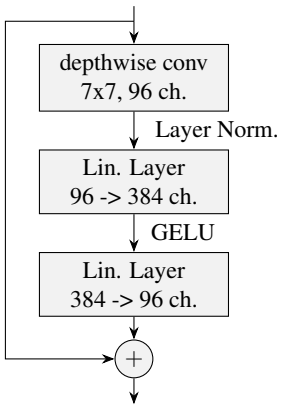


Fig. 1: ConvNeXt block [17] with depthwise convolution, inverted bottleneck, layer normalization, GELU activation, and residual connection.

In our experiments, increasing the number of convolutional channels in each ConvNeXt block with increasing network depth was not beneficial. Consequently, we use five ConvNeXt blocks with 96 convolutional channels each. Using a constant number of channels allows us to remove downsampling layers between ConvNeXt blocks, further reducing complexity. To preserve the frequency and time dimensions, we remove the global average pooling after the ConvNeXt-blocks as done in [17]. We refer to this reduced ConvNeXt network as SmallConvNeXt in the following.

Instead of GRU layers, the ConvNeXt-based network uses a TCN [16] to model temporal relationships. TCNs have been applied to beat tracking in [21], are extensively evaluated in [20], and, to the best of our

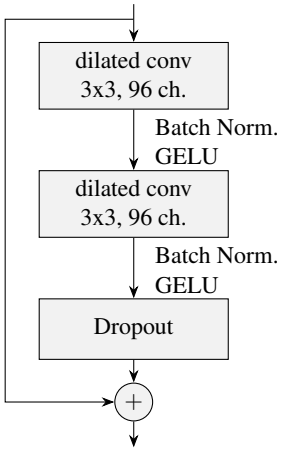


Fig. 2: TCN block [20] consisting of two dilated convolutions, batch normalization, GELU activation, dropout layer, and residual connection.

knowledge, have not yet been applied to ADT. Besides their promising performance in [21], TCNs offer improved parallelization capabilities compared to RNNs [20], making them an attractive choice, especially when computational efficiency is important.

TCNs leverage dilated convolutions with the dilation factor increasing exponentially in each layer, increasing the temporal context accessible to each time step. We use three layers with dilations of 1, 2, and 4, resulting in a receptive field of 29 frames and consequently covering more than half of the block length of 40 frames.

Each TCN block contains two dilated convolutional layers with batch normalization and GELU activation, as shown in Fig. 2 and proposed in [20]. Unlike Bai et al. [20], we use non-causal convolutions, as at each time step the model has access only to the current block of 40 frames and may therefore use the data within the current block in a non-causal way without violating the causality of the overall system. From our experience, providing no look-ahead results in strongly limited transcription performance, while a limited look-ahead significantly improves performance.

Both networks, `CRNN` and the proposed model leveraging ConvNeXt blocks and a TCN, referred to as `TsConvN` from here on, use a dense network including two linear layers to reduce the number of output features to the number of drum classes (eight). Finally, a sigmoid function converts the output values to probabilities.

Table 1: Number of parameters and MAC operations for a single forward pass.

Network	# Parameters [M]	MAC units [M]
`CRNN` [5]	1.02	542
`TsConvN`	0.74	392

Table 1 lists the number of parameters and the number of multiply-accumulate (MAC) operations for a single forward pass of `CRNN` and `TsConvN` computed using the PyTorch package `thop`. `TsConvN` uses approximately 70 % of the parameters and MACs of `CRNN`, which significantly reduces both the memory footprint and the computational load. The computational demand of `TsConvN` corresponds to 36.8 GMAC/s at a framerate of 93.8 Hz, which is well within the capabilities of current edge deep-learning accelerators for automotive applications [22], confirming the suitability of the proposed networks for automotive deployment.

2.3 Peak-Picking

We follow the approach in [12], using a peak-picking algorithm with two thresholds to identify onsets already during a rising probability curve rather than detecting local maxima, which would cause additional delay. While the first threshold is applied directly to the onset probability, the second threshold allows onsets only if the onset probability curve is currently rising, which we evaluate by comparing the output probability to a smoothed version of the output probability.

We fit the threshold parameters on the validation split of the training dataset and use the same thresholds for all classes.

3 Data, Training, and Evaluation

3.1 Datasets

Weber et al. [5] created STAR Drums by source-separating full mixture recordings featuring drum sounds and recordings of melodic instruments and singing into a drum and a non-drum stem. The source-separated drum stem was transcribed using an ADT

algorithm published alongside [6], providing an estimated annotation. This estimated annotation serves to synthesize a drum stem which is similar to the original drum stem. As the estimated annotation perfectly matches the content of the re-synthesized drum stem, the annotation is then regarded as a reference annotation. Finally, the authors mixed the re-synthesized drum stem with the non-drum stem, resulting in the final mix used for ADT training. STAR Drums contains 124.5 h of annotated audio as well as MIDI files and MIDI mappings used to synthesize the drum stems, facilitating further modifications.

We chose the STAR Drums dataset for training as, according to Weber et al. [5], trained models exhibit similar performance to models trained on datasets not containing synthesized sounds, while STAR Drums at the same time offers a high number of classes and a separate drum stem. The experiments in Section 4.2 directly make use of the separate drum stem by augmenting the drum sounds, while the investigations in Section 4.1 leverage the fully accessible creation pipeline to recreate the dataset with more diverse drum timbres.

For testing, we use the publicly available, manually annotated datasets MDB Drums (MDB) [23], ENST-Drums (ENST) [24], RBMA13 (RBMA) [25], and the RWC-P sub-collection of the recently re-published RWC Dataset [26], all containing drum sounds and sounds of melodic instruments and featuring a total duration of 0.4 to 6.8 h. While MDB, RBMA, and RWC-P contain full mixes with real recordings of melodic instruments and singing, ENST contains real recordings of three drummers accompanied by partly synthesized backing tracks without singing.

We remove seven items from RWC-P for which we identified obvious annotation errors. Incorrect annotations were also reported by Ishizuka et al. [15]. Additionally, we remove all items without drums from RWC-P and RBMA.

By using four test datasets, we aim to offer performance metrics covering different genres and recording settings to satisfy the need for robustness to different input signals in automotive audio applications.

3.2 Training Procedure

We train the model such that it predicts an onset at the 34^{th} frame of the current block consisting of 40 frames. This results in a delay of seven STFT frames (74.6 ms), including the delay caused by the STFT itself. As outlined in Section 2.3, the peak-picking algorithm aims at detecting onsets during the attack phase of the output probability, ideally detecting the onset before the probability reaches its peak.

To address class imbalances, we apply undersampling of training samples without drum activity and class weights as explained in [5]. Additionally, we augment the mel spectrograms following [27], and use label-smoothing and target widening as proposed in [5].

We train using AdamW [28] with a batch size of 256 and an initial learning rate of $3 \cdot 10^{-4}$ controlled by a plateau learning-rate scheduler monitoring the validation F-measure. Training runs for up to 30 epochs with early stopping based on the validation F-measure. For refinement, we initialize from the pre-trained model and use the same hyperparameters, except for a reduced initial learning rate of $1 \cdot 10^{-4}$.

3.3 Evaluation

We transcribe eight classes following the mapping established by Vogl et al. [6]: bass drum (BD), snare drum (SD), hi-hat (HH), tom (TT), cymbals (CY), ride cymbals (RD), bell (BE), and clave (CL). All experiments are performed for DTM. The detailed eight-class mapping, including the full drum kit, enables extended visualization possibilities in contrast to smaller three or five-class mappings while at the same time allowing a comparison to related works.

The Python package *mir_eval* [29] is used to identify true and false detections with a tolerance window of 50 ms. Normally, a symmetric tolerance window centered at the ground-truth onset time is applied. However, as it is not possible to detect an onset too early under real-time constraints, we follow Weber et al. [12] and shift the tolerance window by 50 ms when evaluating under real-time conditions. This results in the same total length (100 ms) of the tolerance window and keeps the results comparable to works using a symmetric 50 ms window in non-real-time scenarios, such as [2, 3, 5].

While we acknowledge that a subjective in-car evaluation would be preferable, this would go beyond the scope of this paper and its significance would be limited to the specific visualization design, which is inherently based on artistic decisions. Instead, we rely on the

global F-measure as the established primary performance metric, computed using micro-averaging [30]. As a high F-measure can only be reached with both high recall and high precision, we believe that a model with higher F-measure is also better suited for creating visualizations, and adjustments towards higher recall or precision can be easily achieved by modifying the onset threshold, see Section 2.3.

4 Experiments

4.1 Re-rendering Drum Stems of STAR Drums

To investigate whether increased diversity of drum timbres improves the generalization capabilities of the models, we re-render the drum stems provided in STAR Drums using the original MIDI files and virtual instruments as proposed in [5], but now making use of the internal sound-shaping functionality of *Kontakt 7*, which provides audio effects such as equalizer, transient designer, tape saturation, as well as individual level controls of instrument microphones and room microphones. To allow for reproducibility, we use the existing mixer presets, which provide five to eleven different sets of effect settings for each individual drum kit. Specifically, we distribute the subset of items assigned to one virtual drum kit evenly across all provided mixer presets.

4.2 Augmenting the Drum Stems of STAR Drums

In contrast to re-rendering the drum sounds of STAR Drums, we also experiment with augmenting the existing drum stems during training, which allows us to systematically and efficiently tune augmentation parameters without the need for time-intensive re-rendering. To simulate some of the effects commonly applied during music production, we apply the following augmentations:

- Low-pass and high-pass filtering,
- Adding white noise,
- Pitch-shifting,
- Saturation.

All augmentation parameters, including the probability of being applied to a given training sample, are tuned on the validation split of STAR Drums using the original mix with pseudo labels (see Section 4.3). When tuning on the mix with synthesized drum sounds, parameters consistently converged to no or only very mild augmentation. This is expected, as the synthetic validation split contains the same drum sounds as the training split (on different items). Consequently, any augmentation tends to lower validation performance by moving the distribution of the training data away from the validation distribution. In contrast, using the original mix, which contains the original drum sounds with higher drum timbre diversity, allows us to tune augmentation parameters in a meaningful way, with increased validation performance reflecting improved generalization, despite noisy labels.

Low-pass and high-pass filtering is applied with a probability of 70 %, using randomly selected parameters: cutoff frequencies between 20 and 50 Hz and between 10 and 16 kHz, respectively, with 2^{nd}- or 4^{th}-order filters. We do not include peak filtering, as randomly chosen center frequencies would rarely correspond to musically meaningful resonances.

During music production, filtering is typically applied separately to each drum track, with each drum instrument recorded using a dedicated microphone. Here, we only have access to the full drum mix rather than instrument-wise stems. Consequently, the proposed filtering can only be regarded as a rough approximation of the workflow used during music production.

The level of white noise is randomly chosen between -20 and $-10\,\text{dB}_{\text{FS}}$ and applied to 70 % of the training samples. Adding white noise simulates common signal degradations, especially occurring in amateur productions.

We pre-compute pitch-shifted versions of the drum stems with ± 1 semitone, which allows us to use high-quality plug-ins and avoids quality degradation due to highly workload-optimized pitch-shifting algorithms that would be required for on-the-fly processing during training. One of the pitch-shifted drum stems is chosen randomly for 40 % of the training samples.

Saturation of the input is applied to all training samples and achieved according to

$$y = \tanh(d \cdot x) \tag{1}$$

with

$$d = 10^{\frac{d_{\text{dB}}}{20}}, \tag{2}$$

where x is the input time signal, d_{dB} is a parameter controlling the amount of saturation, and y is the saturated output signal. The output is rescaled to match the input energy, and d_{dB} is drawn uniformly from $[-20, -10]$ dB.

4.3 Refinement Training Using Pseudo Labels

Weber et al. [5] highlight the possibility of using the original, unprocessed mix for training instead of the mix including the re-synthesized drum signal. This introduces pseudo labels, as the provided labels do not perfectly match the drum sounds included in the original mix. Experiments in [5] indicate that, despite their imperfect nature, models trained with pseudo labels from STAR Drums can deliver competitive performance.

Here, we hypothesize that refinement training using pseudo labels can be beneficial, as the drum sounds of the original mix are more diverse than the re-synthesized drum sounds, potentially improving generalization capabilities. By using pseudo labels solely for refinement of models initially trained on accurate synthetic data, we aim to limit the negative influence of imperfect labels. This approach differs from the experiments of Weber et al. [5], who combined training with pseudo labels and accurately annotated synthesized data in a single training stage.

The labels in STAR Drums that we use as pseudo labels for refinement are generated using the model of Vogl et al. [6], which is an ensemble of five networks, three of which were originally trained on MDB, ENST, and RBMA. Consequently, our refined models may indirectly inherit dataset-specific biases of these three benchmarks from the teacher ensemble. This poses a potential source of slight optimistic bias in the reported results on MDB, ENST, and RBMA, even though the corresponding audio and labels are never used directly when training our models. In contrast, RWC-P was neither used in the training of the teacher ensemble nor in the construction of the STAR Drums data and therefore serves as an unbiased test set for assessing the effect of pseudo-label refinement.

5 Results and Discussion

Table 2 lists results for the `CRNN` and `TsConvN` architectures in terms of global F-measure on the STAR Drums test split, MDB, ENST, RBMA, and RWC-P. Both re-rendering the re-synthesized drum stems in STAR Drums and applying drum-stem augmentation generally improve performance on datasets with recorded drums. For example, `TsConvN` on MDB increases from 0.716 (Base) to 0.724 with re-rendered drum stems and to 0.731 with augmentation. The only exception is `CRNN` on RBMA, where re-rendering slightly reduces performance from 0.608 to 0.600.

Using re-rendered drum stems and augmentation simultaneously does not yield further gains, except on RWC-P (`TsConvN` increases from 0.675 to 0.692) and can even reduce performance, likely because augmentation hyperparameters were tuned on the original (non–re-rendered) STAR Drums and become overly strong when combined with re-rendering. As expected, performance on the STAR Drums test set decreases when using these strategies, since overfitting to the limited drum timbre diversity in the original STAR Drums version becomes less likely and the model must generalize to a wider range of drum sounds. Consistently, the combination of re-rendering and augmentation gives the lowest STAR Drums test performance for both networks.

Refinement training with pseudo labels yields the lowest scores on the STAR Drums test split but the largest improvements on MDB, ENST, RBMA, and RWC-P, clearly outperforming models trained only on re-synthesized drums or with augmentation alone. For instance, `TsConvN` on RBMA increases from 0.551 (Base) to 0.648 with pseudo-label refinement, a relative improvement of about 18 %, and on RWC-P from 0.652 to 0.722. The highest overall F-measure is 0.765 for `TsConvN` on MDB. As outlined in Section 4.3, the performance after pseudo-label refinement on MDB, ENST, and RBMA might be slightly optimistic due to the teacher ensemble having been trained on these datasets. However, the substantial improvement on RWC-P, which was not involved in the teacher training or in the construction of STAR Drums, and the identical ranking of all experiment conditions on RWC-P and the other test datasets indicate that pseudo-label refinement yields genuine improvements in generalization and that any optimistic bias on MDB, ENST, and RBMA is likely limited.

Overall, the results suggest that the re-synthesis process in STAR Drums still limits drum timbre diversity compared to unprocessed recordings and that noisy labels, containing incorrect annotations, can nevertheless

Table 2: Global F-measures on MDB, ENST, RBMA, RWC-P, and the STAR Drums test split when transcribing eight classes.

Network	Re-rendered STAR Drums	Augmentation Drum stem	Pseudo Lab. Ref.	MDB	ENST	RBMA	RWC-P	STAR Test
CRNN [5]	✗	✗	✗	0.720	0.711	0.608	0.677	0.776
	✓	✗	✗	0.729	0.720	0.600	0.681	0.770
	✗	✓	✗	0.726	0.713	0.624	0.693	0.754
	✓	✓	✗	0.719	0.710	0.619	0.696	0.747
	✗	✗	✓	0.748	0.760	**0.650**	**0.724**	0.714
	✓	✓	✓	0.750	0.759	0.629	**0.724**	0.711
TsConvN	✗	✗	✗	0.716	0.693	0.551	0.652	**0.792**
	✓	✗	✗	0.724	0.704	0.570	0.670	0.779
	✗	✓	✗	0.731	0.712	0.591	0.675	0.764
	✓	✓	✗	0.724	0.712	0.591	0.692	0.758
	✗	✗	✓	**0.765**	0.755	0.648	0.722	0.711
	✓	✓	✓	0.759	**0.762**	0.638	0.714	0.709

be effective for refining models pre-trained on accurate synthetic annotations.

While TsConvN achieves lower performance than CRNN on MDB, ENST, RWC-P, and RBMA in the base condition (without re-rendered drum sounds, augmentation, or pseudo-label refinement) it outperforms CRNN on the STAR Drums test split (0.792 vs. 0.776), indicating a high potential of this architecture. The performance gap on external datasets can be closed by using pseudo-label refinement, with TsConvN outperforming CRNN on MDB (0.765 vs. 0.750) and achieving very similar performance on ENST, RWC-P and RBMA. Consequently, the TsConvN architecture is an attractive alternative to established ADT architectures, providing SOTA performance while significantly reducing the number of parameters and computational workload.

Figure 3 shows the per-class F-measure for TsConvN on RWC-P. Bass drum performance is already high for the Base system and remains almost unchanged across all conditions. In contrast, snare drum, hi-hat, cymbals, and ride cymbals clearly benefit from pseudo-label refinement, and to a lesser extent from re-rendering and augmentation. Tom and clave also show small improvements but, together with bell, remain at comparatively low absolute performance for all configurations, suggesting that the corresponding timbres in STAR Drums (even after re-rendering and augmentation) and in the pseudo-labeled mixtures still differ substantially from those in RWC-P, or that the pseudo labels for these classes are too noisy.

Table 3: Detection delay in ms of CRNN and TsConvN models refined on pseudo labels on MDB, ENST, RBMA, and STAR test.

Network	MDB	ENST	RBMA	STAR Test
CRNN	53.4	62.9	59.8	61.7
TsConvN	55.2	64.1	61.0	63.4

Table 3 lists the average detection delay of the CRNN and TsConvN models refined on pseudo labels. Detection delay is computed as the mean signed difference between predicted and ground-truth onset times, with positive values indicating late detections. The resulting values range from 53.4 to 64.1 ms, confirming that the peak-picking algorithm, which identifies onsets during a rising probability curve without waiting for the curve to reach its maximum, can slightly reduce the effective detection delay compared to the theoretical delay of 74.6 ms, see Section 3.2. Target widening, applied during training including a step-wise increase of the value of the ground-truth annotation already two frames ahead of the drum sound, may also contribute to the reduced delay.

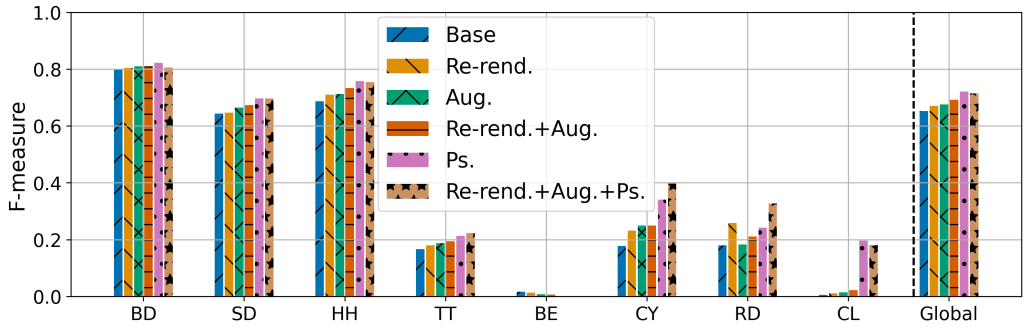


Fig. 3: F-measure per class on RWC-P for `TsConvN` trained on: original STAR Drums (Base), re-rendered drum stems (Re-rend.), augmented drum stems (Aug.), re-rendered and augmented drum stems (Re-rend.+Aug.), refined with pseudo labels (Ps.), and the combination of all three strategies (Re-rend.+Aug.+Ps.).

Table 4: Results in terms of global F-measure on MDB, ENST, and RBMA when transcribing five classes without real-time constraints.

Network	MDB	ENST	RBMA
ADTOF [2]	**0.810**	0.780	0.650
`TsConvNnRT`	0.790	**0.789**	**0.675**

For comparison with SOTA offline models, Table 4 reports five-class, non–real-time global F-measures. The ADTOF results are taken from [2]. Our `TsConvNnRT` model is a non-real-time variant of `TsConvN` trained on eight classes (including pseudo label refinement) but evaluated with the same five-class mapping and a symmetric 50 ms tolerance window to enable a direct comparison to ADTOF. It uses a block length of 250 frames, a look-ahead of 125 frames (corresponding to an algorithmic delay of 1.33 s), as well as five instead of three TCN layers to increase the receptive field and compensate for the larger block length. `TsConvNnRT` outperforms ADTOF on ENST and RBMA, with F-measures of 0.789 and 0.675, respectively, while ADTOF remains better on MDB.

Together with the real-time results in Table 2 for eight classes, this shows that the proposed `TsConvN` architecture is competitive with state-of-the-art offline methods and that the remaining gap in Table 2 is largely due to the more challenging eight-class, real-time evaluation scenario.

6 Summary

In this paper, we explored several improvements to real-time ADT, including the proposed `TsConvN` architecture, which matches the performance of an established ADT architecture with a reduced number of parameters and lower computational workload. Furthermore, we investigated modifications to the STAR Drums dataset that increase the diversity of the included drum timbres and data augmentation applied to the drum stem of STAR Drums. Our experiments indicate limited potential when using re-rendered drum sounds or drum stem augmentation alone, with only minor performance gains.

On the other hand, refinement training using pseudo labels significantly improves performance for both network architectures, providing a low-effort and effective enhancement to real-time ADT. We demonstrated that our models deliver performance competitive with SOTA offline algorithms with global F-measures up to 0.77 on MDB Drums when transcribing eight classes.

The low delay of the proposed models, together with a computational cost well below the capabilities of available deep learning accelerators for automotive use, suggests the suitability for automotive deployment. In future work, we plan to evaluate if the current SOTA ADT performance is sufficient for creating visualizations and in particular, if the F-measure as established performance metric coincides well with the perceived quality and musicality of visualizations. Additionally, we aim to deploy the model on target hardware to measure the workload consumption under real-life conditions.

References

[1] Wu, C., Dittmar, C., Southall, C., Vogl, R., Widmer, G., Hockman, J., Müller, M., and Lerch, A., "A Review of Automatic Drum Transcription," *IEEE ACM Trans. Audio Speech Lang. Process.*, 26(9), pp. 1457–1483, 2018.

[2] Zehren, M., Alunno, M., and Bientinesi, P., "High-quality and reproducible automatic drum transcription from crowdsourced data," *Signals*, 4(4), pp. 768–787, 2023.

[3] Zehren, M., Alunno, M., and Bientinesi, P., "In-Depth Performance Analysis of the ADTOF-Based Algorithm for Automatic Drum Transcription," in *Proc. 25th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, pp. 1060–1067, 2024.

[4] Weyers, P., Uhle, C., Müller, M., and Lang, M., "Understanding Performance Limitations in Automatic Drum Transcription," in *Proc. 26th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, pp. 582–588, 2025.

[5] Weber, P., Uhle, C., Müller, M., and Lang, M., "STAR Drums: A Dataset for Automatic Drum Transcription," *Trans. Int. Soc. Music Inf. Retr. (TISMIR)*, 8(1), pp. 248–264, 2025.

[6] Vogl, R., Widmer, G., and Knees, P., "Towards multi-instrument drum transcription," in *Proc. 21st Int. Conf. on Digit. Audio Effects (DAFx)*, 2018.

[7] Ishizuka, R., Nishikimi, R., and Yoshii, K., "Global structure-aware drum transcription based on self-attention mechanisms," *Signals*, 2(3), pp. 508–526, 2021.

[8] Dittmar, C. and Gärtner, D., "Real-Time Transcription and Separation of Drum Recordings Based on NMF Decomposition." in *Proc. 17th Int. Conf. on Digit. Audio Effects (DAFx)*, pp. 187–194, 2014.

[9] Kaliakatsos-Papakostas, M. A., Floros, A., Vrahatis, M. N., and Kanellopoulos, N., "Real-time drums transcription with characteristic bandpass filtering," in *Proc. Audio Mostly: 7th Conf. on Interaction with Sound*, pp. 152–159, ACM, 2012.

[10] Miron, M., Davies, M. E., and Gouyon, F., "Improving the real-time performance of a causal audio drum transcription system," in *Proc. Sound and Music Computing Conf.*, pp. 402–407, 2013.

[11] Stables, R., Hockman, J., and Southall, C., "Automatic Drum Transcription using Bi-directional Recurrent Neural Networks." in *Proc. 17th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, pp. 591–597, 2016.

[12] Weber, P., Uhle, C., Müller, M., and Lang, M., "Real-Time Automatic Drum Transcription using Dynamic Few-Shot Learning," in *Proc. IEEE 5th Int. Symp. on the Internet of Sounds (IS2)*, pp. 1–8, 2024.

[13] Vatakis, A. and Spence, C., "Audiovisual synchrony perception for speech and music assessed using a temporal order judgment task," *Neuroscience Letters*, 393(1), pp. 40–44, 2006.

[14] Turchet, L., Fischione, C., Essl, G., Keller, D., and Barthet, M., "Internet of Musical Things: Vision and Challenges," *IEEE Access*, 6, pp. 61994–62017, 2018.

[15] Ishizuka, R., Nishikimi, R., Nakamura, E., and Yoshii, K., "Tatum-Level Drum Transcription Based on a Convolutional Recurrent Neural Network with Language Model-Based Regularized Training," in *Proc. IEEE Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pp. 359–364, 2020.

[16] van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. W., and Kavukcuoglu, K., "WaveNet: A Generative Model for Raw Audio," in A. W. Black, editor, *Proc. 9th ISCA Speech Synthesis Workshop, SSW*, p. 125, ISCA, 2016.

[17] Liu, Z., Mao, H., Wu, C., Feichtenhofer, C., Darrell, T., and Xie, S., "A ConvNet for the 2020s," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 11966–11976, IEEE, 2022.

[18] Wei, I., Wu, C., and Su, L., "Improving Automatic Drum Transcription Using Large-Scale Audio-to-Midi Aligned Data," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, pp. 246–250, 2021.

[19] Zehren, M., Alunno, M., and Bientinesi, P., "ADTOF: A large dataset of non-synthetic music for automatic drum transcription," in *Proc. 22nd Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, pp. 818–824, 2021.

[20] Bai, S., Kolter, J. Z., and Koltun, V., "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," *CoRR*, 2018.

[21] Davies, M. E. P. and Böck, S., "Temporal convolutional networks for musical audio beat tracking," in *Proc. 27th Eur. Signal Process. Conf. EUSIPCO*, pp. 1–5, IEEE, 2019.

[22] Alam, S., Yakopcic, C., Wu, Q., Barnell, M., Khan, S., and Taha, T. M., "Survey of deep learning accelerators for edge and emerging computing," *Electronics*, 13(15), p. 2988, 2024.

[23] Southall, C., Wu, C.-W., Lerch, A., and Hockman, J., "MDB Drums: An annotated subset of MedleyDB for automatic drum transcription," *presented at 18th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, 2017.

[24] Gillet, O. and Richard, G., "ENST-Drums: an extensive audio-visual database for drum signals processing," in *Proc. 7th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, pp. 156–159, 2006.

[25] Vogl, R., Dorfer, M., Widmer, G., and Knees, P., "Drum Transcription via Joint Beat and Drum Modeling Using Convolutional Recurrent Neural Networks," in *Proc. 18th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, pp. 150–157, 2017.

[26] Balke, S., Zeitler, J., Arifi-Müller, V., McFee, B., Nakano, T., Goto, M., and Müller, M., "RWC Revisited: Towards a Community-Driven MIR Corpus," *Trans. Int. Soc. Music Inf. Retr. (TISMIR)*, 9(1), pp. 21–35, 2026.

[27] Park, D. S., Chan, W., Zhang, Y., Chiu, C., Zoph, B., Cubuk, E. D., and Le, Q. V., "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. 20th Interspeech*, pp. 2613–2617, ISCA, 2019.

[28] Loshchilov, I. and Hutter, F., "Decoupled Weight Decay Regularization," in *Proc. 7th Int. Conf. on Learning Representations (ICLR)*, 2019.

[29] Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., and Ellis, D. P. W., "MIR_EVAL: A Transparent Implementation of Common MIR Metrics." in *Proc. 15th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, pp. 367–372, 2014.

[30] Takahashi, K., Yamamoto, K., Kuchiba, A., and Koyama, T., "Confidence interval for micro-averaged F_1 and macro-averaged F_1 scores," *Appl. Intell.*, 52(5), pp. 4961–4972, 2022.